Grain Boundary Growth Model Implementation into CD++ Modeller

By: Heather Morris 100686492

Submitted: November 25th, 2010

**1.0 Introduction**

In metallurgy, the grain size is an important characteristic of the metal since it will determine some of its physical properties, but the grain size is not constant since the atoms in the metal are always looking for the lowest energy configuration. This can become a problem if the grain size changes can’t be estimated, since the physical properties will change. In order to reduce the energy in the crystal structure of the material the grains will reorient to the same orientation of the neighbouring cell, if the change can reduce the energy. The energy is reduced if the number of neighbours with the same orientation is increased. This process occurs until a state of equilibrium is obtained. See figure 1 for the growth characteristics of the grains in a metal.

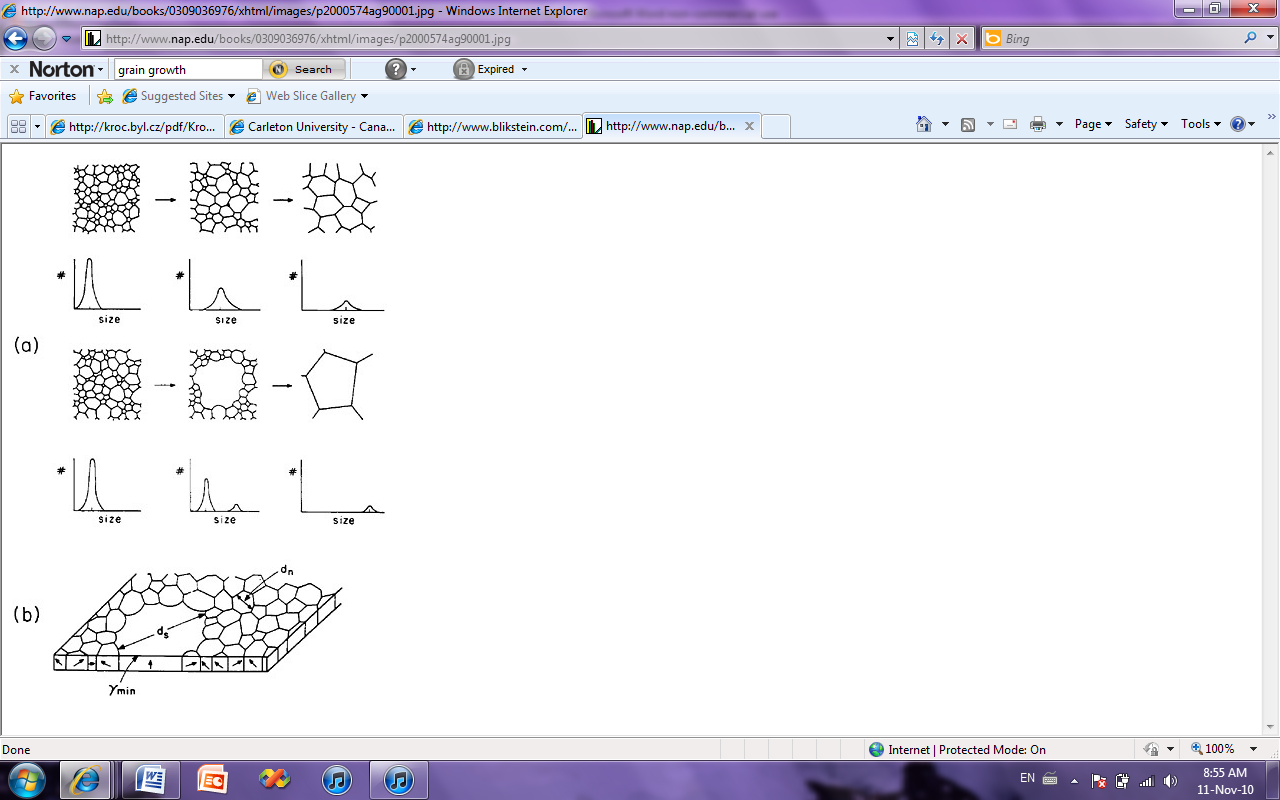


Figure 1: Grain Growth Characteristics

A cellular model has been developed to estimate the grain size changes which can be modelled using a cellular automata approach, see the attached “Monte Carlo Simulation of Grain Growth” by Paulo Blikstein, André Paulo Tschiptschin. For a time step the model will look at the cell and count the number of elements that have the same orientation and different orientation than the cell. The stored energy in the cell will be the represented as numbers of cells with a different orientation to that cell. It will then perform a comparison survey by changing orientation to another orientation that is the same as one of the neighbouring cells. It will check the difference between the energy from the first orientation and the second orientation, and which ever value is the lowest the cell will take that orientation. In this model the time it takes to change from one orientation to the other is not taken into account, thus the time step will be set at a constant value.

1.1 Behaviour of the Model

This model will use the Moore’s neighbourhood, plus one neighbour on the layer above. What the cell will do is look at its neighbours and counts all the possible states. In order to get the number of states that are different, the number of cells with the state of the current cell has to be subtracted.

