AGH

AGH UNIVERSITY OF SCIENCE AND TECHNOLOGY

FACULTY OF METALS ENGINEERING AND INDUSTRIAL COMPUTER SCIENCE

MULTISCALE MODELLING DIVISION

MULTISCALE MODELLING – PROJECT REPORT

Author:	*Grzegorz Samolej*
Subject:	*Multiscale Modelling*
Course instructor:	*Prof. Dr Hab. Eng. Łukasz Madej*

Kraków, *05.02.2021*

Table of contents

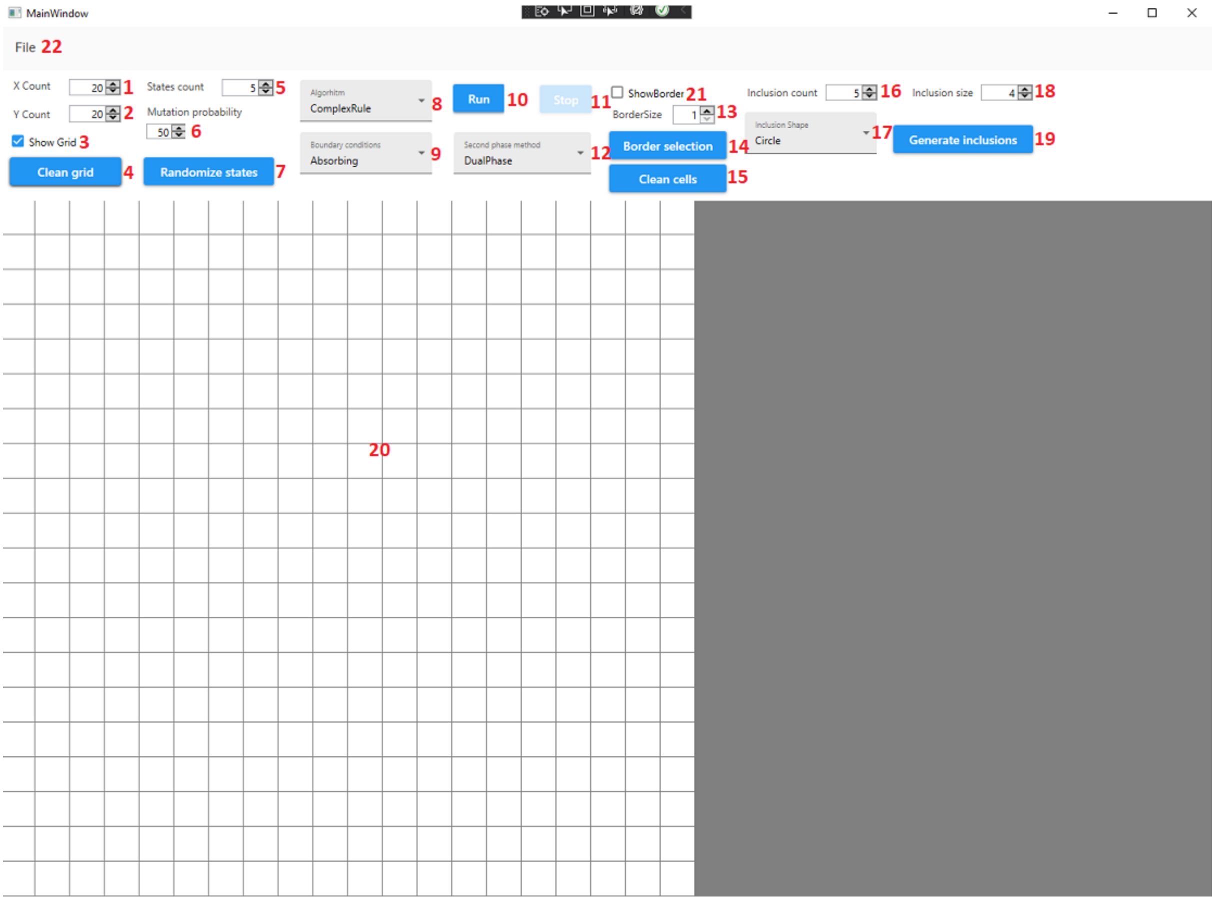
Introduction

The aim of the design part of the Multiscale Modelling course is to present a self-created computer program in which the student will use the knowledge gained during the lectures and supplementary books. The computer program should enable basic Grain Growth simulations with various user selectable parameters.

This application was written in C #, using the Windows Presentation Foundation (WPF) engine. The code is in the public repository at the link: GIHUB

1. An Application

1.1. *User Interface clarification*



X Count 20

Y Count 20

☑ Show Grid

States count 5

Mutation probability 50

Algorhitm: ComplexRule

Boundary conditions: Absorbing

Second phase method: DualPhase

□ ShowBorder

BorderSize 1

Inclusion count 5

Inclusion size 4

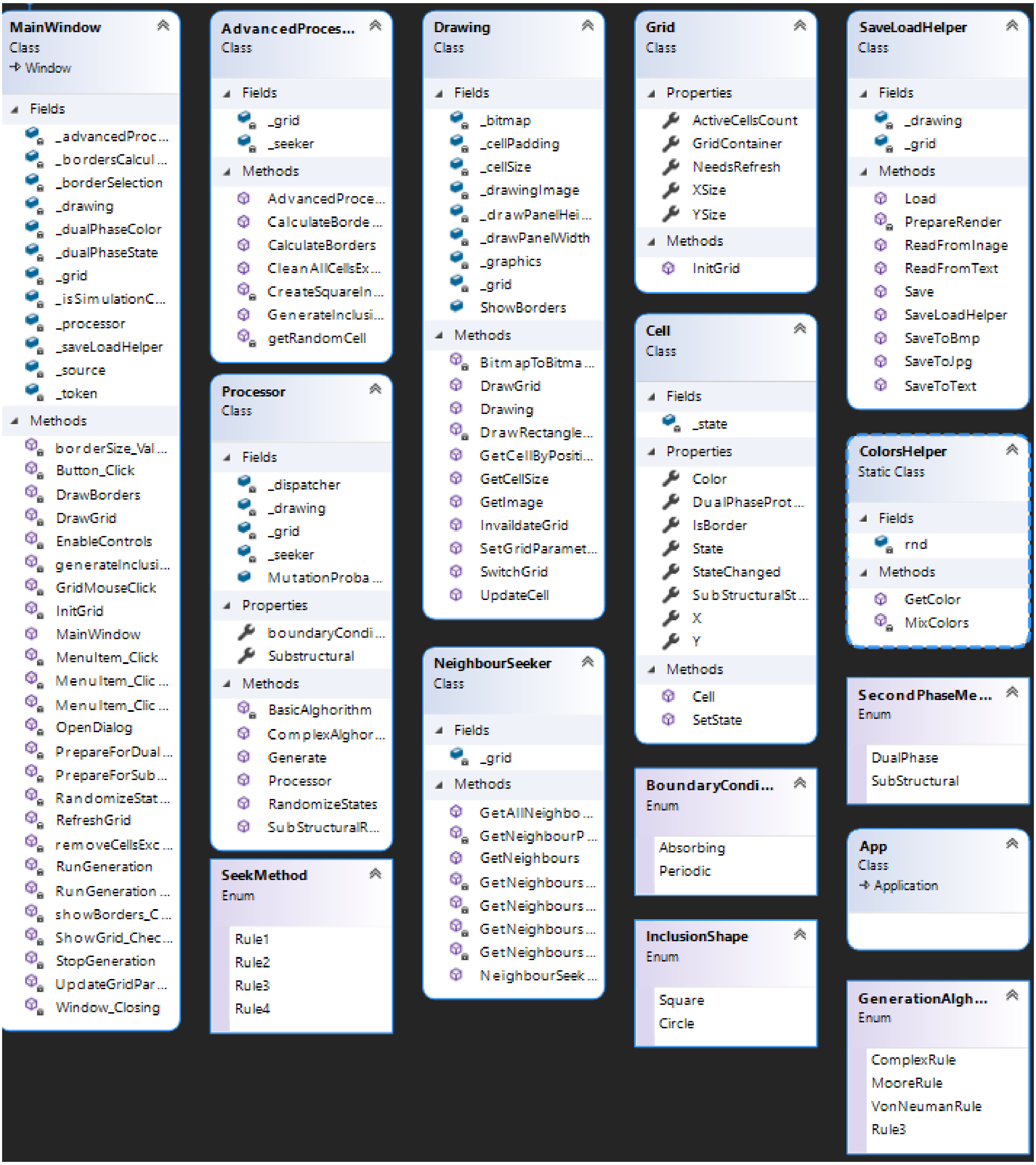
Inclusion Shape: Circle

Fig.1. User Interface

1. X count input (integer, from 3, to 2000)
2. Y count input (integer, from 3, to 2000)
3. Show grid checkbox (on/off, not recommended for dense grids)
4. Clean grid button (restores the mesh to its original state)
5. States count input (integer, from 1, no upper limit)
6. Mutation probability (integer, from 0 to 100) – for Complex CA
7. Randomize states button (generates random grains in random places)
8. Algorithm dropdown (Complex, Moore, Von Neumann, Rule3)
9. Boundary conditions dropdown (absorbing/periodic)
10. Run button (disabled during simulation)
11. Stop button (enabled during simulation)
12. Show Border checkbox (can be checked in any moment)

13. Border Size input (integer, from 1 to 10)
14. Border Selection (after selecting user is able to select borders of only one grain)
15. Cleans cells (makes grains invisible and borders visible, automatically turns on ShowBorder checkbox)
16. Inclusions amount (integer, from 1, no upper limit)
17. Inclusions shape dropdown (circle/square)
18. Inclusion size input (integer, from 1, no upper limit)
19. Generate inclusions button (If the simulation hasn’t been started, inclusions are generated in the whole grid. If the simulation has been finished, inclusions are generated only on the grains borders.)
20. The grid
21. Show Border checkbox (on/off)
22. Save/load menu

1.2. *Class diagram*



MainWindow
Class
→ Window
Fields
_advancedProc...
_bordersCalcul...
_borderSelection
_drawing
_dualPhaseColor
_dualPhaseState
_grid
_isSimulationC...
_processor
_saveLoadHelper
_source
_token
Methods
borderSize_Val...
Button_Click
DrawBorders
DrawGrid
EnableControls
generateInclusi...
GridMouseClick
InitGrid
MainWindow
MenuItem_Click
MenuItem_Clic...
MenuItem_Clic...
OpenDialog
PrepareForDual...
PrepareForSub...
RandomizeStat...
RefreshGrid
removeCellsExc...
RunGeneration
RunGeneration...
showBorders_C...
ShowGrid_Chec...
StopGeneration
UpdateGridPar...
Window_Closing

AdvancedProces...
Class
Fields
_grid
_seeker
Methods
AdvancedProce...
CalculateBorde...
CalculateBorders
CleanAllCellsEx...
CreateSquareIn...
GenerateInclusi...
getRandomCell

Processor
Class
Fields
_dispatcher
_drawing
_grid
_seeker
MutationProba...
Properties
boundaryCondi...
Substructural
Methods
BasicAlghorithm
ComplexAlghor...
Generate
Processor
RandomizeStates
SubStructuralR...

