November 11, 2014

Team Members

Name	Student Number
Damian Vicino	100916765
Daniella Niyonkuru	100950649

SYSC5104

Methodologies for Discrete-Event Modelling and Simulation

Assignment 2

Free Dendritic Growth in Supercooled Liquids

Presented to

Professor Gabriel Wainer

Fall 2014

Carleton University

Contents

I. Introduction

From causing planes accidents [1] [2] to creating perfect slush beverages and preserving organs [3], supercooling [4] can be both bad and good. Supercooling [5] keeps liquids in the same state below their freezing points without them becoming solid. In a pure liquid such as distilled water, particles (crystal nuclei around which crystal structure can form) responsible of solidification are missing and hence can be supercooled down to -42°C. Yet, supercooled liquids are unstable and will freeze in seconds on contact with tiny particles. For this assignment, we will build a Cell-DEVS model that illustrates what happens when a tiny particle comes in contact with a supercooled liquid, and especially the free dendritic growth that follows. This model can be reused later to study events such as those that affect aeroplanes when supercooled drops of water congregate on the wings and suddenly freeze due to the low temperature at high altitude.

II. Conceptual Model

Free Dendritic Growth

Dendrites [6] [7] are usually observed in materials solidifying with low entropies of fusion and are branching structures that freeze such that dendrite arms grow in specific crystallographic directions. As a dendrite grows, additional side arms can grow behind the growing tip. Dendrites are termed free dendrites when they form individually and grow in supercooled liquids. Growth happens radially leading to an equiaxed shaped grain until collision with another growing dendrite takes place.
Two factors control the growth of the dendrites: Heat liberated at the solid/liquid interface (Hence, heat conduction into the liquid will be modelled), and the local curvature of the solid/liquid interface determines the local equilibrium freezing temperature (The Gibbs-Thomson effect).

Modelling Free Dendritic Growth

In [6], a cellular automaton model is presented and will be the reference for developing our cell-DEVS model. An orthogonal grid will be used. Each cell will have its own temperature and have its state (liquid or solid, respectively 0 or 1) defined. Ultimately, we will have a cubic structure with 100x100x100 elements and run different nucleus (composed of solid state cells) configurations. Initial temperatures will be set to supercooling values for each cell and borders will be wrapped for opposite faces to be in contact with each other. Rules that will be used include:

1) Liquid to solid phase transition only if there are solid sites in the surrounding 8 nearest neighbours
2) Growth occurs if the cell temperature is less than a critical temperature (See equation 2 in [6])

3) For a solid transition, the cell temperature is raised to a temperature to stimulate the heat release
4) Each cell temperature is updated using the modelled conductive heat transfer

Note that the main reference ([6] *) is provided in PDF format along with this description.*

III. Cell-DEVS Definition

Atomic Cell-DEVS Specification

CD = < X, Y, I, S, θ, N, d, δ_{int}, δ_{ext}, τ, λ, D >

X = Ø

Y = Ø

S = {0-20000}

N = neighborhood = {(0,-1,-1), (0,0,-1), (0,1,-1), (1,0,-1), (-1,0,-1), (-1,0,1), (-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,-1,1), (0,0,1), (0,1,1), (1,0,1) }

d = 100 ms

τ: N→S:

S = Phase(cell(0,0,0))*10000 + Temperature(cell(0,0,0)) with

- Phase(cell(0,0,0)) = 1 **and** Temperature(cell(0,0,0)) = cst **if** Phase(cell(0,0,0))=0 **and** # of solid neighbors along X,Y or Z plane >=3 **and** cell's temperature < T_{crit} (T_{crit} : Critical temperature defined by the Gibbs-Thomson effect, cst: constant temperature for heat release)
- Temperature(cell(0,0,0)) = Average (Temperature(cell(0,0,1)), Temperature(cell(0,0,-1)), Temperature(cell(0,1,0)), Temperature(cell(0,-1,0)), Temperature(cell(1,0,0)), Temperature(cell(-1,0,0)))

Note

We have combined two variable states into one. Indeed, the simulation has to update both the phase (i.e. solid or liquid) and the temperature of the state. Since only one state is available in CD++ we combine both by assigning the Most Significant Digit (MSD) to the phase (1 for solid and 0 for liquid as described in the original specification) and the 3 Least Significant Digits(LSD) to the temperature (which is expressed in Kelvin in order to always have positive temperatures). For instance, **10213** indicates a solid (MSD=1) cell for which the temperature (LSD=213) is 213°K (-60°C). Hence, the phase and temperature are computed using the following formulas:

$$\boxed{Phase(S) = \left\lfloor \frac{S}{10000} \right\rfloor \; and\; Temperature(S) = S \bmod 10000}$$

If two states were available, we would have:

- S={0,1} for the phase where 0 indicates the liquid phase and 1 the solid phase and,
- S'={0-999} for the temperatures (i.e. between -273.5°C and 725.5°C which is wide enough to capture the supercooling behavior)

IV. CD++ Implementation

a) State Definition

We use the above-described set of values and mechanism for the states. Hence, the state is always between 0 and 20000. In addition,

- To get Phase(S), we verify if (0,0,0)>10000. If true, Phase(S)=1 meaning that the cell's phase is solid. In the other case, Phase(S)=0 therefore the state is liquid. When a cell becomes solid, the temperature is set to a certain constant value ([6]). *10230* was chosen to model this case and each cell that becomes solid is first set to this value.
- To get the temperature of a cell, we use remainder((0,0,0),10000) which is equivalent to (0,0,0) modulus 10000. The temperature is updated at each neighborhood change.

b) Neighborhood Definition

Each cell has 18 neighbors. The neighborhood corresponds to a cross shaped structure that corresponds to a 3x3 cube deprived from its corners. It is shown in the following figure:

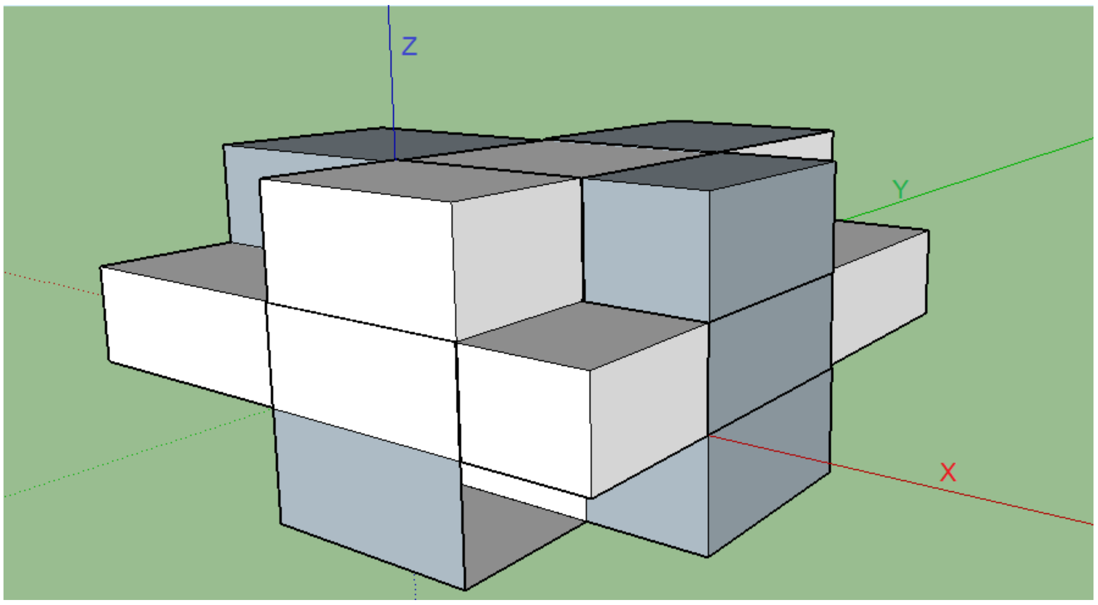
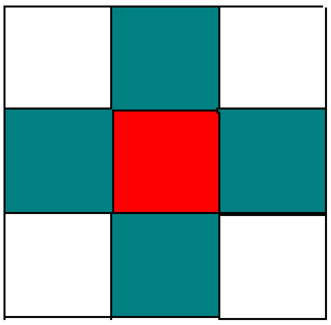
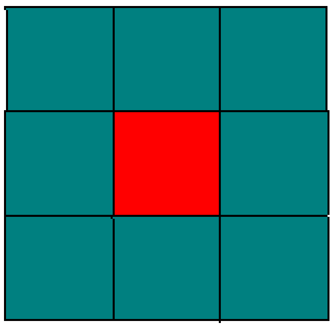


Figure 1. Neighborhood

The faces (front planes) are similar to the Von Neumann neighborhood, while the center plans are identical to Moore’s neighborhood.