There are 6 with the state 1 and 3 with the state 2, then subtracting

1

2

2

the number of cells in the state 2, there are 6 different stated cells

1

2

1

1

1

1

One issue is when counting the cells with the states, it takes into account the whole neighbourhood, so what will happen with the cell in the second layer. If we know that the cell on the second layer is a different state then the cell in question, then the total count would have been increased by 1. So, subtracting one from the total count would eliminate the contribution of the cell on the second level.

**2.0 Formal Cell-DEVS Specification**

The representation of this model as a cell-DEVS model is as follows. The model will be a 10 by 10 grid with two layers. The first layer is to show the grains moving and growing where the second layer holds the “guess” orientation to perform the energy change check to see if the grain boundary will move to the new orientation.

CD =<Xlist, Ylist, I, *X*, *Y*, η, N, {r, c}, C, B, Z>

Xlist ={0}

Ylist={0}

X={0}

Y={0}

I= <Px, Py> Px={0} , Py ={0}

S = {0,1,2,3,4,5,6,7}

η = 10

N= (-1,-1,0) (-1,0,0) (-1,1,0) (0,-1,0) (0,0,0) (0,1,0) (1,-1,0) (1,0,0) (1,1,0) (0,0,1)

r=10

c=10

h=2

B={0} wrapped

C= PijzY1 → P i-1 j-1 zX1 P i+1 j+1 zX1→ PijzY1

PijzY2 → P i j-1 z X2 P i j+1 z X2 → PijzY2

PijzY3 → P i+1 j-1 zX3 P i-1 j+1 zX3→ PijzY3

PijzY4 → P i-1 j zX4 P i+1 j zX4 → PijzY4

PijzY5 → P i j zX5 P i j zX5 → PijzY5

PijzY6 → P i+1 j zX6 P i-1 j zX6 → PijzY6

PijzY7 → P i-1 j+1 zX7 P i+1 j-1 zX7→ PijzY7

PijzY8 → P i j+1 zX8 P i j-1 zX8 → PijzY8

PijzY9 → P i j z+1X9 P i j z-1X9→ PijzY9

The cell model was integrated into the CD++ modeller tool and the model is represented by the following.

type : cell

dim : (10,10,2)

delay : transport

defaultDelayTime : 100

border : wrapped

neighbors : GrainGrowth(-1,-1,0) GrainGrowth(-1,0,0) GrainGrowth(-1,1,0)

neighbors : GrainGrowth(0,-1,0) GrainGrowth(0,0,0) GrainGrowth(0,1,0)

neighbors : GrainGrowth(1,-1,0) GrainGrowth(1,0,0) GrainGrowth(1,1,0)

neighbors : GrainGrowth(0,0,1)

initialvalue : 0

initialCellsValue : InitialGrain.val

localtransition : Growth-rule

zone : top-rule { (0,0,1)..(9,9,1) }

[Growth-rule]