SeekMethod
Enum
Rule1
Rule2
Rule3
Rule4

Drawing
Class
Fields
_bitmap
_cellPadding
_cellSize
_drawingImage
_drawPanelHei...
_drawPanelWidth
_graphics
_grid
ShowBorders
Methods
BitmapToBitma...
DrawGrid
Drawing
DrawRectangle...
GetCellByPositi...
GetCellSize
GetImage
InvaildateGrid
SetGridParamet...
SwitchGrid
UpdateCell

NeighbourSeeker
Class
Fields
_grid
Methods
GetAllNeighbo...
GetNeighbourP...
GetNeighbours
GetNeighbours...
GetNeighbours...
GetNeighbours...
GetNeighbours...
NeighbourSeek...

Grid
Class
Properties
ActiveCellsCount
GridContainer
NeedsRefresh
XSize
YSize
Methods
InitGrid

Cell
Class
Fields
_state
Properties
Color
DualPhaseProt...
IsBorder
State
StateChanged
SubStructuralSt...
X
Y
Methods
Cell
SetState

BoundaryCondi...
Enum
Absorbing
Periodic

InclusionShape
Enum
Square
Circle

SaveLoadHelper
Class
Fields
_drawing
_grid
Methods
Load
PrepareRender
ReadFromImage
ReadFromText
Save
SaveLoadHelper
SaveToBmp
SaveToJpg
SaveToText

ColorsHelper
Static Class
Fields
rnd
Methods
GetColor
MixColors

SecondPhaseMe...
Enum
DualPhase
SubStructural

App
Class
→ Application

GenerationAlgh...
Enum
ComplexRule
MooreRule
VonNeumanRule
Rule3

Fig.2. Class diagram

1.3. Associations between GUI (MainWindow) and rest of classes

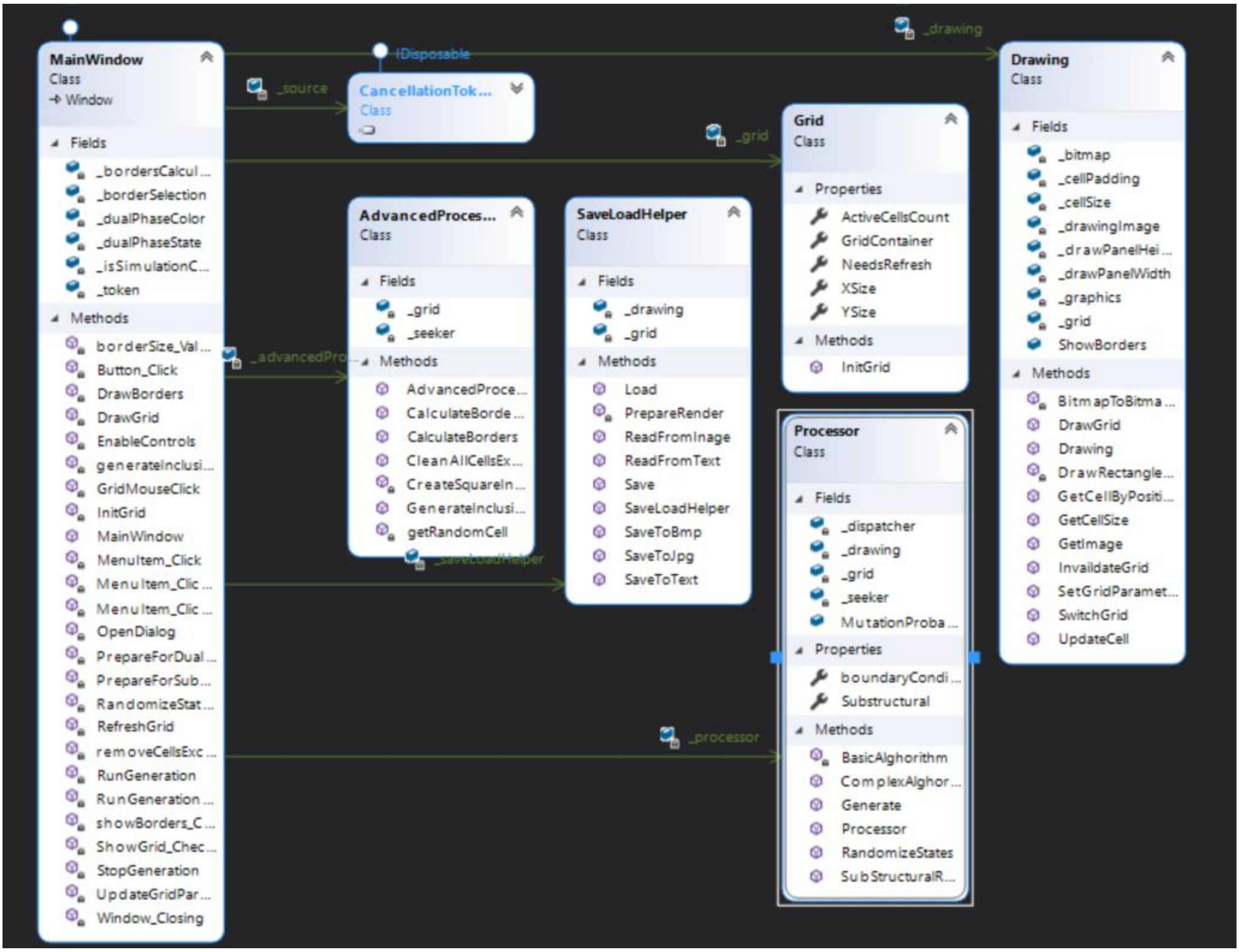
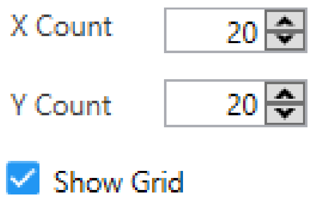


Fig.3. Associations between GUI and other classes diagram

2. Features

2.1.Mesh Size input



X Count 20

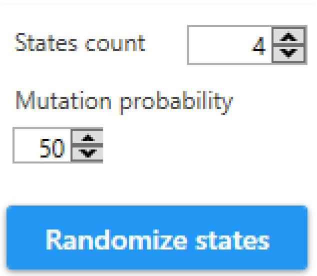
Y Count 20

☑ Show Grid

Fig.4. X, Y inputs, Show Grid checkbox

The user can enter any dimensions of the grid. X and Y don't have to be equal - the mesh doesn't have to be a square. When changing the X and Y values, the mesh automatically adjusts to the size of the window, so that it is as large as possible without losing its aspect ratio. The user also has the option of selecting "visibility" of the cell edges.

2.2. Generating random grains



States count 4

Mutation probability 50

Fig.5. States count input, Randomize States button

The basic functionality of the application is adding a user-specified number of grains in random cells. To generate grains, enter any number (integer) in the "States count" field and click the "Randomize states" button. The program will generate and colour the selected number of grains. Before generating new grains, the mesh is automatically cleaned.

2.3. Clean grid

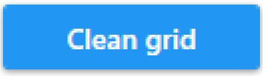


Fig.6. Clean grid button

Clean grid button provides cleaning all cells in the grid. State is being changed for 0 from all types of cells (grains, inclusions, none).

2.4. Boundary Conditions

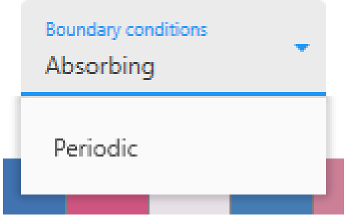
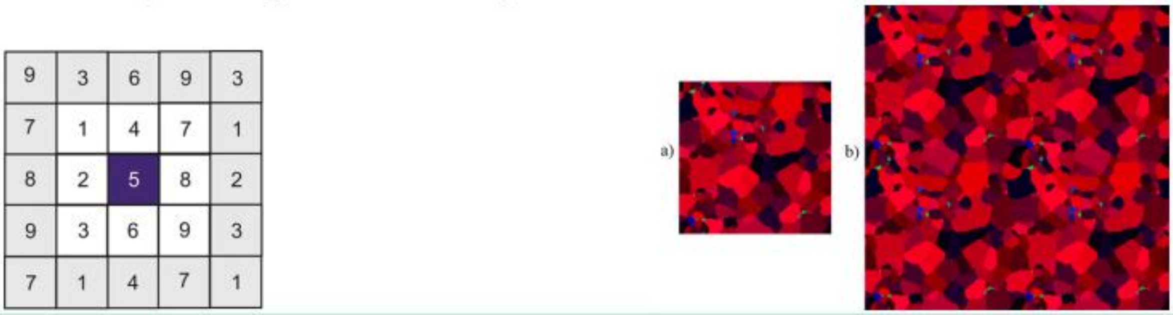

Boundary conditions
Absorbing
Periodic

Fig.7. Boundary Conditions button

Application allows to choose type of algorithm:

- periodic boundary conditions
- absorbing boundary conditions

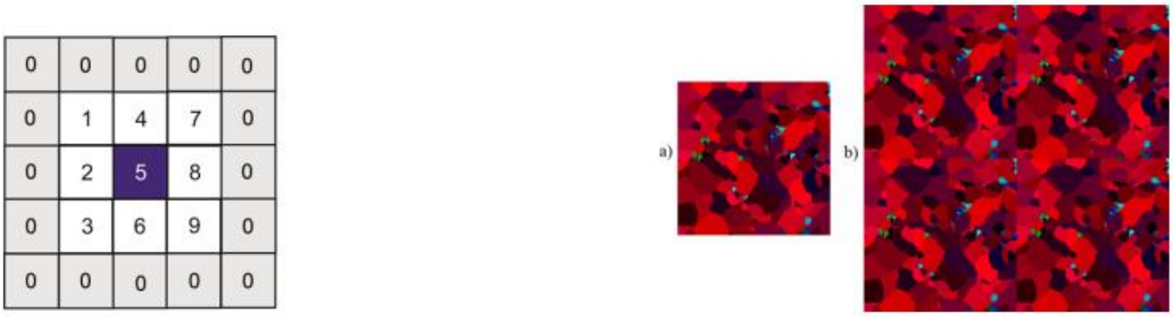
Periodic Boundary Conditions – "the CA neighbourhood is properly defined and take into account cells located on subsequent edges of the CA space"[1].



