Spatial Geometry in CompuCell3D

CellML Workshop 2009

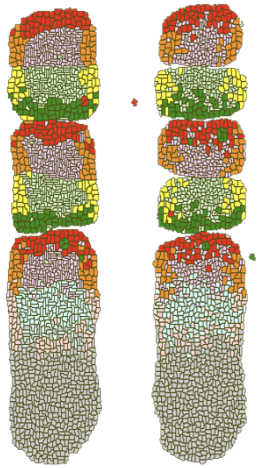
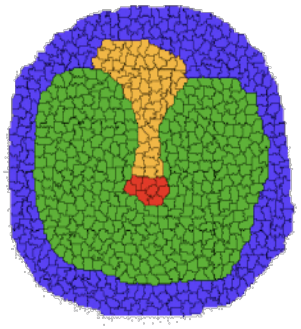
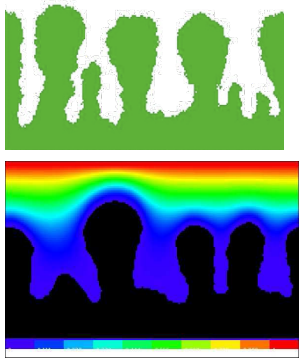
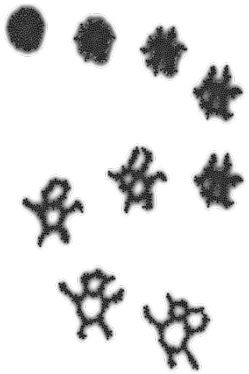
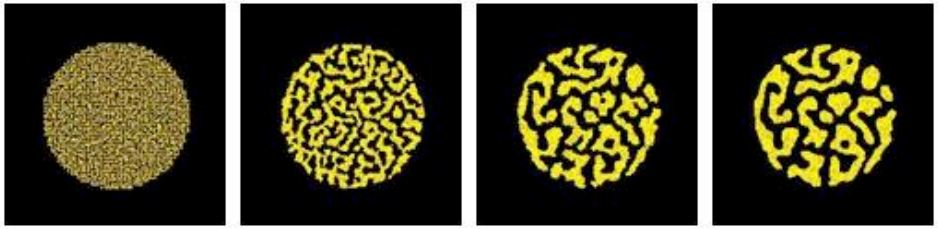
Waiheke, New Zealand

Benjamin Zaitlen

www.compucell3d.org

the biocomplexity institute

What is CompuCell3D?