rule : {(0,0,0)} 100 { cellpos(2)=0 and (0,0,0) != (0,0,1) and statecount(0) + statecount(1) + statecount(2) + statecount(3) + statecount(4) + statecount(5) + statecount(6) + statecount(7) - statecount((0,0,0)) - 1 <= statecount(0) + statecount(1) + statecount(2) + statecount(3) + statecount(4) + statecount(5) + statecount(6) + statecount(7) - statecount((0,0,1)) - 1 }

rule : {(0,0,1)} 100 { cellpos(2)=0 and (0,0,0) != (0,0,1) and statecount(0) + statecount(1) + statecount(2) + statecount(3) + statecount(4) + statecount(5) + statecount(6) + statecount(7) - statecount((0,0,0)) - 1 > statecount(0) + statecount(1) + statecount(2) + statecount(3) + statecount(4) + statecount(5) + statecount(6) + statecount(7) - statecount((0,0,1)) - 1 }

rule : {(0,0,0)} 100 { cellpos(2)=0 and (0,0,0) = (0,0,1) }

[top-rule]

rule : 0 100 { (0,0,0) = 7 }

rule : 1 100 { (0,0,0) = 0 }

rule : 2 100 { (0,0,0) = 1 }

rule : 3 100 { (0,0,0) = 2 }

rule : 4 100 { (0,0,0) = 3 }

rule : 5 100 { (0,0,0) = 4 }

rule : 6 100 { (0,0,0) = 5 }

rule : 7 100 { (0,0,0) = 6 }

As you can see in the growth rule, it makes sure that the cell is on the bottom plane, that the cell above is different from the current cell, and counts all the states of the cells around it, then subtracts the number that are in the same state as the cell and then subtracts 1 for the cell above.

**3.0 Simulation Results**

The simulation was performed with an input file that was created to represent a grain structure of a metal crystal structure. See figure 2 for the initial grain structure.