9	3	6	9	3
7	1	4	7	1
8	2	5	8	2
9	3	6	9	3
7	1	4	7	1

Fig.8. Periodic Boundary Conditions [1]

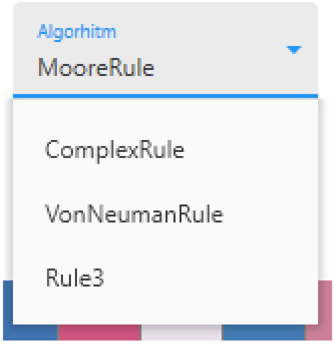
Absorbing Boundary Conditions – "the state of cells located on the edges of the CA space are properly fixed with a specific state to absorb moving quantities."[1]



0	0	0	0	0
0	1	4	7	0
0	2	5	8	0
0	3	6	9	0
0	0	0	0	0

Fig.9. Absorbing Boundary Conditions [1]

2.5.Grain growth algorithms



Algorhitm
MooreRule

Fig.10.Grain growth algorithms types dropdown

The user can choose from several types of grain growth algorithms:

- Moore (Simple CA)
- Von Neumann (Simple CA)
- Rule 3 – Further Moore (Simple CA)
- Complex Rule (Grain boundary shape control)

2.5.1. Moore (Simple CA)

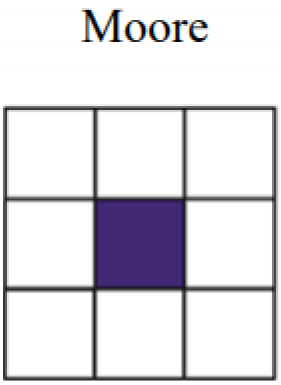


Fig.11. Moore neighborhood [1]

The Moore neighbourhood is the most basic type of algorithm. Cell "x" in the next generation will change its state (colour), if at least one of the neighbouring cells [reference to the drawing] was in the "seed" state. If there are several different grains among the neighbours, the cell changes to the type with the most neighbours. If the number of neighbours of two colours is equal, the cell changes its state to one of them (randomly).

2.5.2. Nearest Moore (Simple CA)

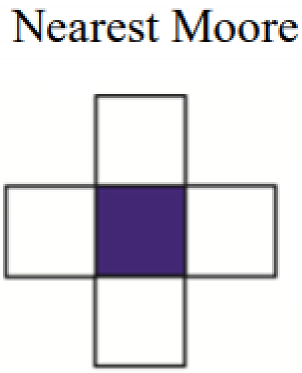


Fig.12. Nearest Moore neighborhood [1]

The Nearest Moore algorithm works the same as the Moore [2.5.1] above, but the neighbours only have 4 "side" cells.

2.5.3. Further Moore (Simple CA)

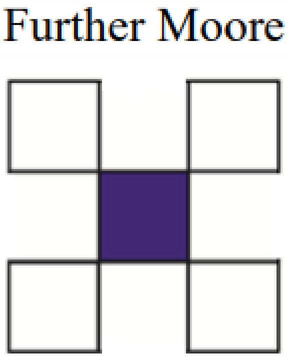


Fig.13. Further Moore neighborhood [1]

The Nearest Moore algorithm works the same as the Moore [2.5.1] and Nearest Moore, but the neighbours only have 4 "corner" cells.

2.5.4. Complex rule (Grain boundary shape control)

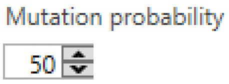
This rule combines the above three algorithms, but in a special way:

1. Neighbourhood used: Moore. A cell changes its state to a seed if at least 5 of its neighbours are in a seed state. If the number of neighbours in the seed state is smaller, the cell does not change its state.

2. Neighbourhood used: Moore Nearest. A cell changes its state to a seed if at least 3 of its neighbours are in the seed state. If the number of neighbours in the seed state is smaller, the cell does not change its state.

3. Neighbourhood used: Further Moore. A cell changes its state to a seed if at least 3 of its neighbours are in the seed state. If the number of neighbours in the seed state is smaller, the cell does not change its state.

4. Neighbourhood used: Moore. The id of particular cell depends on its all neighbours, and has X % probability chance to change. Probability can be changed by the input below.



Mutation probability

50

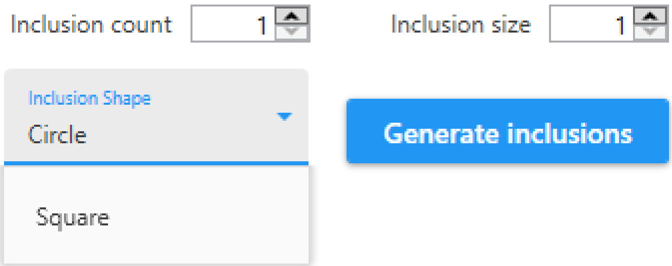
Fig.14. Mutation probability input

It is a percentage value, therefore the user can enter integer values from 1 to 100.

2.6.Inclusions

Inclusions are cells that are "closed". They cannot change state to a grain, nor do they affect the cells surrounding them.

We distinguish inclusions in the shape of:
- circle,
- a square.



Inclusion count 1

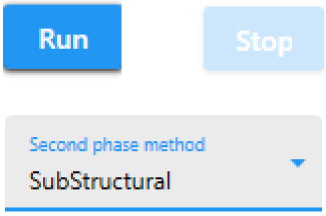
Inclusion size 1

Inclusion Shape Circle

Fig.15. Inclusions type dropdown, amount and size inputs

The user specifies the radius for the circle and the side length for the square. Inclusions may be added before or after simulation of grain expansion. In the first case, inclusions are added at random places throughout the mesh. In the second case (when simulation has been finished), inclusions are added only at the boundaries of different grains.

2.7.Substructure

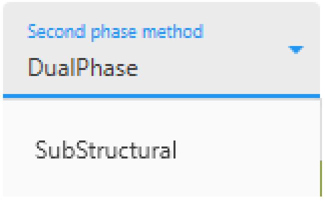


Second phase method
SubStructural

Fig.16. Substructure dropdown

The substructure allows the grains to grow again after the first phase has been completed. After the simulation is over, the user can select "Substructure" in the dropdown and click the "Start" button. Then X grains (states count) will be generated in each area (grains of a given type). Most importantly, the re-generation of grains does not go beyond the original area. After the "Substructure" simulation, there should still be more or less visible boundaries of the "old" areas (depending on the parameters).

2.8.Dual Phase

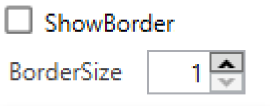


Second phase method
DualPhase

Fig.17. Dual Phase dropdown

Dual phase allows you to generate the second phase of the beans. After completing the first stage of growth, the user should select the "Dual Phase" value in the dropdown. Then you can select any number of areas, enter the appropriate number of States Count and start the simulation. The selected areas will not be changed. The selected number of new grains will be generated in the remaining places.

2.9. Visible areas borders



□ ShowBorder

BorderSize 1

Fig.18. Show Borders checkbox, Border Size input

After selecting the "ShowBorders" checkbox, the area boundaries are shown at the end of the simulation. The checkbox can be checked or unchecked at any time. Input allows user to choose the thickness of the border.

2.10. Clean grains

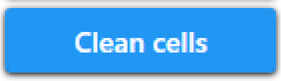


Fig.19. Substructure dropdown

After clicking the Clean Cells all grains are changed to “invisible”. At the same time the “ShowBorders” checkbox is checked.

2.11. One area selection

Fig.20. Substructure dropdown

After simulation user can use the “Border selection” feature. After clicking the “Border selection” button, user can choose one of the grains. Consequently, only the boundaries of the selected area remain visible on the grid.

2.12. *Save and load mesh*

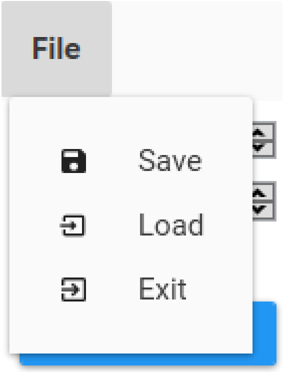


Fig.21. Save/load menu

When the grains are generated, user can save it as a txt/bmp/jpeg file. Application provides also loading generated grains from the same file formats as saving. Grains loaded from the text file are fully supported by application mesh, so user is still able to take actions such as adding the borders.

2.13. *Mouse Actions*

Due to the need of testing individual aspects of the application (eg. algorithms), it is possible to manually add grains and inclusions. Adding grains - clicking the left mouse button on any white cell results in "painting" it in a new colour (it will give it a grain state), - right clicking on any cell adds a single "single-cell" inclusion.

3. *Usage Examples*

3.1. Moore Neighbourhood

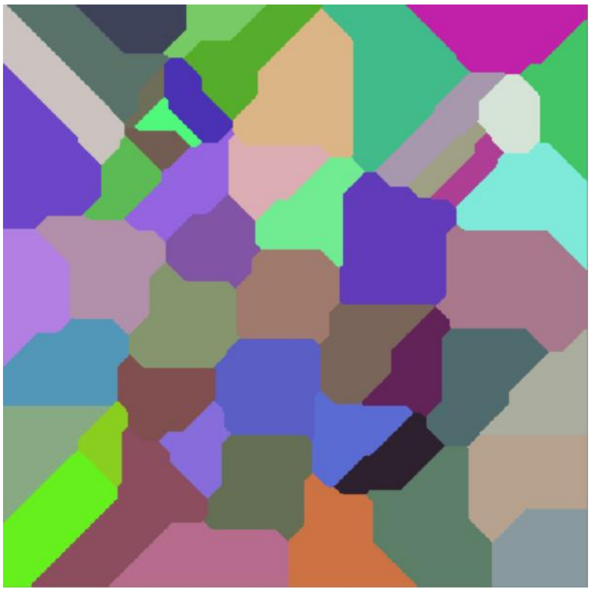


Fig.22. Simulation result

Grid size: 400 x 400
Show grid: off
States count: 50
Boundary: absorbing
Rule chosen: Moore
Mutation probability: -
Second phase: -
Borders: off
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

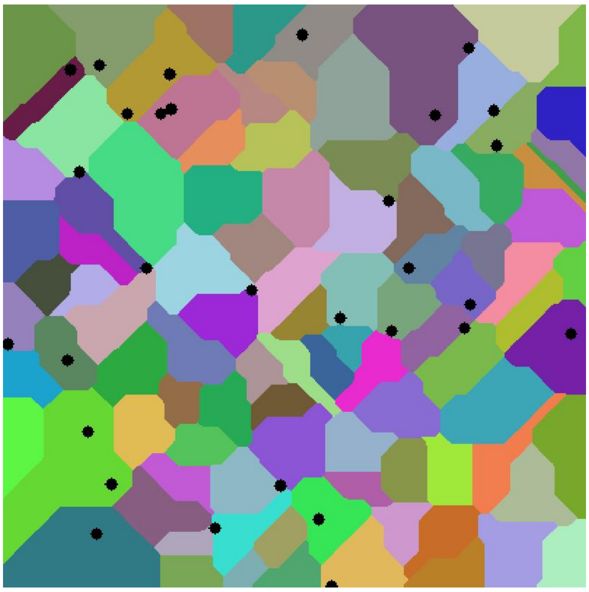


Fig.23. Simulation result

Grid size: 400 x 400
Show grid: off
States count: 100
Boundary: absorbing
Rule chosen: Moore
Mutation probability: -
Second phase: -
Borders: off
Inclusions: off
Inclusions type: Before
Inclusions shape: Circle
Inclusions size: 5
Inclusions amount: 30