 (1)Front View	 (2)Center

- The Von Neumann neighborhood part is useful to simulate heat release, as each cell/cube’s temperature in contact with the cell being evaluated will be used to compute the new temperature of the cell. This procedure uses the front, back, left, right, up and down cells, i.e. cells in contact with each of the six faces of the cell (or cube since we are in 3D). The temperature update process will be more detailed in the rule section.
- The Moore neighborhood part is mainly for spreading the solidification in a radial fashion (up, down, left, right and along the diagonals for each face of the cube)

Hence, the global neighborhood defined in the ma file is:

```
neighbors : supercooling(0,-1,-1)  supercooling(0,0,-1)  supercooling(0,1,-1)
neighbors : supercooling(1,0,-1)   supercooling(-1,0,-1) supercooling(-1,0,1)
neighbors : supercooling(-1,-1,0)  supercooling(-1,0,0)  supercooling(-1,1,0)
neighbors : supercooling(0,-1,0)   supercooling(0,0,0)   supercooling(0,1,0)
neighbors : supercooling(1,-1,0)   supercooling(1,0,0)   supercooling(1,1,0)
neighbors : supercooling(0,-1,1)   supercooling(0,0,1)   supercooling(0,1,1)
neighbors : supercooling(1,0,1)
```

c) Rules

As per the conceptual model, the following rules

1) *Liquid to solid phase transition only if there are solid sites in the surrounding 8 nearest neighbours*
2) *Growth occurs if the cell temperature is less than a critical temperature (See equation 2 in* [6]*)*
3) *For a solid transition, the cell temperature is raised to a temperature to stimulate the heat release*
4) *Each cell temperature is updated using the modelled conductive heat transfer*

were implemented in CD++ using :

```
[supercooling-rule]
rule : 10230 100 {((0,0,0) < 10000) and ( #macro(Cx) >= 3) and (#macro(temperature) <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}
```

```
rule : 10230 100 {((0,0,0) < 10000) and ( #macro(Cy) >= 3) and (#macro(temperature) <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule : 10230 100 {((0,0,0) < 10000) and ( #macro(Cz) >= 3) and (#macro(temperature) <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule : { (0,0,0) - remainder((0,0,0),10000) + ( (remainder((0,0,1),10000)+
remainder((0,0,-1),10000) + remainder((0,1,0),10000) + remainder((0,-1,0),10000) +
remainder((1,0,0),10000) + remainder((-1,0,0),10000))/6) } 100 { t }
```

Conditions are shown in **bold blue**, the value in **bold black** and the transport delay is always 100ms.

Rules for transition from liquid to solid

1), 2), and 3) are combined and expressed in the first three CD++ rules:

- A liquid site transforms into a solid site only if $C_x \geq 3$ and/or $C_y \geq 3$ and/or $C_z \geq 3$ where C_x, C_y, C_z are the number of solids sites present in the surrounding 8 neighbors in X, Y and Z planes respectively. To compute C_x, C_y and C_z, we use three macros named Cx, Cy, Cz. Each of those returns the number of solid sites present in each plane using a Moore neighborhood.
- Growth can only occur of the temperature of the liquid site is less than a critical temperature computed using the following equation that models the Gibbs-Thomson effect:

$$T_{crit} = -\gamma\left(f(C_x) + f(C_y) + f(C_z)\right)$$

where γ is a constant for the solid/liquid interfacial energy ($\gamma = 20$ for our simulation),

$$\text{And } f(C_i) = \frac{1}{C_i} if\ C_i \geq 1, f(C_i) = \frac{1}{C_i} if\ C_i < 1$$

These two conditions are expressed by:

```
{((0,0,0) < 10000) and ( #macro(Cx) >= 3) and (#macro(temperature) < CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}
```
(a)

```
{((0,0,0) < 10000) and ( #macro(Cy) >= 3) and (#macro(temperature) < CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}
```
(b)

```
{((0,0,0) < 10000) and ( #macro(Cz) >= 3) and (#macro(temperature) < CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}
```
(c)

Conditions (a), (b) and (c) have three parts each: the first part, i.e. ((0,0,0) < 10000), asserts if the cell is liquid. The second part verifies the number of solid sites present in the X plane (**(a)** verifies the X plane, **(b)** the Y plane and **(c)** the Z plane) Finally, the last part verifies if the temperature of the cell is less than the critical temperature (note that the critical temperature is first transformed into Kelvins using the function CToK). Both parts must be true for a liquid site to become solid

- If a liquid element transforms into a solid, the temperature of the element is raised to a fixed value to simulate the release of latent heat. We use a constant of -43^{o}C or 230^{o}K for this purpose. Hence, the cell state becomes 10230 after a solid transition

Rule for conductive heat transfer

The last rule operates conductive heat transfer. It sets the temperature of the cell to the average of the six nearest neighbor elements. The new state becomes:

```
{ (0,0,0) - remainder((0,0,0),10000) + ( (remainder((0,0,1),10000)+ remainder((0,0,-1),10000) + remainder((0,1,0),10000) + remainder((0,-1,0),10000) + remainder((1,0,0),10000) + remainder((-1,0,0),10000))/6) }
```

where `(0,0,0)-remainder((0,0,0),10000)` represents the cell state and is therefore 0 for liquid sites, 10000 for solid sites. The following part is the new temperature and is the average of the temperatures of the cells located up, down, left, right, in front and at the back of the evaluated cell.

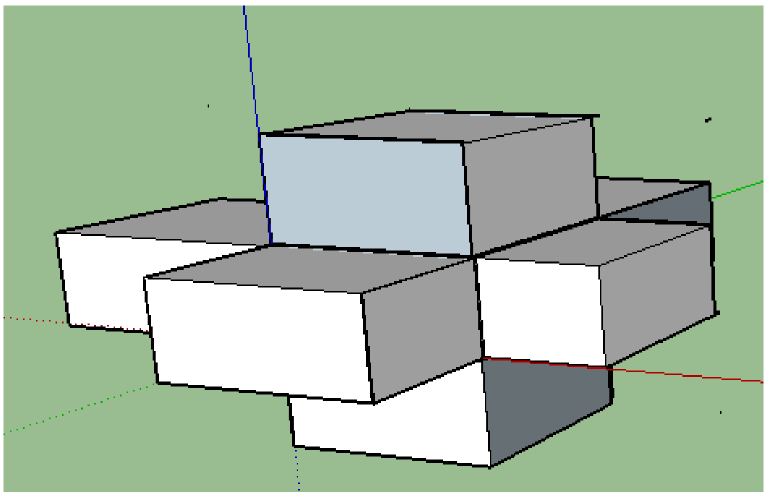


Figure 2. Cells used for Conductive Heat Transfer

Note that only conductive heat transfer was modeled here; convective heat flow was not modeled.

d) Macros

Cx, Cy and Cz macros

These macros return the number of solid sites present along the X, Y, Z planes in the neighborhood.

```
#BeginMacro(Cx)
(  trunc((0,1,-1)/10000)  +  trunc((0,1,0)/10000)  +  trunc((0,1,1)/10000)  +  trunc((0,-1,-
1)/10000)  +  trunc((0,-1,0)/10000)  +  trunc((0,-1,1)/10000)  +  trunc((0,0,-1)/10000)  +
trunc((0,0,1)/10000) )
#EndMacro

#BeginMacro(Cy)
(  trunc((1,0,-1)/10000)  +  trunc((1,0,0)/10000)  +  trunc((1,0,1)/10000)  +  trunc((-1,0,-
1)/10000)  +  trunc((-1,0,0)/10000)  +  trunc((-1,0,1)/10000)  +  trunc((0,0,-1)/10000)  +
trunc((0,0,1)/10000) )
#EndMacro

#BeginMacro(Cz)
(  trunc((-1,-1,0)/10000)  +  trunc((-1,0,0)/10000)  +  trunc((-1,1,0)/10000)  +  trunc((1,-
1,0)/10000)  +  trunc((1,0,0)/10000)  +  trunc((1,1,0)/10000)  +  trunc((0,-1,0)/10000)  +
trunc((0,1,0)/10000) )
#EndMacro
```

Note: The *trunc* function rounds down the quotient, and is used to get the state of the cell.

Each of the evaluated planes is shown in Figure 3 below. The colors show the cells that are evaluated and are equivalent to Moore neighborhoods obtained in different planes.