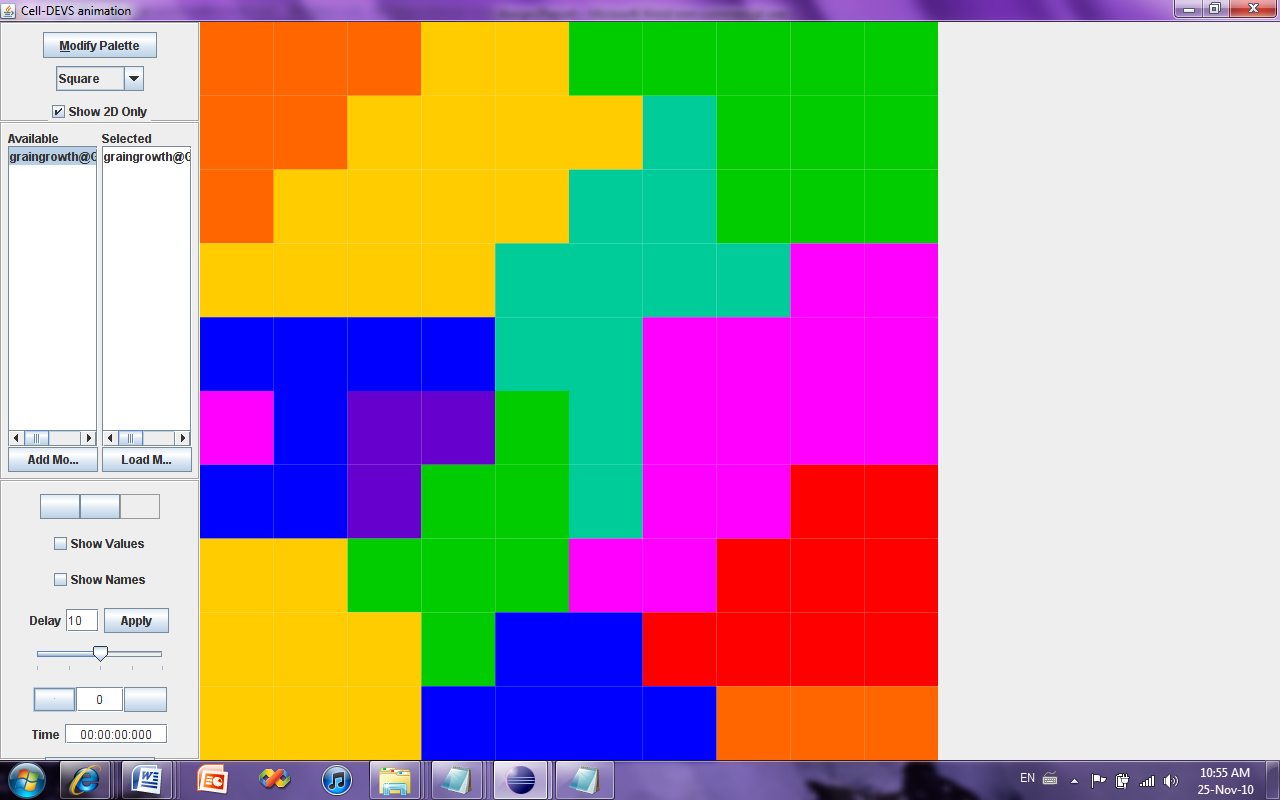
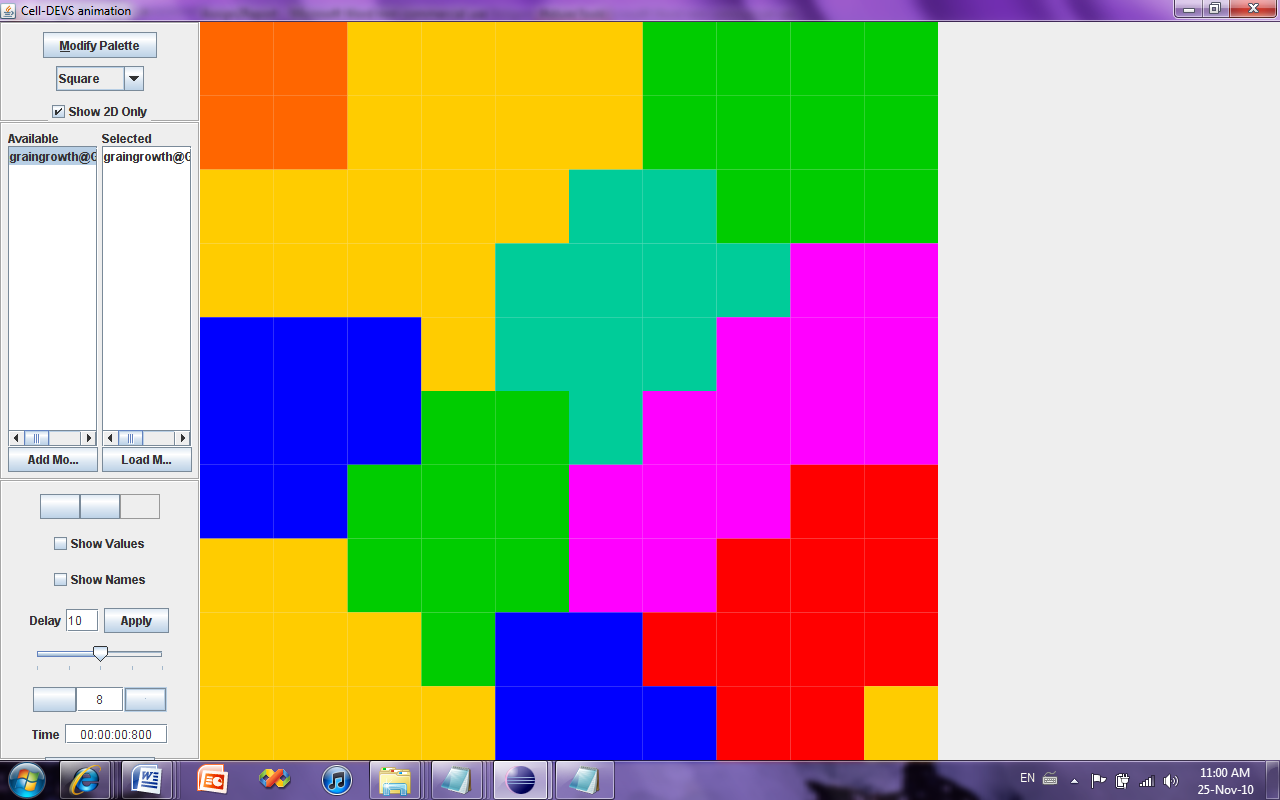