3.2. *Nearest Moore*

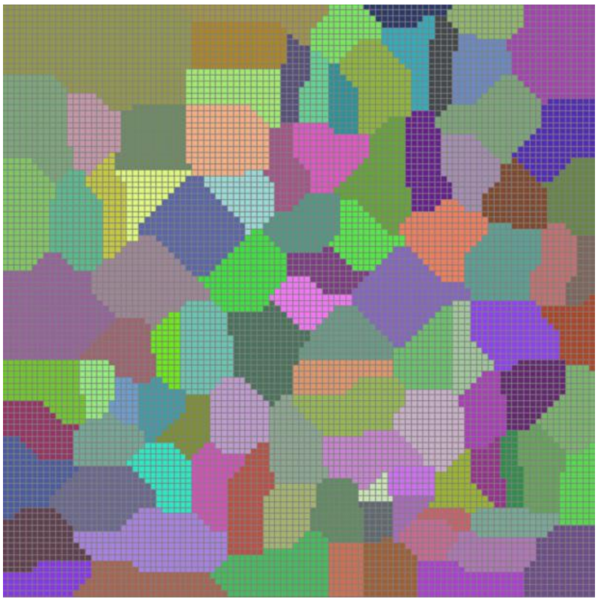


Fig.24. Simulation result

Grid size: 100 x 100
Show grid: on
States count: 100
Boundary: absorbing
Rule chosen: Nearest Moore
Mutation probability: -
Second phase: -
Borders: off
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

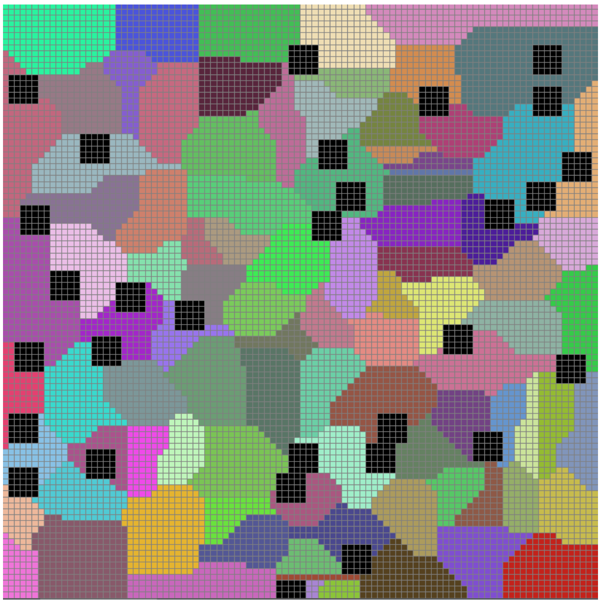


Fig.25. Simulation result

Grid size: 100 x 100
Show grid: on
States count: 20
Boundary: absorbing
Rule chosen: Nearest Moore
Mutation probability: -
Second phase: -
Borders: off
Inclusions: off
Inclusions type: Before
Inclusions shape: Square
Inclusions size: 5
Inclusions amount: 30

3.3. Grain Boundary Shape Control

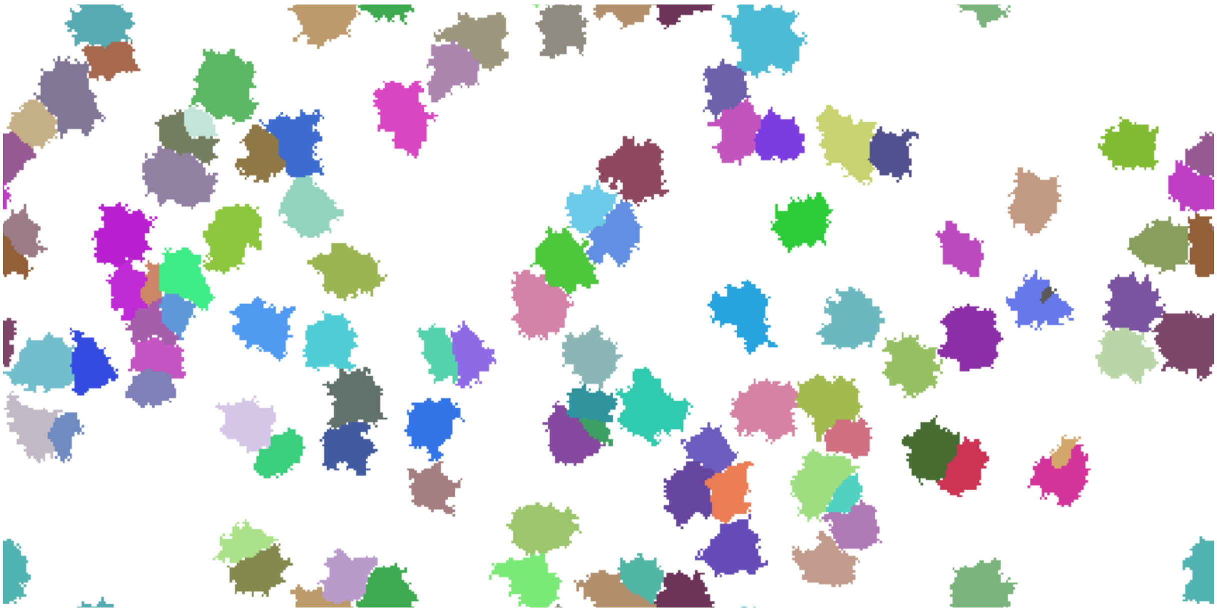


Fig.26 Simulation result after several steps

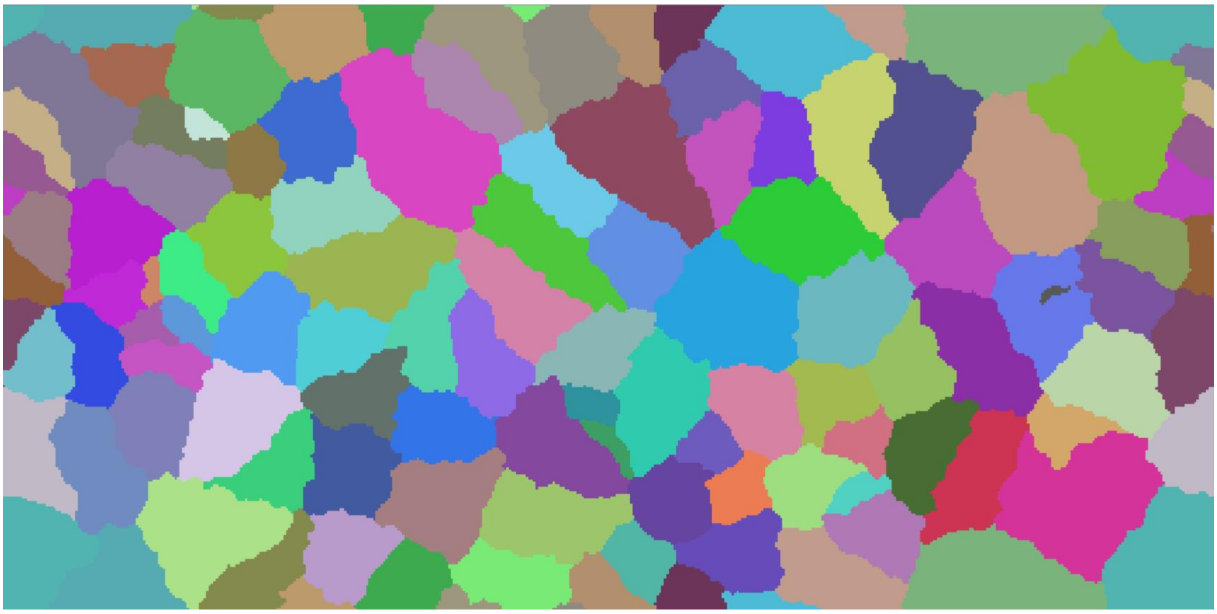


Fig.27 Simulation result

Grid size: 100 x 100
Show grid: on
States count: 100
Boundary: Periodic
Rule chosen: Grain boundary shape control
Mutation probability: 10

Second phase: -
Borders: off
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

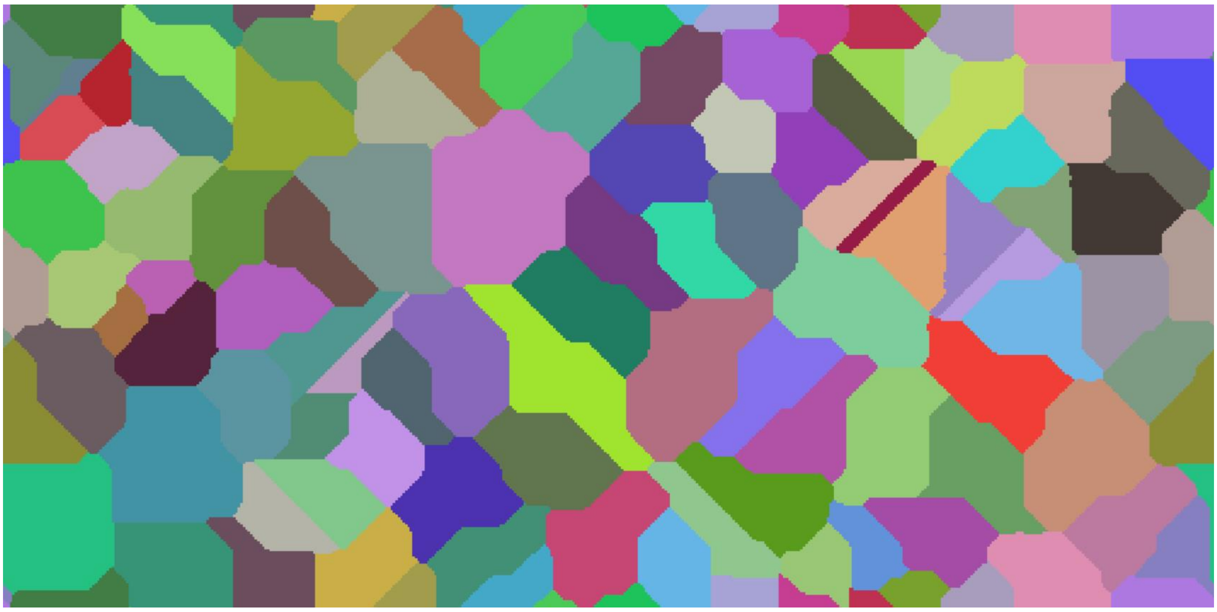


Fig. 28. Simulation result – options the same as above, despite Mutation probability: 99