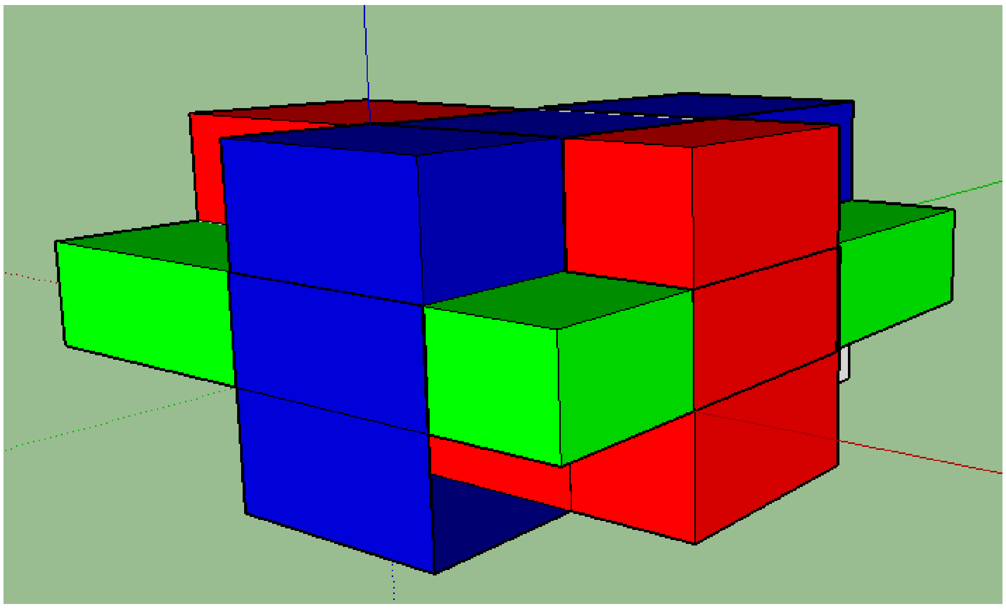


Figure 3. Moore Neighborhood in different planes

Temperature macro

Returns the temperature of the cell

```
#BeginMacro(temperature)
remainder((0,0,0),10000)
#EndMacro
```

Phase macro

Returns true if the cell is liquid and false if the cell is solid

```
#BeginMacro(isLiquid)
((0,0,0) < 10000)
#EndMacro
```

All the above macros are included in the *SupercoolingMacros.inc* file

e) Variation from the original specification

The main variation from the original specification is in the state that combines both the temperature and the phase because only one state variable is available in CD++. We decided to use this approach since it provides a simple way to simulate the model, without creating additional layers. This last option would require more memory and processing time that we cannot afford since we are in 3D and our simulations should be able to run 100x100x100 cells. We also use Kelvin degrees instead of Celsius degrees to keep positive temperatures and handle temperature simply.

f) MA file

The resulting MA file for a 100x100x100, a 3x3x3 nucleus placed in the centre (defined in supercooling.val), an initial temperature of -60^{o}C (213^{o}K) at which all the cells at set at is shown below. The borders are wrapped as per the original specification.

```
#include(SupercoolingMacros.inc)

[top]
components : supercooling

[supercooling]
type : cell
dim : (100,100,100)
delay : transport
defaultDelayTime  : 100
border : wrapped
neighbors : supercooling(0,-1,-1)  supercooling(0,0,-1)  supercooling(0,1,-1)
neighbors : supercooling(1,0,-1)   supercooling(-1,0,-1) supercooling(-1,0,1)
neighbors : supercooling(-1,-1,0)  supercooling(-1,0,0)  supercooling(-1,1,0)
neighbors : supercooling(0,-1,0)   supercooling(0,0,0)   supercooling(0,1,0)
neighbors : supercooling(1,-1,0)   supercooling(1,0,0)   supercooling(1,1,0)
neighbors : supercooling(0,-1,1)   supercooling(0,0,1)   supercooling(0,1,1)
neighbors : supercooling(1,0,1)
initialvalue : 213
initialCellsValue : supercooling.val
localtransition : supercooling-rule

[supercooling-rule]
rule  :  10230  100  {((0,0,0)  <  10000)  and  (  #macro(Cx)  >  2)  and  (#macro(temperature)  <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule  :  10230  100  {((0,0,0)  <  10000)  and  (  #macro(Cy)  >  2)  and  (#macro(temperature)  <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule  :  10230  100  {((0,0,0)  <  10000)  and  (  #macro(Cz)  >  2)  and  (#macro(temperature)  <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule   :   {   (0,0,0)   -   remainder((0,0,0),10000)   +   (   (remainder((0,0,1),10000)+
remainder((0,0,-1),10000)   +   remainder((0,1,0),10000)   +   remainder((0,-1,0),10000)   +
remainder((1,0,0),10000) + remainder((-1,0,0),10000))/6) } 100 { t }
```

g) VAL file

The initial nucleus, i.e. the seed from which supercooling is initiated is defined in the val file. This is done by setting the desired cells' value to 10000 added to the initial temperature. The VAL file that corresponds to the previous MA file is:

```
(48,48,48) = 10213
(48,48,49) = 10213
(48,48,50) = 10213
(48,49,48) = 10213
(48,49,49) = 10213
(48,49,50) = 10213
(48,50,48) = 10213
(48,50,49) = 10213
(48,50,50) = 10213
(49,48,48) = 10213
(49,48,49) = 10213
(49,48,50) = 10213
(49,49,48) = 10213
(49,49,49) = 10213
(49,49,50) = 10213
(49,50,48) = 10213
(49,50,49) = 10213
(49,50,50) = 10213
(50,48,48) = 10213
(50,48,49) = 10213
(50,48,50) = 10213
(50,49,48) = 10213
(50,49,49) = 10213
(50,49,50) = 10213
(50,50,48) = 10213
(50,50,49) = 10213
(50,50,50) = 10213
```

It shows a 3x3x3 nucleus placed in the center of a supercooled liquid at 213^{o}K.

h) Automating the MA and VAL file generation

Since, we want to run multiple experiments with different temperatures (varying from -60^{o}C to -32^{o}C) and different nucleus (size and location), scripts were written to automatically generate MA and VAL files for different experiments. The prototype used for the model file generation is shown below:

```
#include(SupercoolingMacros.inc)

[top]
components : supercooling

[supercooling]
type : cell
dim : (DIM_VAL,DIM_VAL,DIM_VAL)
delay : transport
defaultDelayTime  : 100
border : wrapped
neighbors : supercooling(0,-1,-1)  supercooling(0,0,-1)  supercooling(0,1,-1)
neighbors : supercooling(1,0,-1)   supercooling(-1,0,-1) supercooling(-1,0,1)
neighbors : supercooling(-1,-1,0)  supercooling(-1,0,0)  supercooling(-1,1,0)
neighbors : supercooling(0,-1,0)   supercooling(0,0,0)   supercooling(0,1,0)
neighbors : supercooling(1,-1,0)   supercooling(1,0,0)   supercooling(1,1,0)
neighbors : supercooling(0,-1,1)   supercooling(0,0,1)   supercooling(0,1,1)
neighbors : supercooling(1,0,1)
initialvalue : INIT_VAL
initialCellsValue : supercooling_INIT_CELLS_VAL.val
localtransition : supercooling-rule

[supercooling-rule]
rule  :  10230  100  {((0,0,0)  <  10000)  and  (  #macro(Cx)  >  2)  and  (#macro(temperature)  <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule  :  10230  100  {((0,0,0)  <  10000)  and  (  #macro(Cy)  >  2)  and  (#macro(temperature)  <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule  :  10230  100  {((0,0,0)  <  10000)  and  (  #macro(Cz)  >  2)  and  (#macro(temperature)  <
CToK(-20*(
if(#macro(Cx)>0,1/#macro(Cx),0)+if(#macro(Cy)>0,1/#macro(Cy),0)+if(#macro(Cz)>0,1/#macro(C
z),0) ) ))}

rule   :   {   (0,0,0)   -   remainder((0,0,0),10000)   +   (   (remainder((0,0,1),10000)+
remainder((0,0,-1),10000)   +   remainder((0,1,0),10000)   +   remainder((0,-1,0),10000)   +
remainder((1,0,0),10000) + remainder((-1,0,0),10000))/6) } 100 { t }
```

The parameters that should be provided are *DIM_VAL*, *INIT_VAL* and *supercooling_INIT_CELLS_VAL.val*. DIM_VAL is the for the dimensions of the model, INIT_VAL is for the initial temperature and supercooling_INIT_CELLS_VAL.val specifies the val file that should be used. This file is also autogenerated. The experiments and scripts used to run them will be described in the following section.

V. Experiments

Multiple experiments were run in order to test the correctness of our model and also exhibit the relationship between the initial supercooling temperatures and the dendritic growth velocity. A set of simulations were performed using initial supercooling temperatures ranging from -60 to -32^{o}C. In addition to the temperatures variation, we performed simulations using different dimensions: 7,15,25,49 and 99. Different configurations (location and size) of the nucleus were tested: Nucleus of size 1 and 3, plus center and corner locations. The scripts used to generate these experiments will first be described, and then we will present the results.