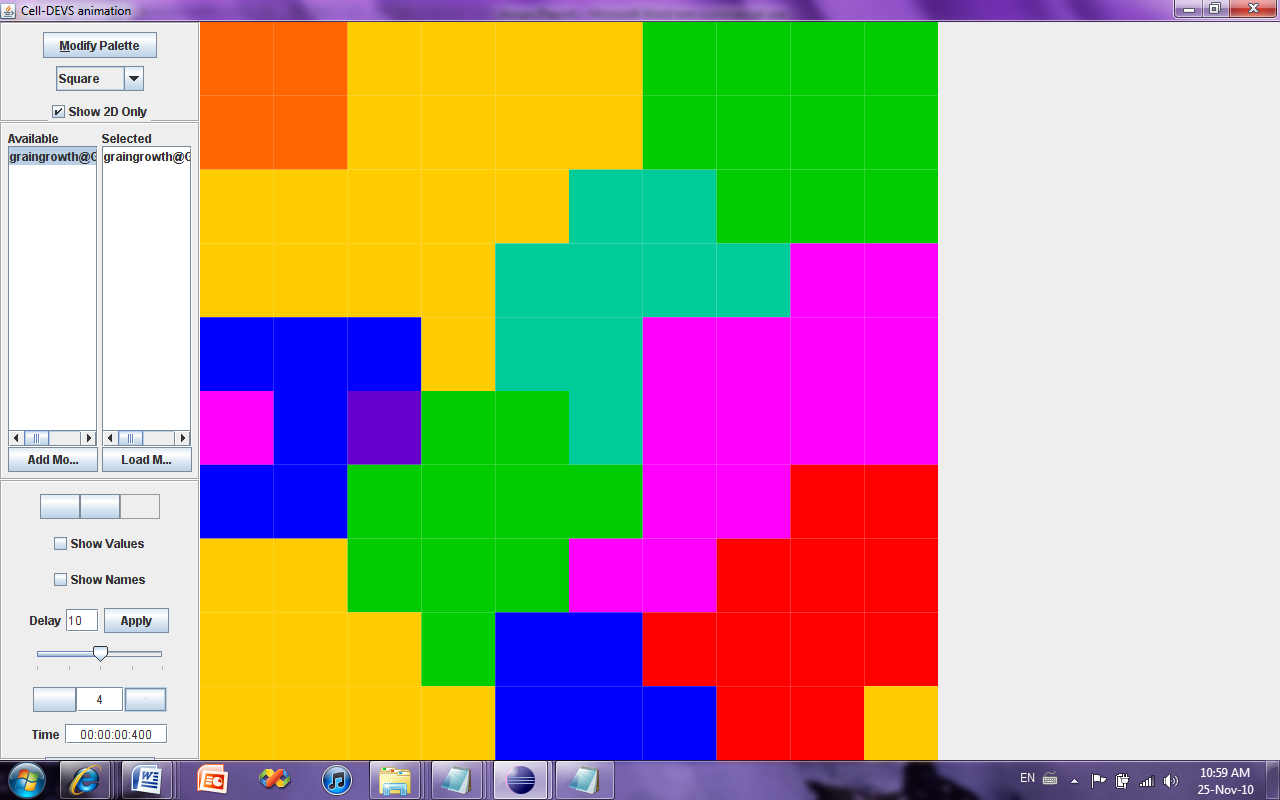