3.4. *Inclusions added before simulation*

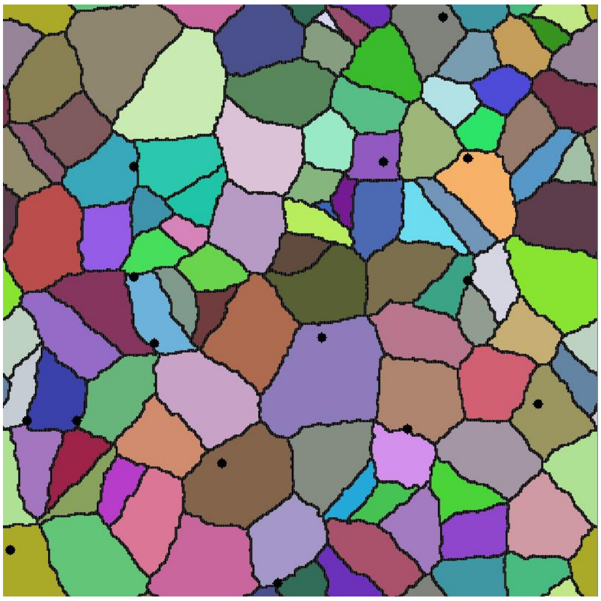


Fig.29. Simulation result

Grid size: 500 x 500
Show grid: off
States count: 100
Boundary: Periodic
Rule chosen: Grain boundary shape control

Mutation probability: 50
Second phase: -
Borders: on
Inclusions: on
Inclusions type: Before
Inclusions shape: Circle
Inclusions size: 5
Inclusions amount: 15

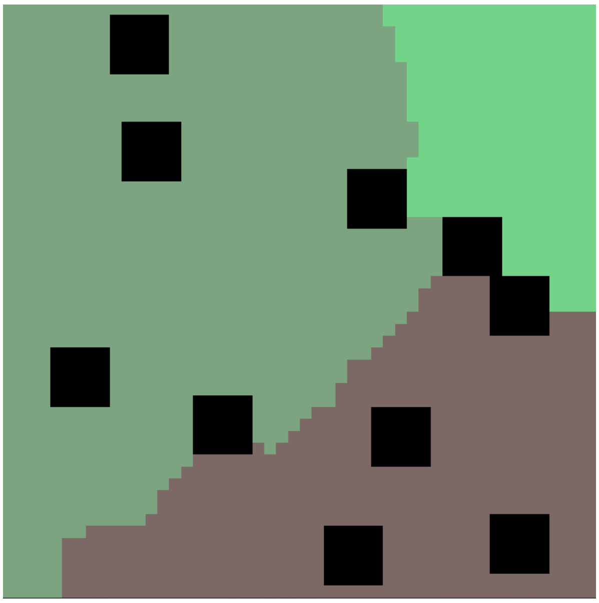


Fig.30. Simulation result

Grid size: 50 x 50
Show grid: off
States count: 3
Boundary: Absorbing
Rule chosen: Grain boundary shape control
Mutation probability: 50
Second phase: -
Borders: off
Inclusions: on
Inclusions type: Before
Inclusions shape: Square
Inclusions size: 5
Inclusions amount: 10

3.5. *Inclusions added before simulation*

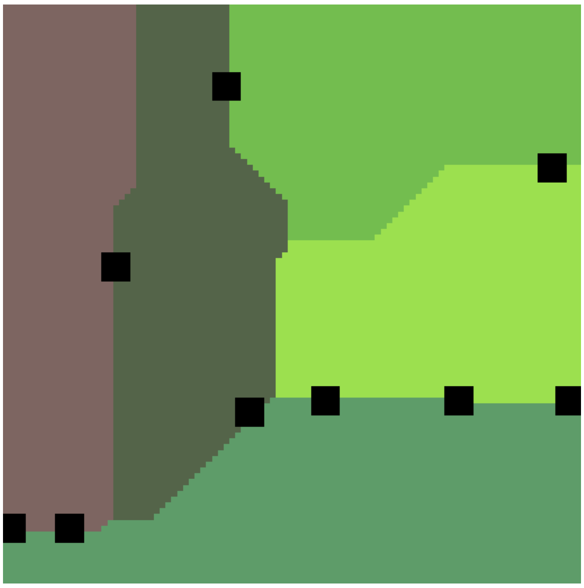


Fig.31. Simulation result

Grid size: 100 x 100
Show grid: off
States count: 5
Boundary: Absorbing
Rule chosen: Nearest Moore

Mutation probability: -
Second phase: -
Borders: off
Inclusions: on
Inclusions type: After
Inclusions shape: Square
Inclusions size: 5
Inclusions amount: 5

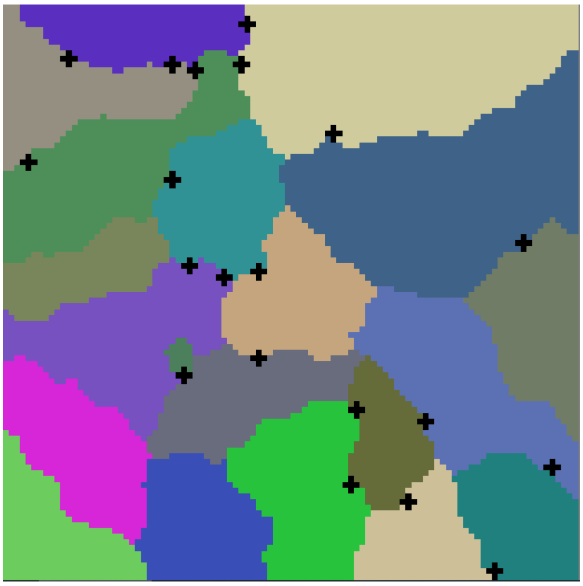


Fig.32. Simulation result

Grid size: 100 x 100
Show grid: off
States count: 20
Boundary: Absorbing
Rule chosen: Grain boundary shape control
Mutation probability: 1
Second phase: -
Borders: off
Inclusions: on
Inclusions type: After
Inclusions shape: Circle
Inclusions size: 2
Inclusions amount: 20

3.6. *Dual Phase*

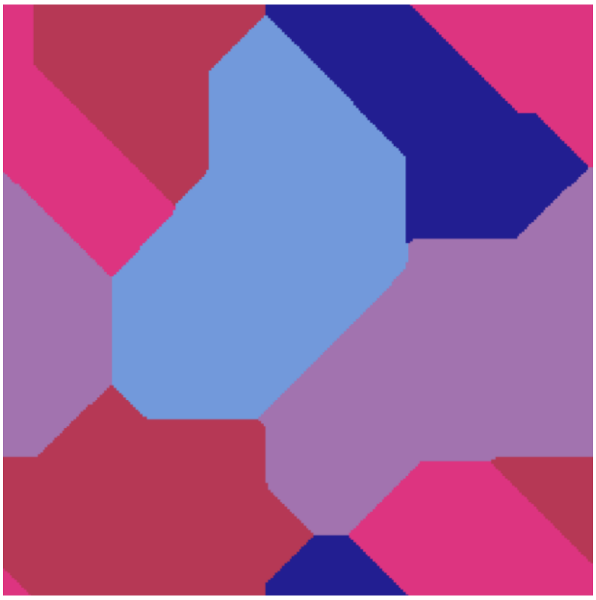


Fig.33. Simulation result

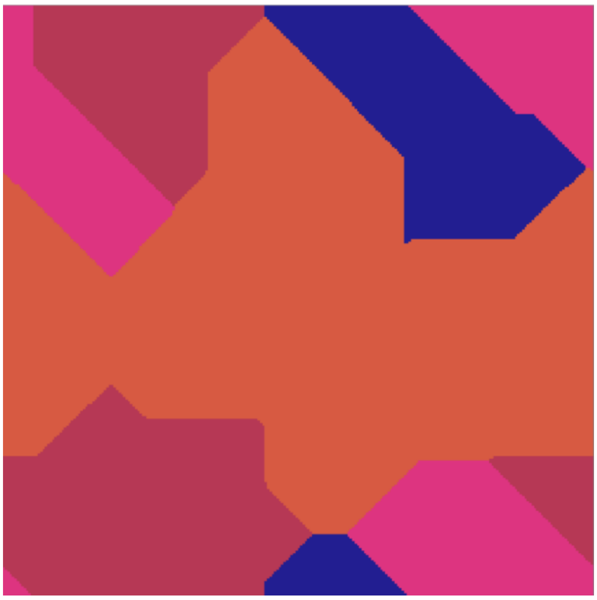


Fig.34. Simulation result with selected 3 types of grains

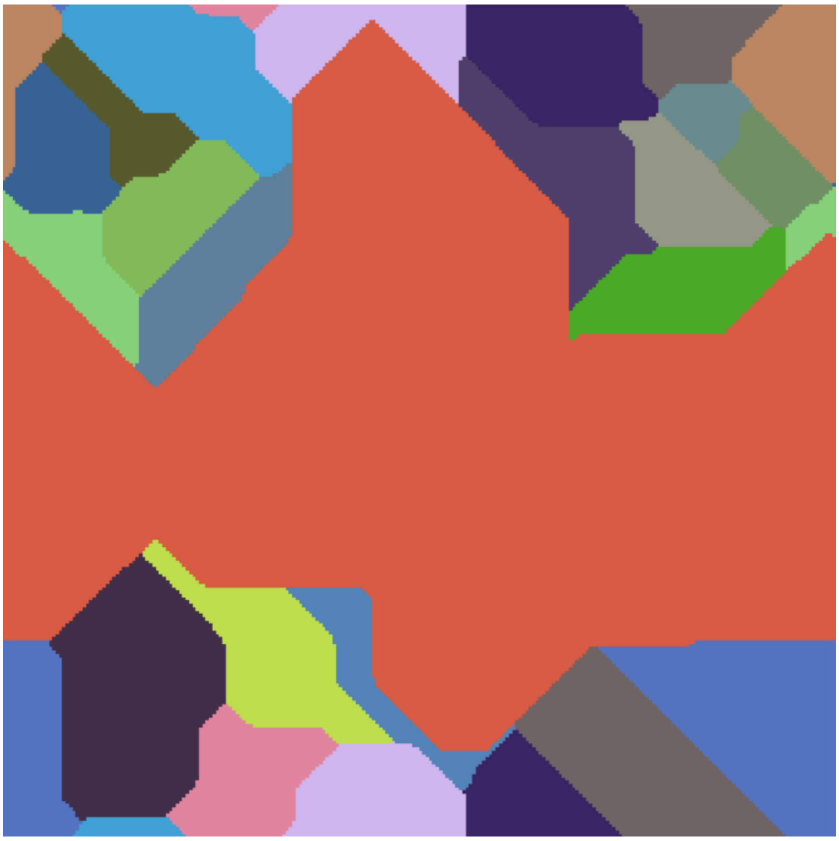


Fig.35. Second phase with 20 new grains result

Grid size: 250 x 250
Show grid: off
States count: 5
Boundary: Periodic
Rule chosen: Grain boundary shape control
Mutation probability: 10
Second phase: Dual Phase, second generation with 20 new grains
Borders: off
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