a) Scripts

File generation

The following script was used to generate the needed MA and VAL files.

```
#!/bin/bash
echo '#!/bin/bash' > exp_runner.sh
for d in 7 15 25 49 99; do
    for t in -32 -35 -38 -39 -40 -42 -43 -44 -45 -47 -48 -49 -50 -51 -52 -55 -57
-60; do
        for core in 1 3; do
            echo Creating the MA file Dim: DIM_VAL: ${d}, Temp: INIT_VAL: $((273
+ $t)), INIT_CELLS_VAL: ${t}_${d}_${core}
            cat proto_supercooling.ma | sed "s/DIM_VAL/${d}/g" | sed
"s/INIT_VAL/$((273+$t))/g" | sed "s/INIT_CELLS_VAL/${t}_${d}_${core}/g" >
supercooler_${t}_${d}_${core}.ma
            echo Registering runner
            echo
'/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/simu
Orig.exe  -msupercooler_'${t}_${d}_${core}'.ma -
lsupercooler_'${t}_${d}_${core}'.log' >> exp_runner.sh
            echo
'/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/draw
log.exe  -msupercooler_'${t}_${d}_${core}'.ma -csupercooling -
lsupercooler_'${t}_${d}_${core}'.log > supercooler_'${t}_${d}_${core}'.drw' >>
exp_runner.sh
        done
        echo Creating initial vals
        #size 1
        echo '('$((${d}/2)),$((${d}/2)),$((${d}/2))') = 10'$((${t}+273)) >
supercooling_${t}_${d}_1.val
        echo '' >> supercooling_${t}_${d}_1.val
        : > supercooling_${t}_${d}_3.val
        for a in `seq $((${d}/2-1)) 1 $((${d}/2+1))`; do
            for b in `seq $((${d}/2-1)) 1 $((${d}/2+1))`; do
                for c in `seq $((${d}/2-1)) 1 $((${d}/2+1))`; do
                    echo '('${a},${b},${c}') = 10'$((${t}+273)) >>
supercooling_${t}_${d}_3.val
                done
            done
        done
        echo '' >> supercooling_${t}_${d}_3.val
    done
done
```

A script named *exp_runner.sh* is generated and will run the experiments. The first for loop has a variable d that iterates through different dimension sizes; the second loop uses a variable t for different temperature needed for our experiments; the last loop sets the size of the nucleus. Corresponding MA files are generated using the prototype described section IV.h. At the same time, experiments are added to *exp_runner* in order to call CD++ simu engine and drawlog tool, and generate log and drw files. The VAL file is generated as well; the above script generate nucleus of size 1 and 3 places in the center of the grid. A variant that places the nucleus in a corner location is also available.

Experiment Runner

This script is generated by the previous one and contains instructions to run experiments. This script can be partitioned and run across different computers in order to gather results quickly. Here is a snippet of the runner script:

```
#!/bin/bash
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/simuO
rig.exe  -msupercooler_-60_49_1.ma -lsupercooler_-60_49_1.log
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/drawl
og.exe  -msupercooler_-60_49_1.ma -csupercooling -lsupercooler_-60_49_1.log >
supercooler_-60_49_1.drw
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/simuO
rig.exe  -msupercooler_-60_49_3.ma -lsupercooler_-60_49_3.log
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/drawl
og.exe  -msupercooler_-60_49_3.ma -csupercooling -lsupercooler_-60_49_3.log >
supercooler_-60_49_3.drw
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/simuO
rig.exe  -msupercooler_-57_49_1.ma -lsupercooler_-57_49_1.log
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/drawl
og.exe  -msupercooler_-57_49_1.ma -csupercooling -lsupercooler_-57_49_1.log >
supercooler_-57_49_1.drw
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/simuO
rig.exe  -msupercooler_-57_49_3.ma -lsupercooler_-57_49_3.log
/cygdrive/c/eclipse/plugins/cdBuilder.simulator_1.0.0.201109022248/internal/drawl
og.exe  -msupercooler_-57_49_3.ma -csupercooling -lsupercooler_-57_49_3.log >
supercooler_-57_49_3.drw
```

simuOrig.exe is called to run the simulation using the specified MA file, and generate the log file. Note that we have also added a stop time, and different width and precision parameters to speed up the simulation. *Drawlog.exe* output a drw file from the specified ma, model name and log file.

VI. Simulation Results

a) CD++ Modeller

The first observations will be presented using a 7x7x7 model to provide an easier visualization (Larger models do not easily load in CD++ Modeller and are harder to visualize in 3D).

- Impact of the nucleus size

A small nucleus will have no effect and will not initiate dendritic growth. The top model has a 1x1x1 nucleus while the bottom one has a 3x3x3 as illustrated in the first figure. The dendritic growth only occurs for the second case, which is completely solid by the end of the experiment. Note that since no heat convective model was implemented, the solid particle does not melt, the solution stays steady. The initial temperature is set to -35^{o}C for the results shown in the figure. Each of the seven layers (z=0, z=1…z=7) are shown horizontally. Navy blue indicates solid sites while light blue is for liquid cells.

a) Initial state, Time=00:00:00:000

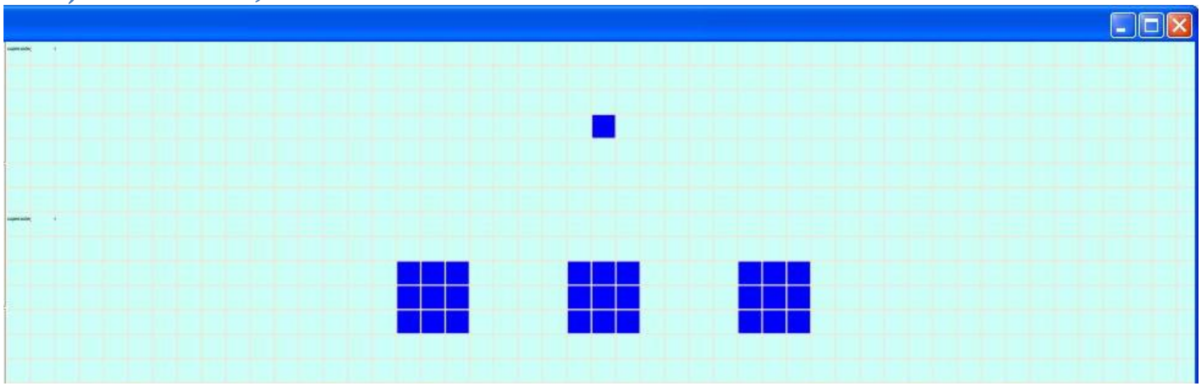
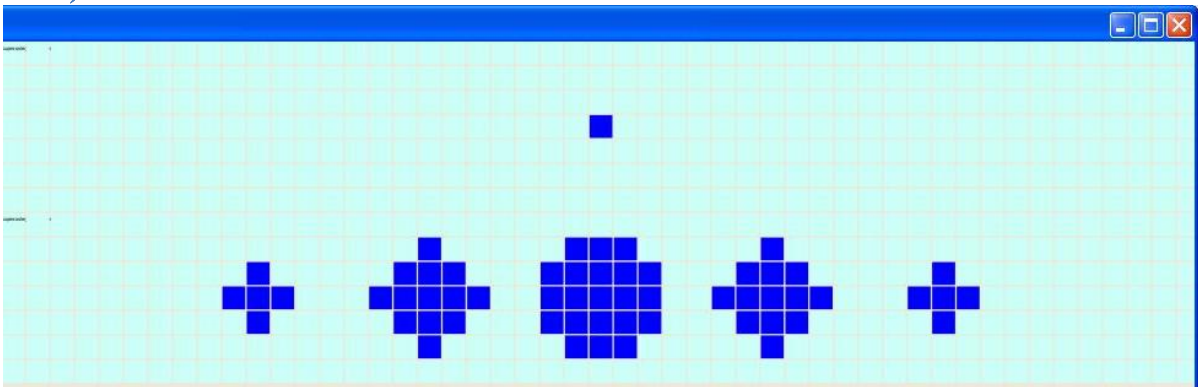
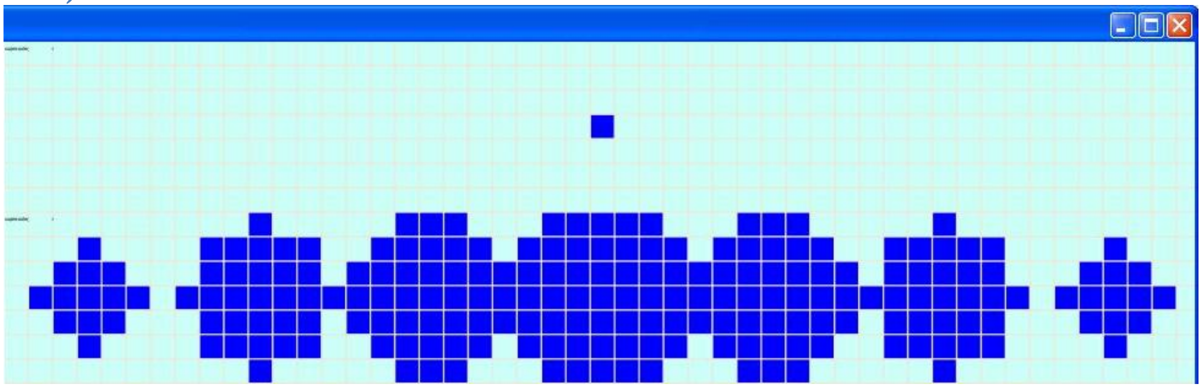