Figure 2: Initial Grain Structure

Each colour in the figure represents a different grain angle orientation. In this simulation we would expect that the grains will grow and consume some of the smaller grains until the equilibrium condition is obtained. The equilibrium will be obtained when the grain boundaries all have angles of 120 degrees. This is the lowest energy configuration and thus the grains will not be changing because energy will only be increased. It can be seen in the following figures how the grain boundaries change.

It can be seen that the small purple grain and the orange grain get consumed by the growing yellow and red grains. In the final step the grain boundaries can be seen to all be at the 120 degree angles, thus making the structure in equilibrium. This is the expected result from the simulation.



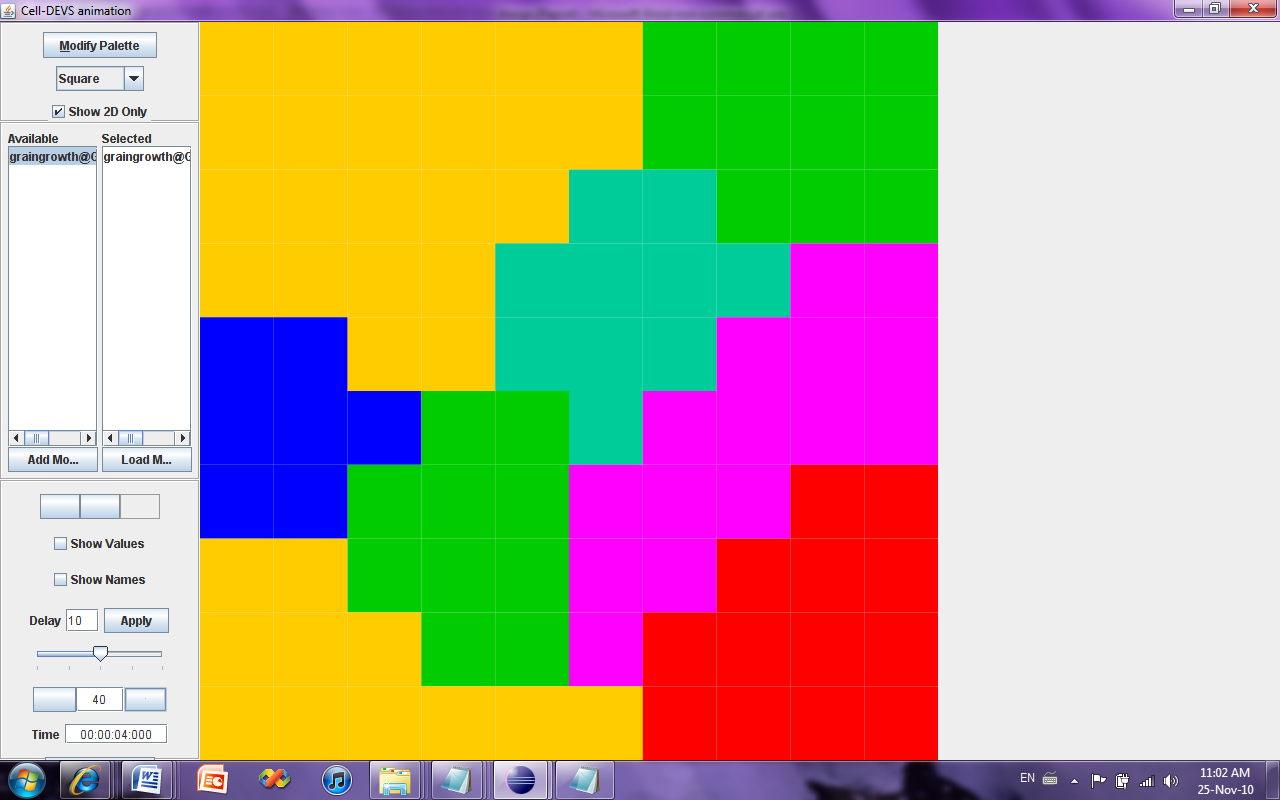
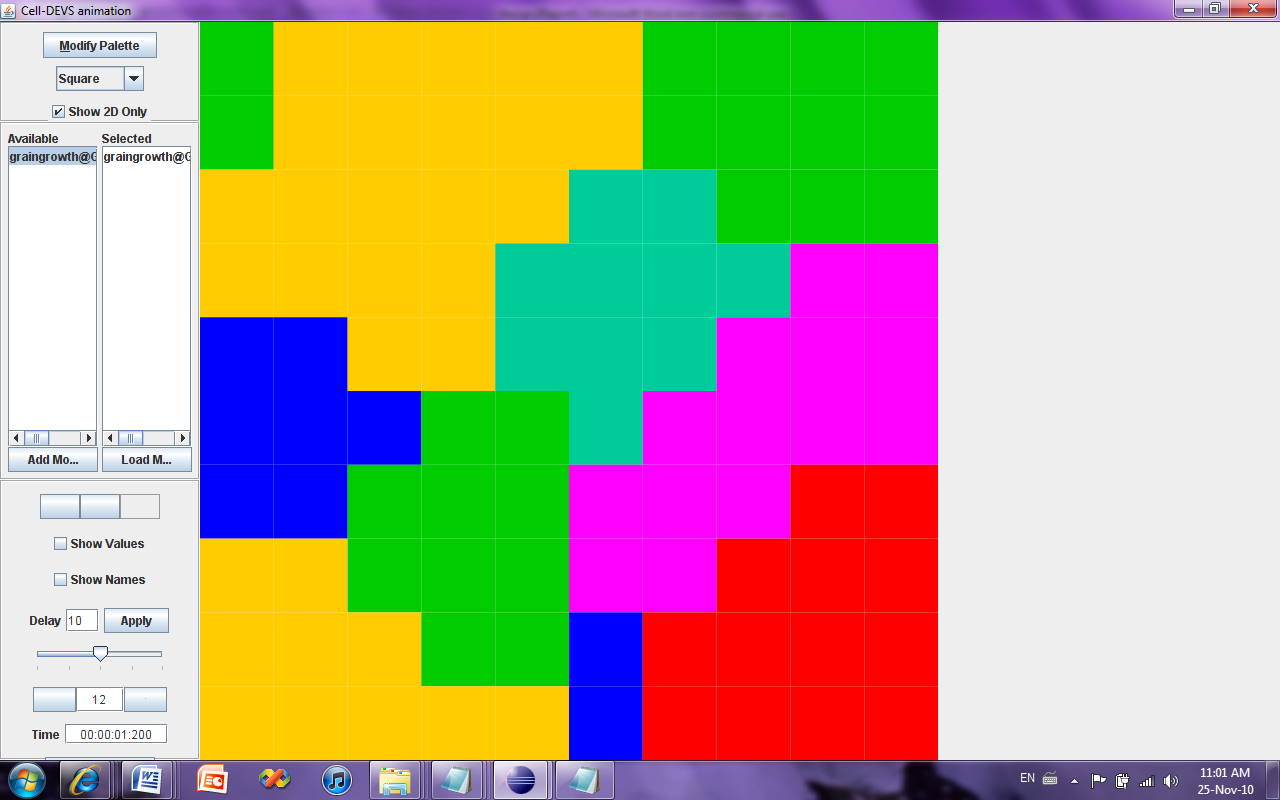


Figure 3: The development of the grain boundary growth

**4.0 References**

[1] Paulo Blikstein, André Paulo Tschiptschin, “Monte Carlo Simulation of Grain Growth”, *Depto. de Eng. Metalúrgica e de Materiais, Escola Politécnica da Universidade de São Paul,* August 15, 1998; Revised: March 30, 1999