3.7. Sub-structural

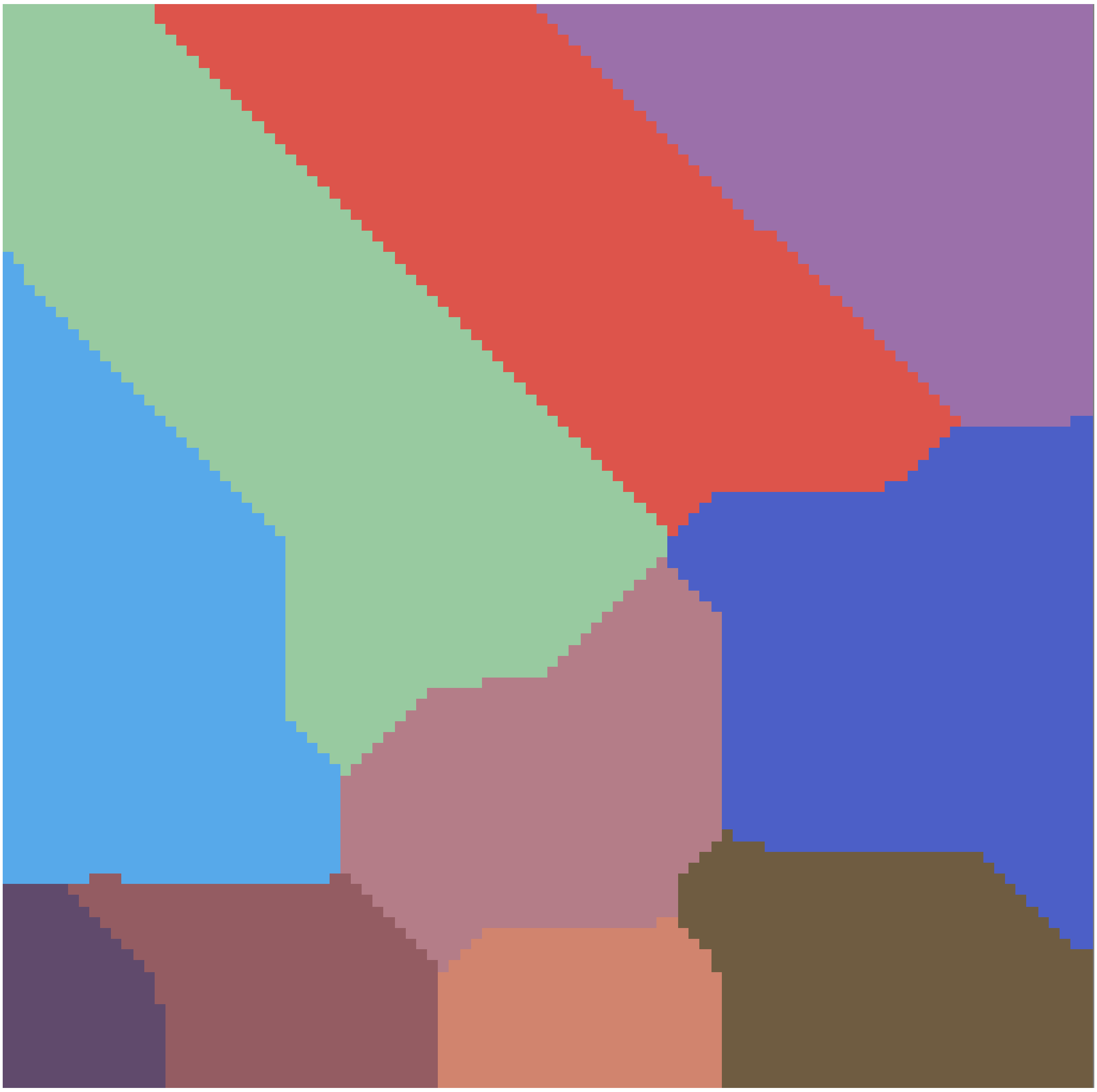


Fig.36. Simulation result

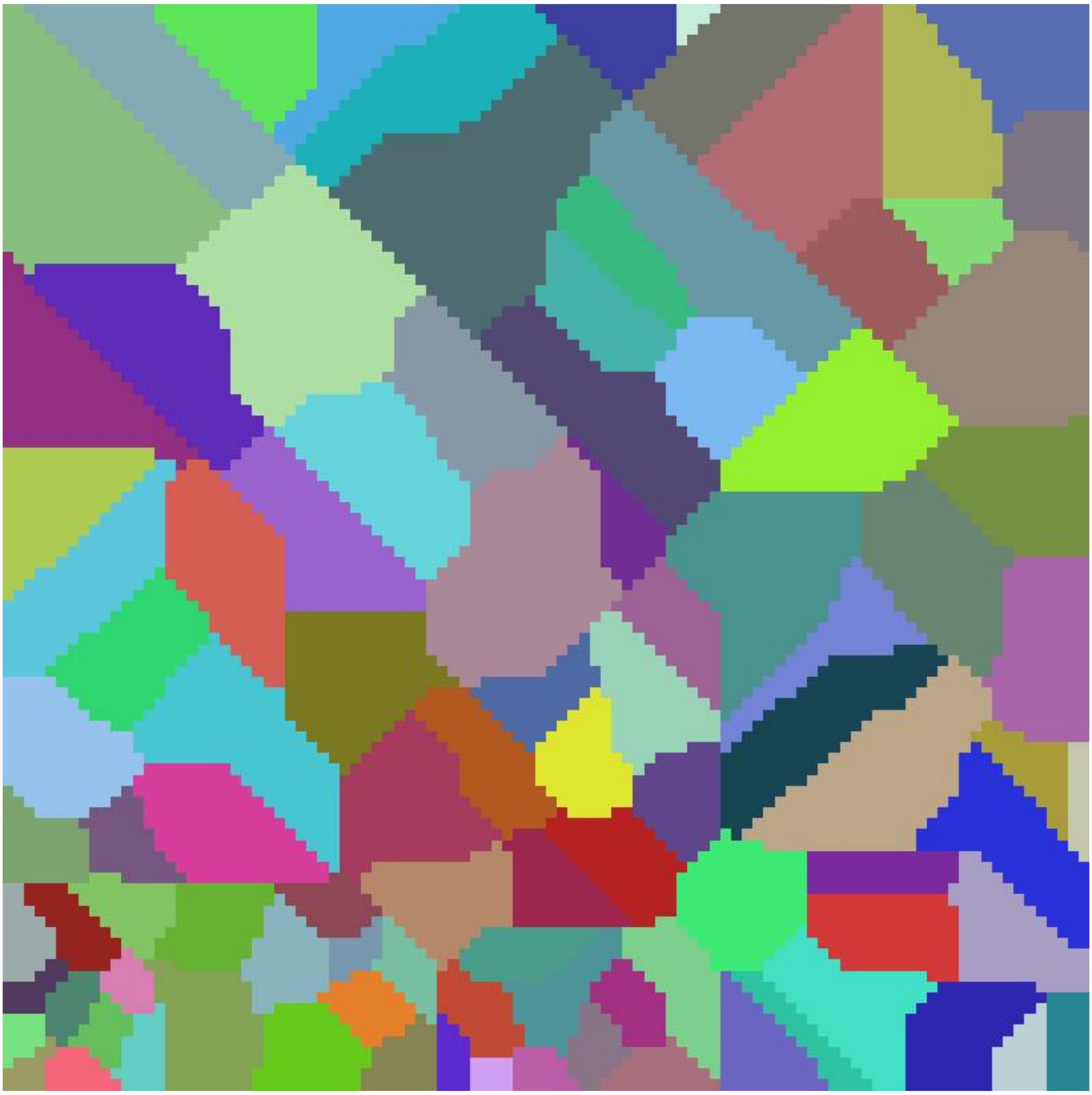


Fig.37. Second phase with 10 grains in each area simulation results

Grid size: 100 x 100
Show grid: off
States count: 100
Boundary: Absorbing
Rule chosen: Grain boundary shape control
Mutation probability: 99
Second phase: Substructural, 10 grains generated in the second phase
Borders: off
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