Figure 4. 1x1x1 nucleus for the top model, 3x3x3 nucleus for bottom model

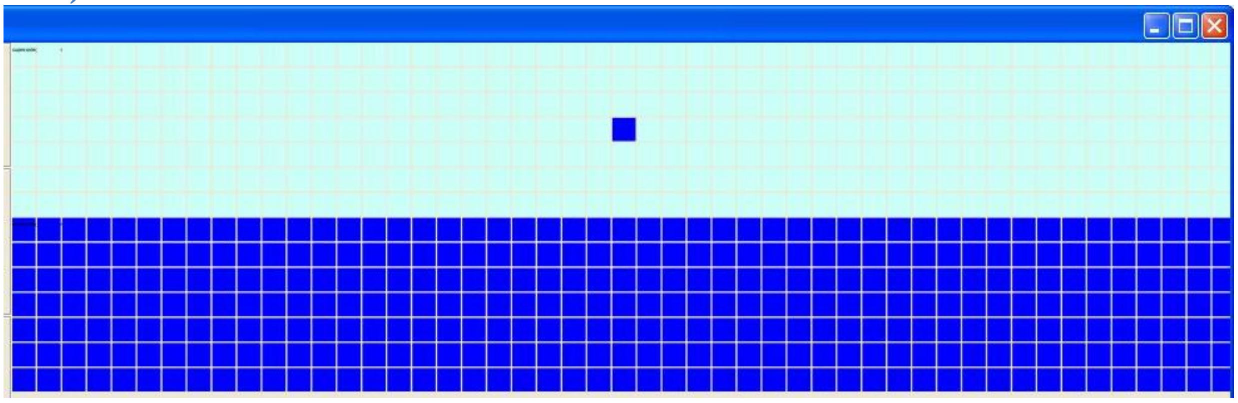
b) Time=00:00:00:100



c) Time=00:00:00:400



d) Time=00:00:00:800

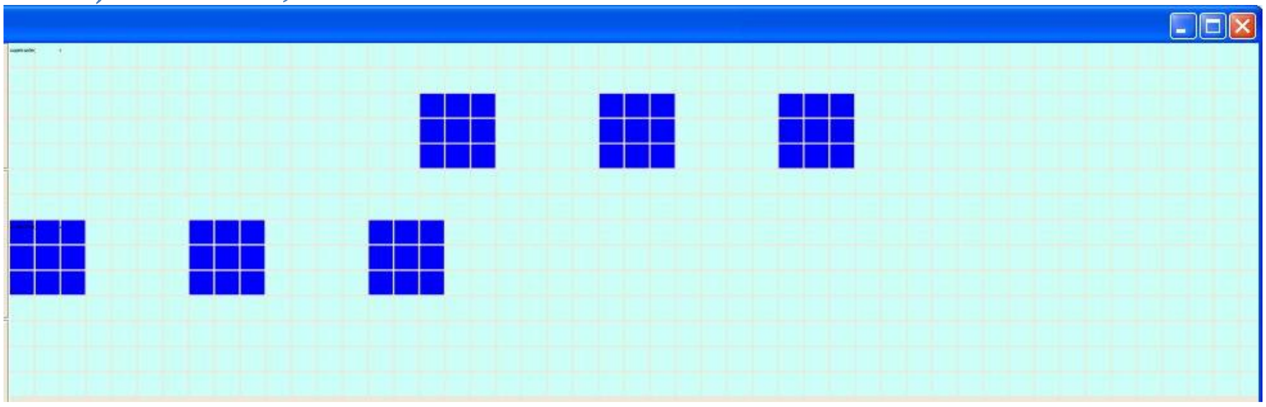


Therefore, the nucleus size affects the dendritic growth. It should be large enough to cause Cx or Cy or Cz to be equal or greater than 3

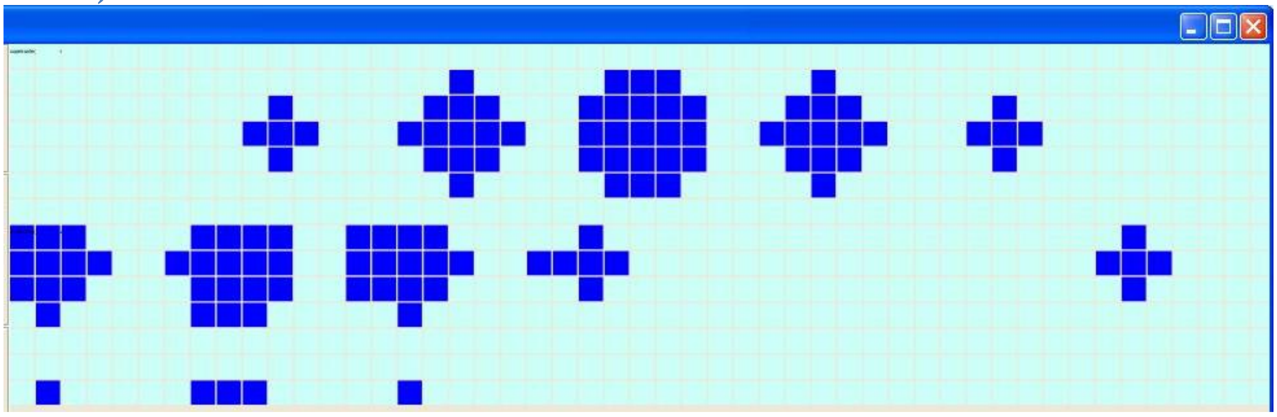
- <u>Impact of the nucleus location</u>

For the results shown below, we use the same initial temperature as previously (-35°C). This time the top view has the 3x3x3 in its center, while the bottom one shows a 3x3x3 nucleus placed on the left front corner.

a) **Initial state, Time=00:00:00:000**

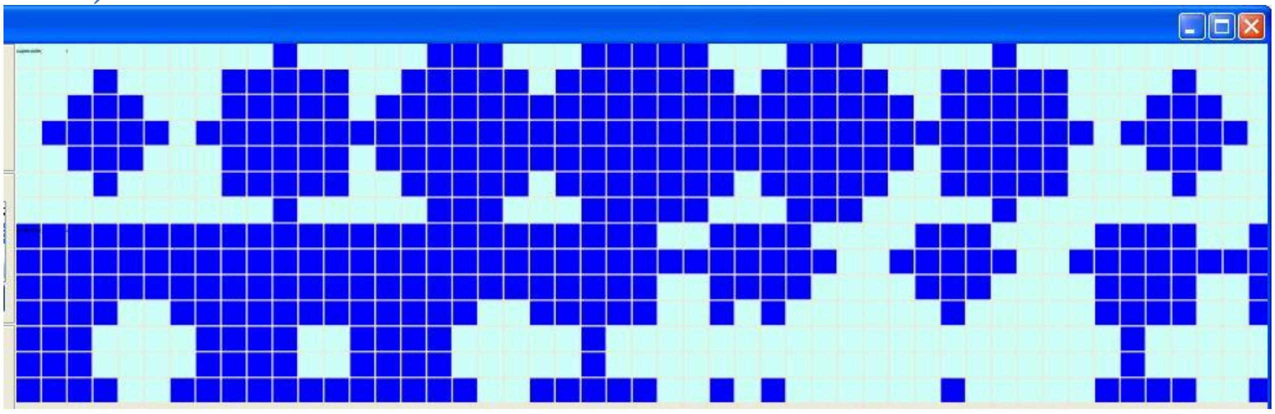


b) **Time=00:00:00:100**



We notice that the centered model spread quickly and produces 30 new solid sites at time 100ms. The non-centered one generates 18 new solid sites (the dendrites that grow from the wrapped borders are not counted in this number). We can also notice that since our model is wrapped new growth will appear on the opposite faces.

c) **Time=00:00:00:400**

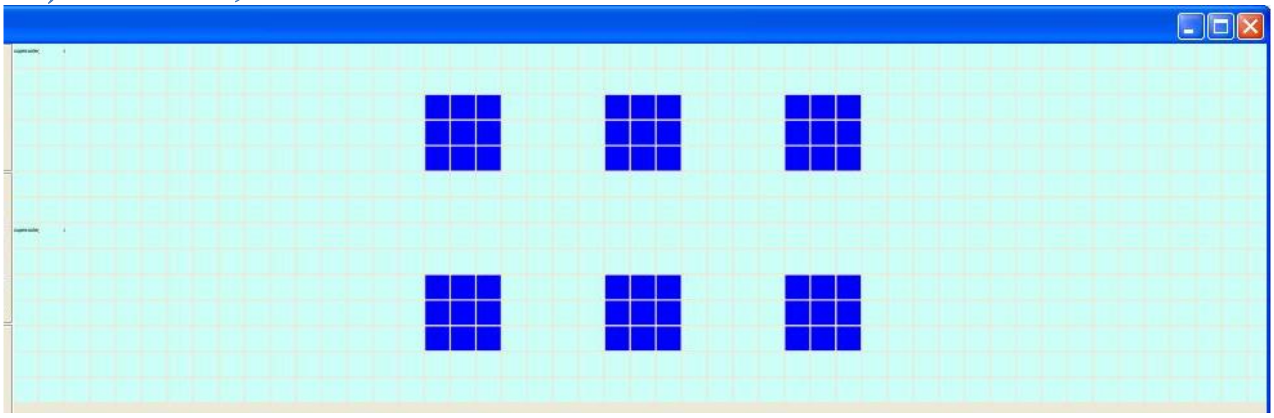


The wrapped model allows equivalent speed for both cases. A non-wrapped one however, would result in a slower growth for the non-centered nucleus.

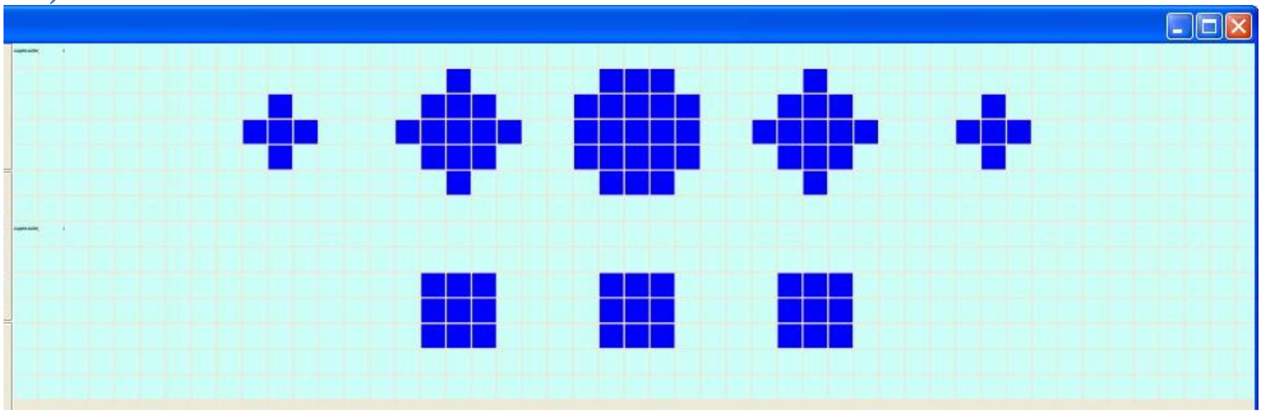
- Impact of a non-supercooling initial temperature

To analyze the impact of a non-supercooling temperature, we use a model with a -35ºC initial temperature (see top part), and another with an initial temperature of 39ºC at which the liquid is not supercooled. Apart from the initial temperature, these models have the same parameters.

a) Initial state, Time=00:00:00:000

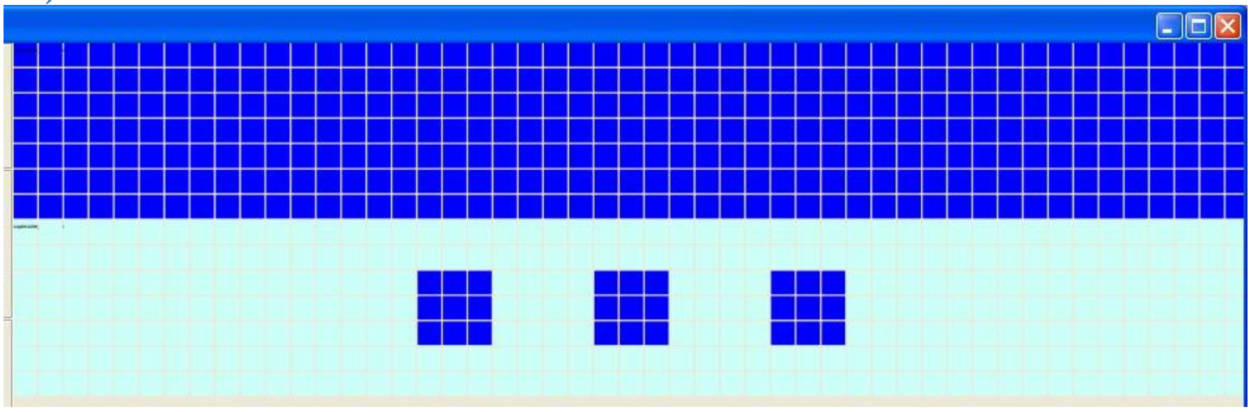


b) Time=00:00:00:100



Dendritric growth occurs only in the -35ºC temperature while the 39ºC one remains unchanged

c) Time=00:00:00:800

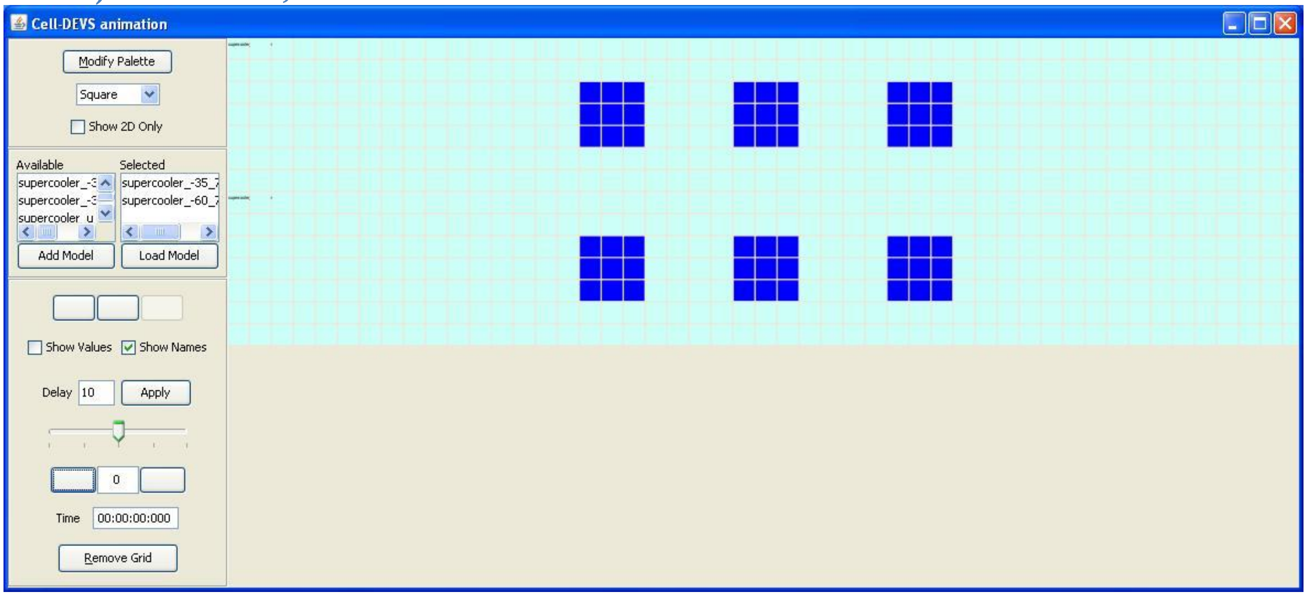


At the end, we notice that the -35ºC model is completely solid while the other model stays liquid.

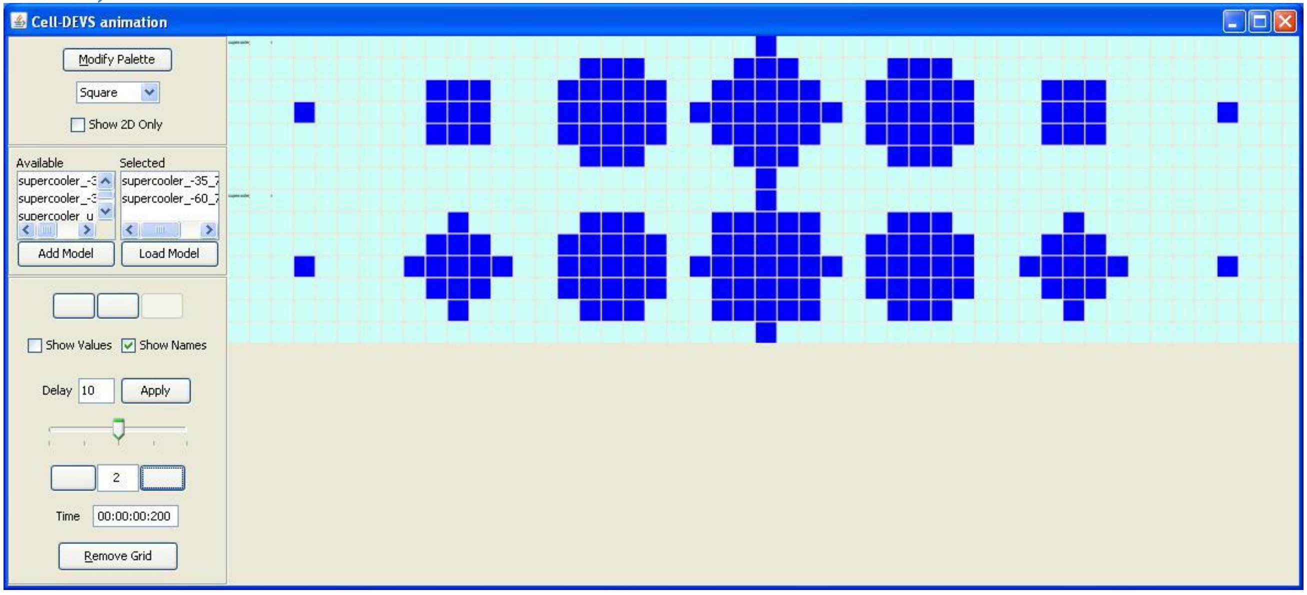
- Initial temperature and Dendritic Growth