3.8.Display boundaries

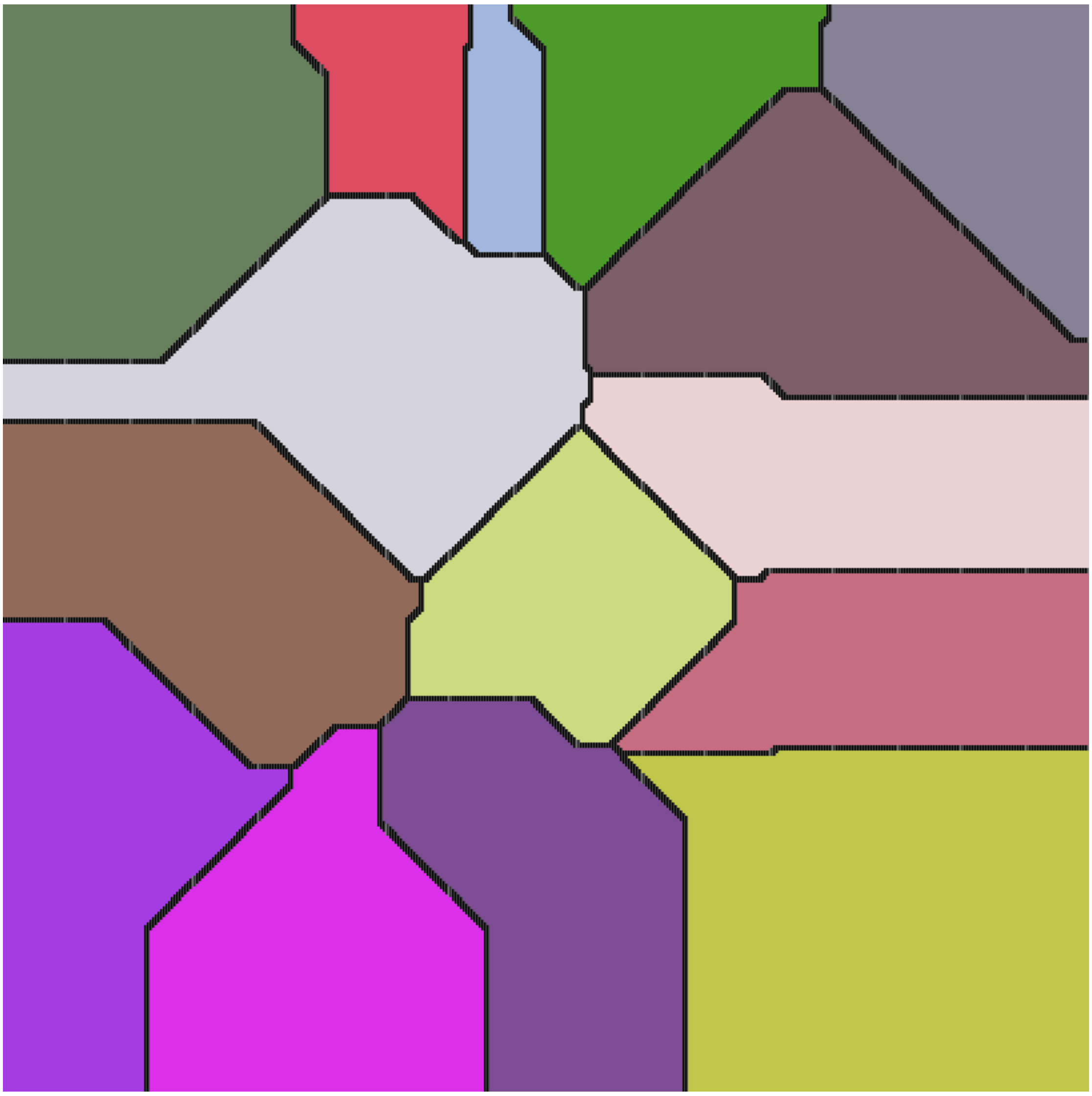


Fig.38. Simulation results with boundaries, size:1

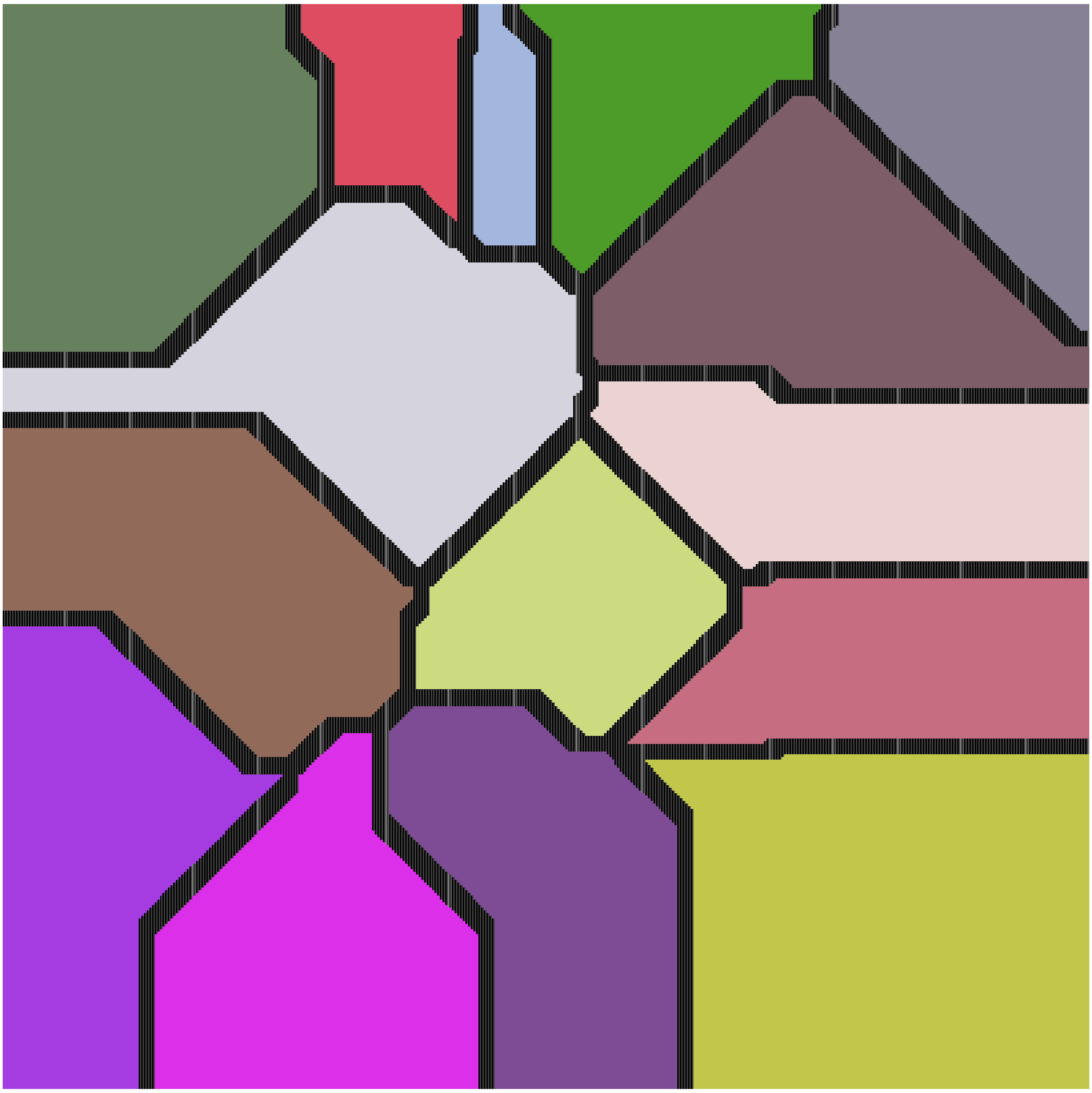


Fig.38. Simulation results with boundaries, size:3

Grid size: 400 x 400
Show grid: off
States count: 15
Boundary: Absorbing
Rule chosen: Nearest Moore
Mutation probability: -
Second phase: -
Borders: on
Borders size: 1 -> 3
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