For this experiment, we use models with different initial supercooling temperature. The model shown on top has a -35ºC initial temperature, while the one on the bottom has an initial temperature of -60ºC. Both have the same initial configuration as shown in the figure at time 00:00:00:000.

a) Initial state, Time=00:00:00:000

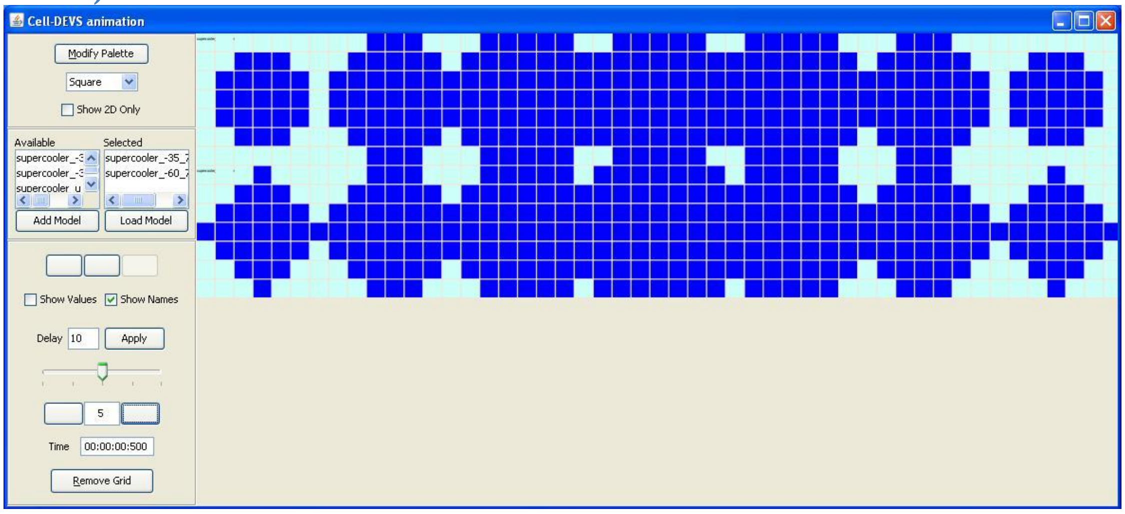


b) Time=00:00:00:200



We notice that the models have different growth patterns. In addition to that, the -60ºC that is the coldest one growth rapidly compared to the -32ºC.

c) Time=00:00:00:500



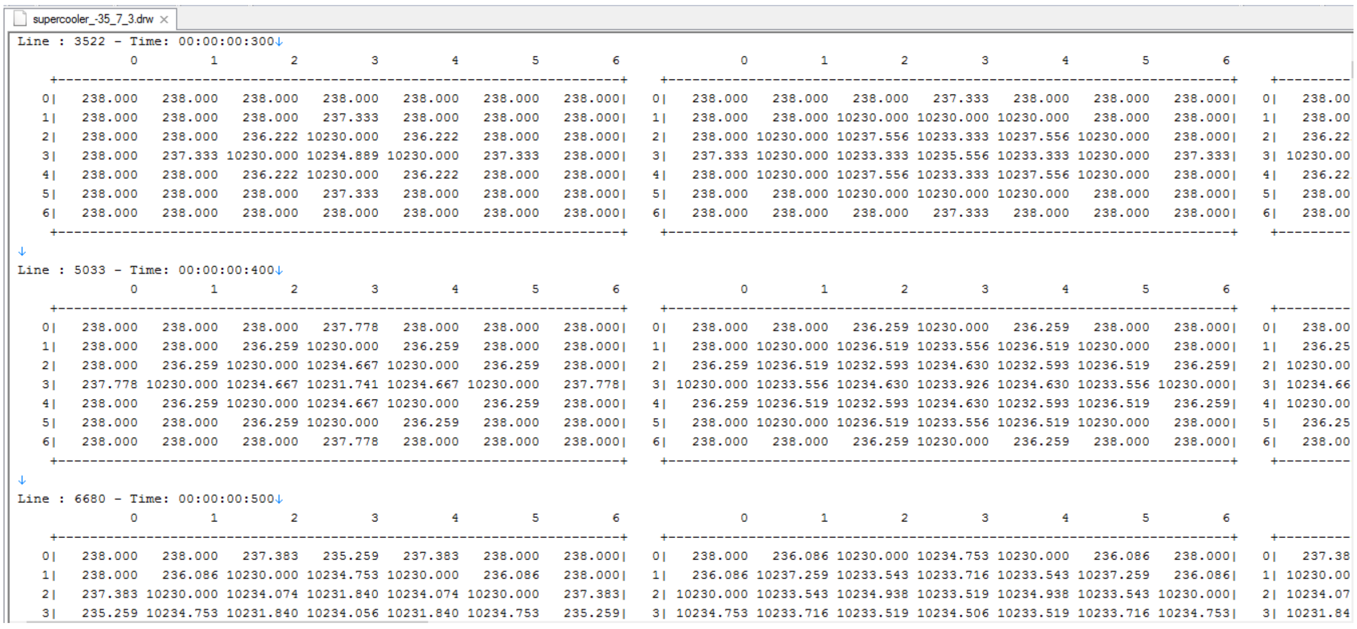
Delay 10
5
Time 00:00:00:500

Different patterns are observed here as well. The -60ºC looks quite packed and spherical. For observing patterns similar to the reference in the paper, larger models are needed and have been run. Logs are available. For smaller dimensions and close temperatures, simulations may look similar as in this video. The difference becomes more obvious as the model becomes larger.

We were not able to run 100x100x100 models as in [6] using CD++; the largest model we were able to simulate was 99x99x5. The simulation took more than 2 days and the generated log file was larger than 120GB.

- <u>Conductive heat transfer</u>

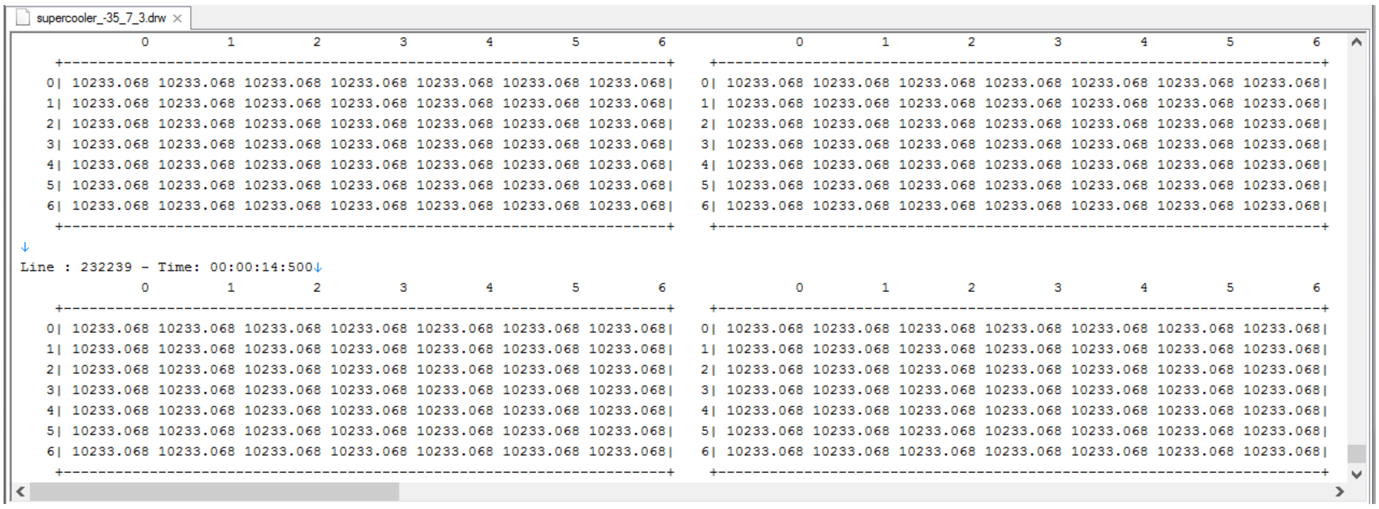
Although the previous cells focused on the phase of the cell, we can also follow the conductive heat transfer process by looking at the *drw* file or by showing the values of the cells. We notice that temperatures are being averaged at each time step as illustrated in the following figure (-35 supercooling temperature, 7x7x7 nucleus).



supercooler_-35_7_3.drw

```
Line : 3522 - Time: 00:00:00:300↓
              0         1         2         3         4         5         6                   0         1         2         3         4         5         6
      +----------------------------------------------------------------------+   +----------------------------------------------------------------------+   +---------
     0|   238.000   238.000   238.000   238.000   238.000   238.000   238.000|  0|   238.000   238.000   238.000   237.333   238.000   238.000   238.000|  0|   238.00
     1|   238.000   238.000   238.000   237.333   238.000   238.000   238.000|  1|   238.000   238.000 10230.000 10230.000 10230.000   238.000   238.000|  1|   238.00
     2|   238.000   238.000   236.222 10230.000   236.222   238.000   238.000|  2|   238.000 10230.000 10237.556 10233.333 10237.556 10230.000   238.000|  2|   236.22
     3|   238.000   237.333 10230.000 10234.889 10230.000   237.333   238.000|  3|   237.333 10230.000 10233.333 10235.556 10233.333 10230.000   237.333|  3| 10230.00
     4|   238.000   238.000   236.222 10230.000   236.222   238.000   238.000|  4|   238.000 10230.000 10237.556 10233.333 10237.556 10230.000   238.000|  4|   236.22
     5|   238.000   238.000   238.000   237.333   238.000   238.000   238.000|  5|   238.000   238.000 10230.000 10230.000 10230.000   238.000   238.000|  5|   238.00
     6|   238.000   238.000   238.000   238.000   238.000   238.000   238.000|  6|   238.000   238.000   238.000   237.333   238.000   238.000   238.000|  6|   238.00
      +----------------------------------------------------------------------+   +----------------------------------------------------------------------+   +---------
↓
Line : 5033 - Time: 00:00:00:400↓
              0         1         2         3         4         5         6                   0         1         2         3         4         5         6
      +----------------------------------------------------------------------+   +----------------------------------------------------------------------+   +---------
     0|   238.000   238.000   238.000   237.778   238.000   238.000   238.000|  0|   238.000   238.000   236.259 10230.000   236.259   238.000   238.000|  0|   238.00
     1|   238.000   238.000   236.259 10230.000   236.259   238.000   238.000|  1|   238.000 10230.000 10236.519 10233.556 10236.519 10230.000   238.000|  1|   236.25
     2|   238.000   236.259 10230.000 10234.667 10230.000   236.259   238.000|  2|   236.259 10236.519 10232.593 10234.630 10232.593 10236.519   236.259|  2| 10230.00
     3|   237.778 10230.000 10234.667 10231.741 10234.667 10230.000   237.778|  3| 10230.000 10233.556 10234.630 10233.926 10234.630 10233.556 10230.000|  3| 10234.66
     4|   238.000   236.259 10230.000 10234.667 10230.000   236.259   238.000|  4|   236.259 10236.519 10232.593 10234.630 10232.593 10236.519   236.259|  4| 10230.00
     5|   238.000   238.000   236.259 10230.000   236.259   238.000   238.000|  5|   238.000 10230.000 10236.519 10233.556 10236.519 10230.000   238.000|  5|   236.25
     6|   238.000   238.000   238.000   237.778   238.000   238.000   238.000|  6|   238.000   238.000   236.259 10230.000   236.259   238.000   238.000|  6|   238.00
      +----------------------------------------------------------------------+   +----------------------------------------------------------------------+   +---------
↓
Line : 6680 - Time: 00:00:00:500↓
              0         1         2         3         4         5         6                   0         1         2         3         4         5         6
      +----------------------------------------------------------------------+   +----------------------------------------------------------------------+   +---------
     0|   238.000   238.000   237.383   235.259   237.383   238.000   238.000|  0|   238.000   236.086 10230.000 10234.753 10230.000   236.086   238.000|  0|   237.38
     1|   238.000   236.086 10230.000 10234.753 10230.000   236.086   238.000|  1|   236.086 10237.259 10233.543 10233.716 10233.543 10237.259   236.086|  1| 10230.00
     2|   237.383 10230.000 10234.074 10231.840 10234.074 10230.000   237.383|  2| 10230.000 10233.543 10234.938 10233.519 10234.938 10233.543 10230.000|  2| 10234.07
     3|   235.259 10234.753 10231.840 10234.056 10231.840 10234.753   235.259|  3| 10234.753 10233.716 10233.519 10234.506 10233.519 10233.716 10234.753|  3| 10231.84
```

At the end of the simulation, we observe that all the elements are solid and have the same final temperature. Note that conductive heat transfer continues even after all cells are solid; this is in order to have a uniform temperature in the solution.



```
              0         1         2         3         4         5         6                     0         1         2         3         4         5         6
    +-----------------------------------------------------------------------+     +-----------------------------------------------------------------------+
   0| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    0| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   1| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    1| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   2| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    2| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   3| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    3| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   4| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    4| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   5| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    5| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   6| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    6| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
    +-----------------------------------------------------------------------+     +-----------------------------------------------------------------------+
↓
Line : 232239 - Time: 00:00:14:500↓
              0         1         2         3         4         5         6                     0         1         2         3         4         5         6
    +-----------------------------------------------------------------------+     +-----------------------------------------------------------------------+
   0| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    0| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   1| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    1| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   2| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    2| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   3| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    3| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   4| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    4| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   5| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    5| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
   6| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|    6| 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068 10233.068|
    +-----------------------------------------------------------------------+     +-----------------------------------------------------------------------+
```

d) DEVSView

We have run 3D animations using DEVSView. Only 7x7x7 were successfully loaded. Larger models could not be visualized using the tool. A video can be found here.

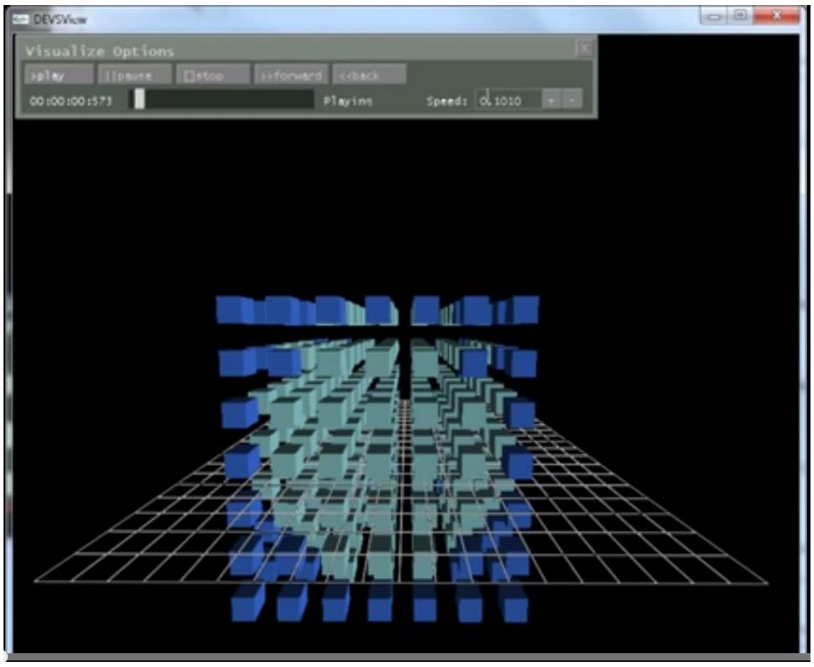


Figure 5. DEVSView Animation

We can easily observe dendritic growth using DEVSView and compare different patterns obtained for various initial temperatures. However, to obtain the same patterns shown in the reference for a 100x100x100 another visualization tool would be needed.

VII. Conclusion

A Cell-DEVS model that shows free dendritic growth in supercooled liquids was developed. Results have confirmed that the growth speedups with colder supercooling temperatures. In addition, various patterns can be observed depending on the initial temperature. We were not able to run 100x100x100 as in the reference paper due to CD++ and memory limitations.

For future work, convective heat can be modelled. Different substances/liquids, as in [10] where solute-driven dendrite growth is analyzed, could be simulated.

VIII. References

[1] Aviation Safety Network. ASN Aircraft Accident ATR-72-212. [Online]. http://aviation-safety.net/database/record.php?id=19941031-1

[2] (2012, Dec.) Lost - The Mystery Of Flight 447 [Air France Flight 447]. [Online]. https://www.youtube.com/watch?v=peHo0ajeTec

[3] Koshiro Monzen et al., "The use of a supercooling refrigerator improves the preservation of organ grafts," *Biochemical and biophysical research communications*, vol. 337, no. 2, pp. 534-539, 2005.

[4] Imperial College London. (2012, Mar.) Supercooling explained. [Online]. http://wwwf.imperial.ac.uk/blog/reporter/2012/03/12/supercooling-explained/

[5] A Lindsay Greer, "Materials science: A cloak of liquidity," *Nature*, vol. 464, pp. 1137-1138, 2010.

[6] SGR Brown and NB Bruce, "A 3-dimensional cellular automaton model of 'free'dendritic growth," *Scripta metallurgica et materialia*, vol. 32, no. 2, pp. 241-246, 1995.

[7] S-C Huang and ME Glicksman, "Overview 12: Fundamentals of dendritic solidification—I. Steady-state tip growth," *Acta Metallurgica*, vol. 29, no. 5, pp. 701-715, 1981.

[8] Planet Earth Online. (2010, Sep.) Supercooled water threat to aircraft fuel systems. [Online]. http://planetearth.nerc.ac.uk/news/story.aspx?id=813&cookieConsent=A

[9] Super Cooling Water. [Online]. https://www.youtube.com/watch?v=3p5zu4h0JKQ

[10] Mohsen Eshraghi, Sergio D Felicelli, and Bohumir Jelinek, "Three dimensional simulation of solutal dendrite growth using lattice Boltzmann and cellular automaton methods," *Journal of Crystal Growth*, vol. 354, no. 1, pp. 129-134, 2012.