3.9.Clear all grains, display only boundaries

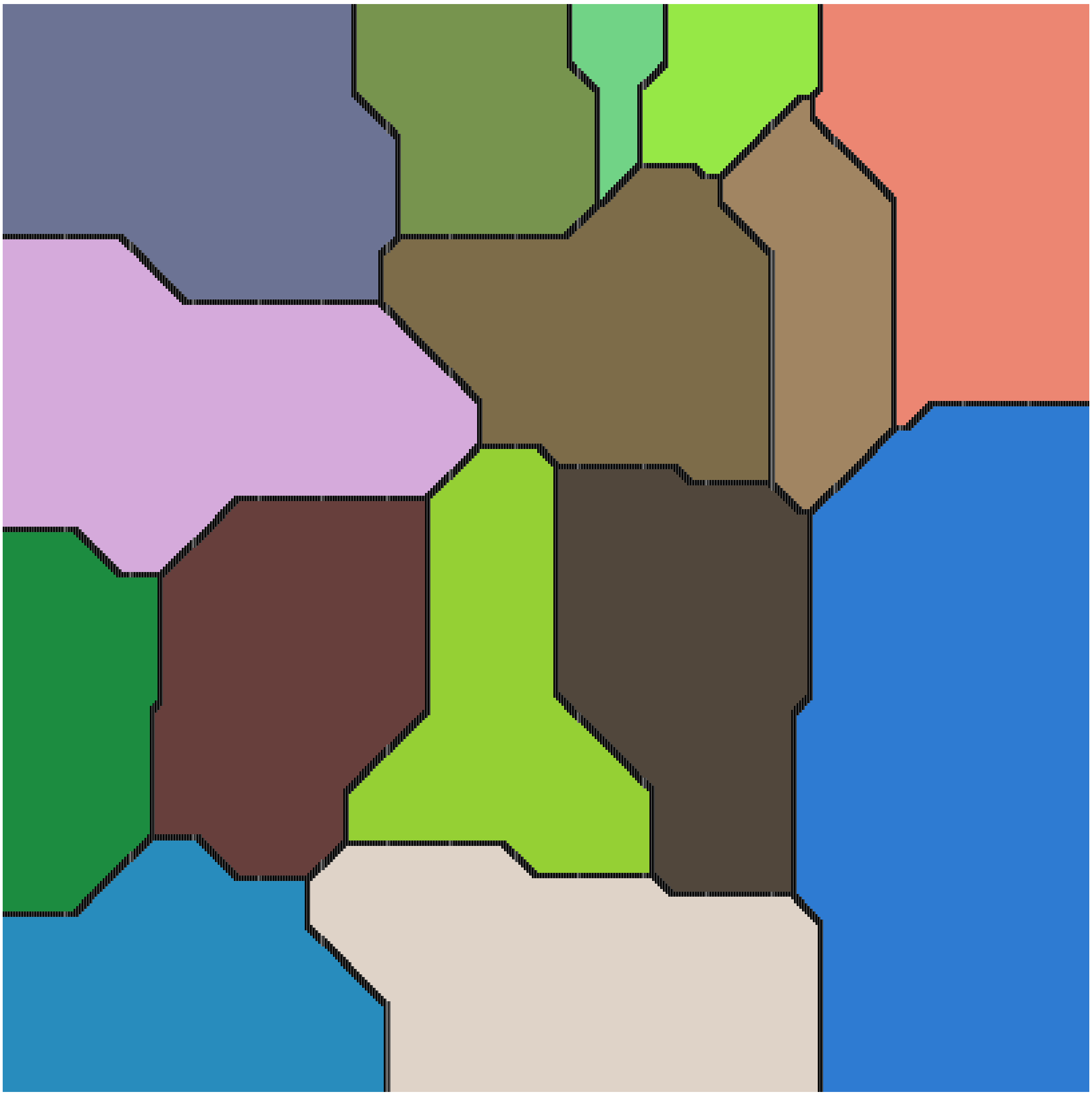


Fig.39. Simulation results with boundaries

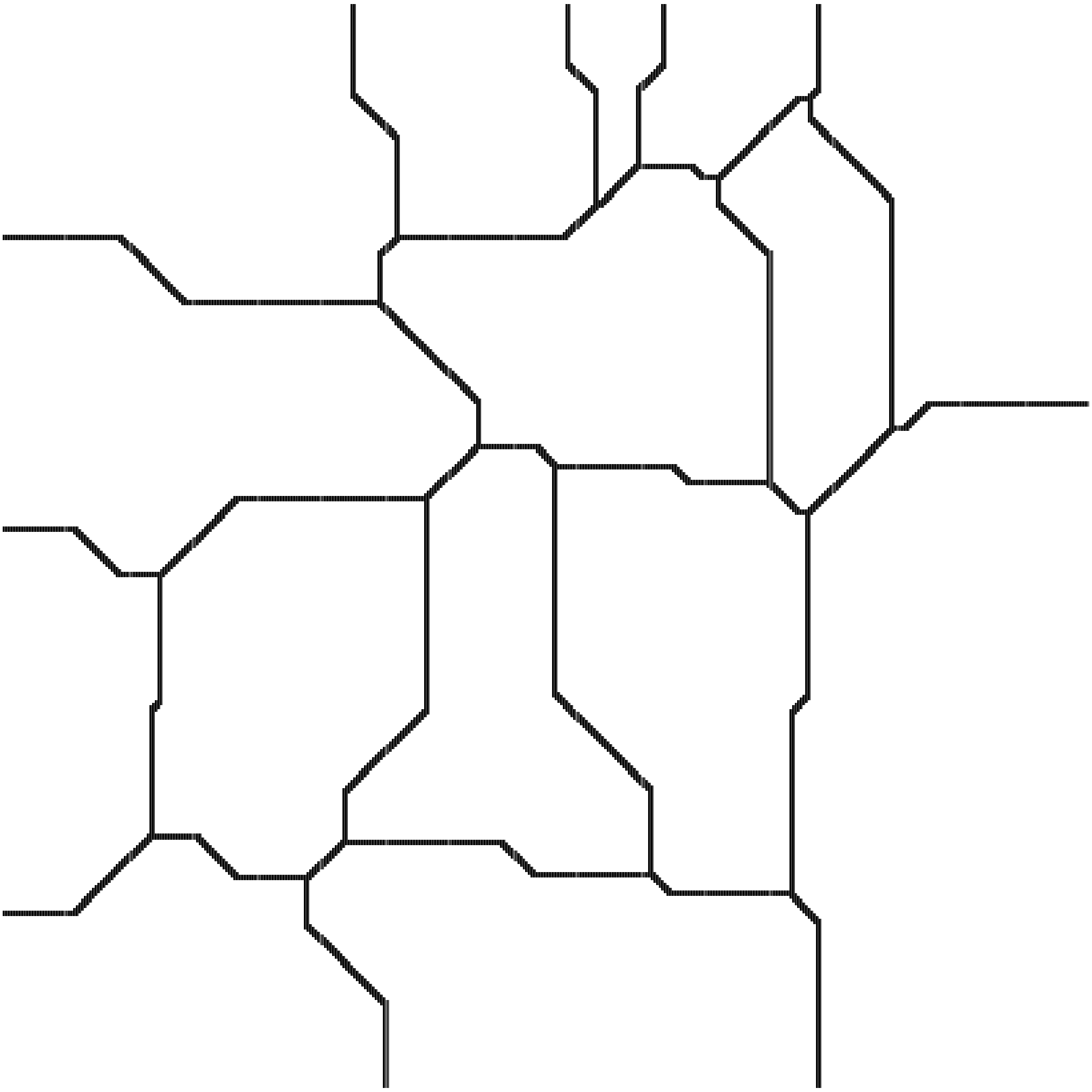


Fig.40. Simulation results, only borders are displayed

Grid size: 400 x 400
Show grid: off
States count: 15
Boundary: Absorbing
Rule chosen: Nearest Moore
Mutation probability: -
Second phase: -
Borders: on
Borders size: 1
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

3.10. Display only one area boundaries

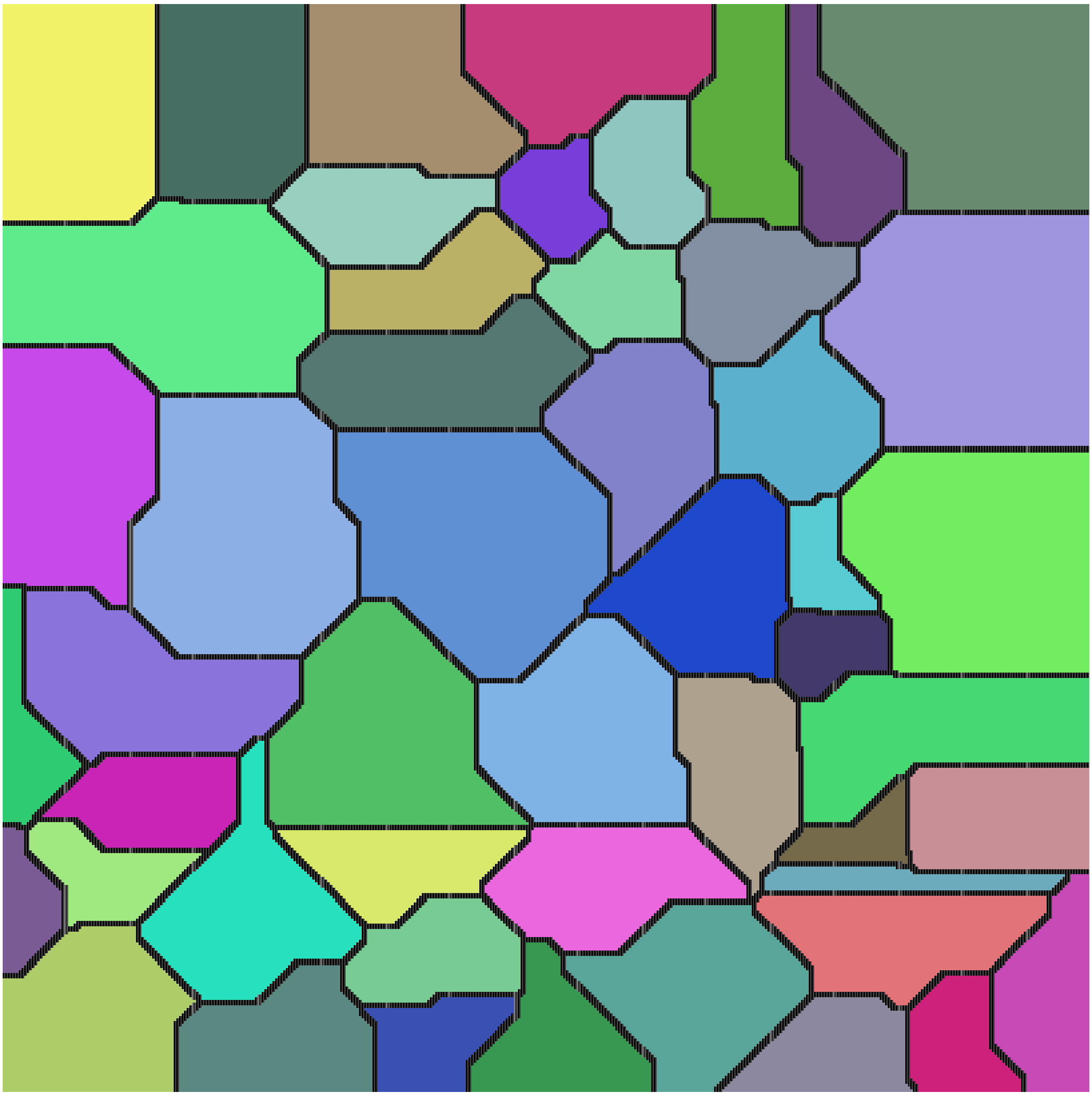


Fig.41. Simulation results, all areas are displayed

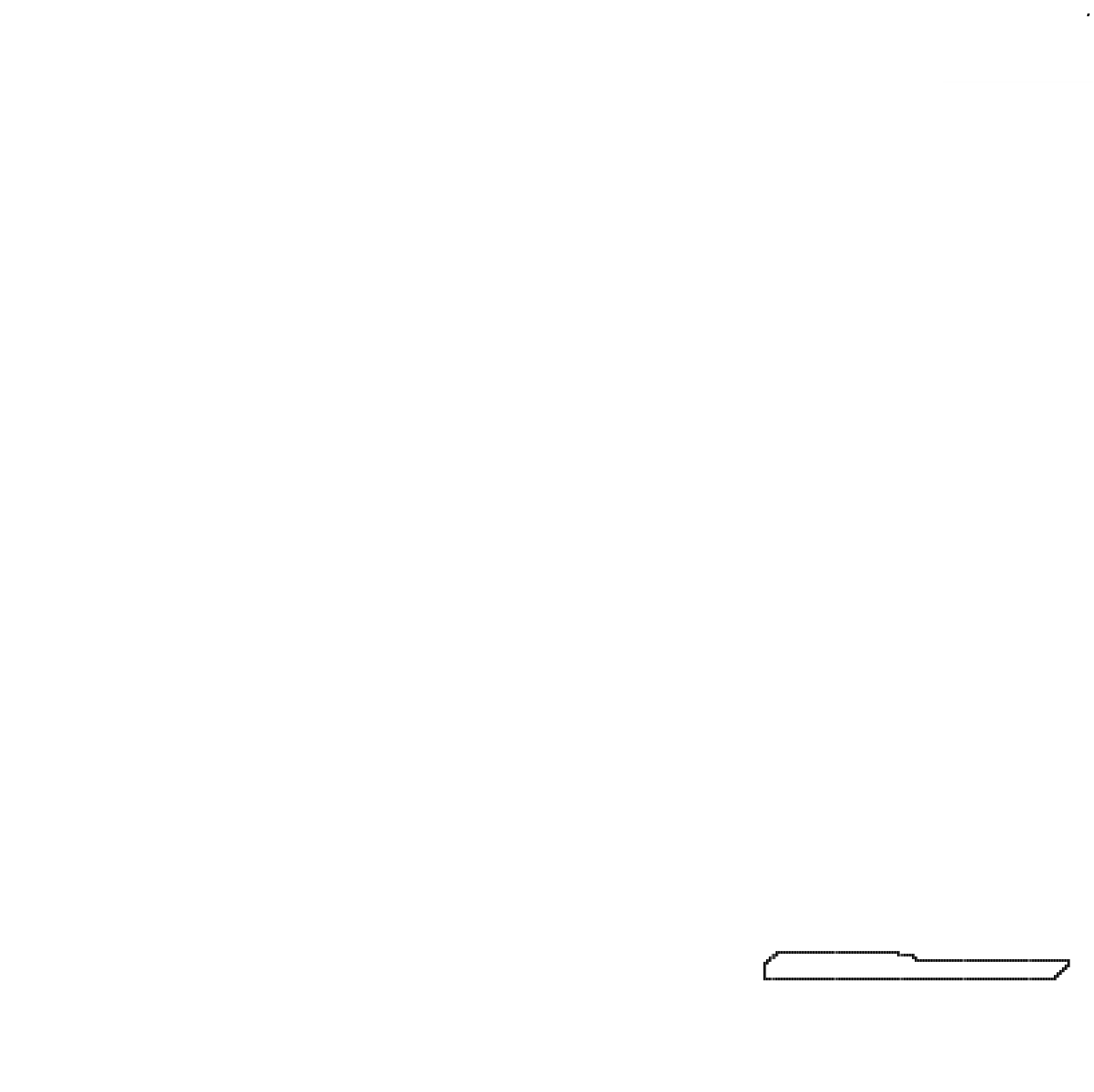


Fig.41. Simulation results, only one area border is displayed

Grid size: 400 x 400
Show grid: off
States count: 50
Boundary: Absorbing
Rule chosen: Nearest Moore
Mutation probability: -
Second phase: -
Borders: on
Borders size: 2
Inclusions: off
Inclusions type: -
Inclusions shape: -
Inclusions size: -
Inclusions amount: -

Bibliography

[1] http://home.agh.edu.pl/~lmadej/multiscale-modelling/

[2] Zienkiewicz, O.C. and R.L. Taylor, „The Finite Element Method. Basic Formulation and Linear Problems”