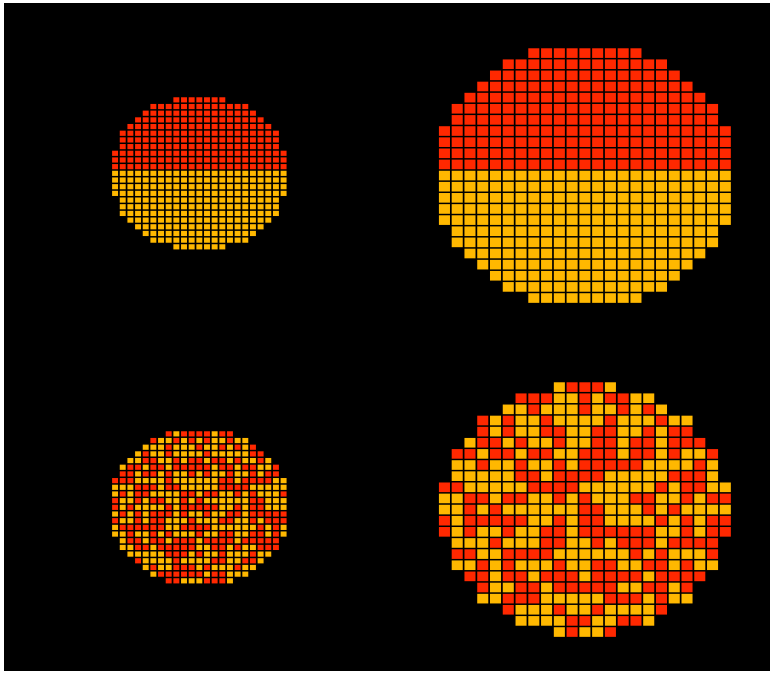






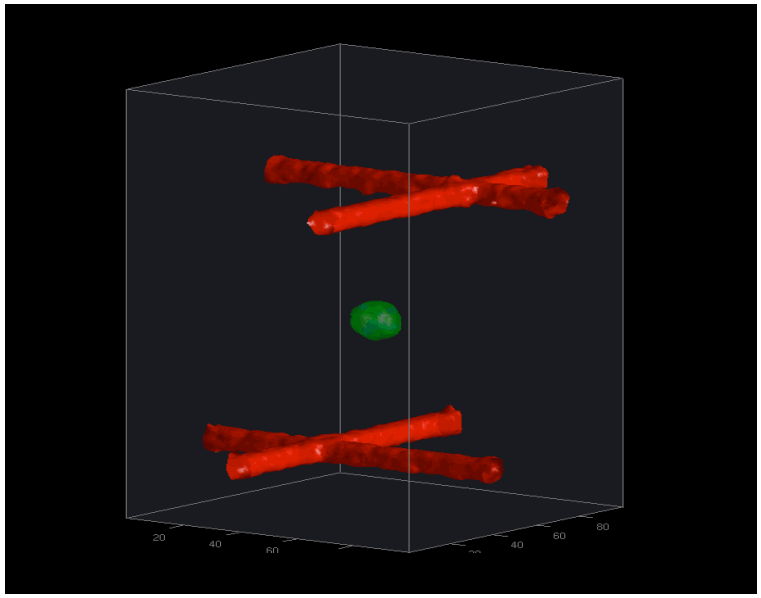
Multi-Scale Modeling Environment
Open Source
Available at:

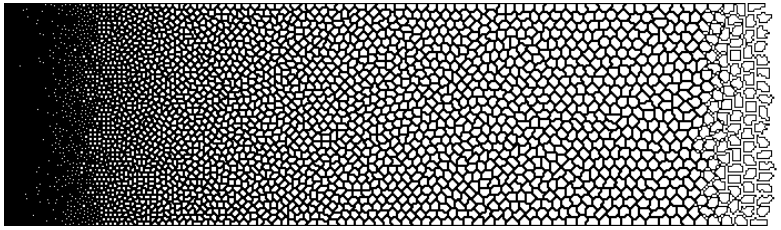
www.compucell3d.org

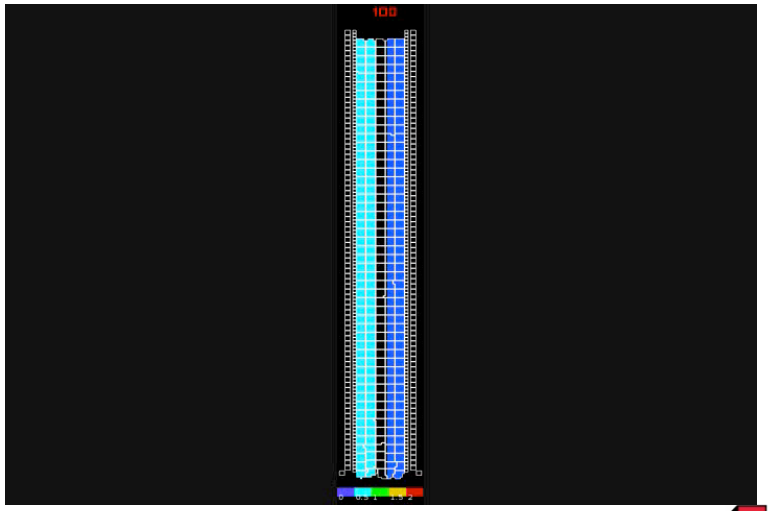
the biocomplexity institute

Demo Simulations



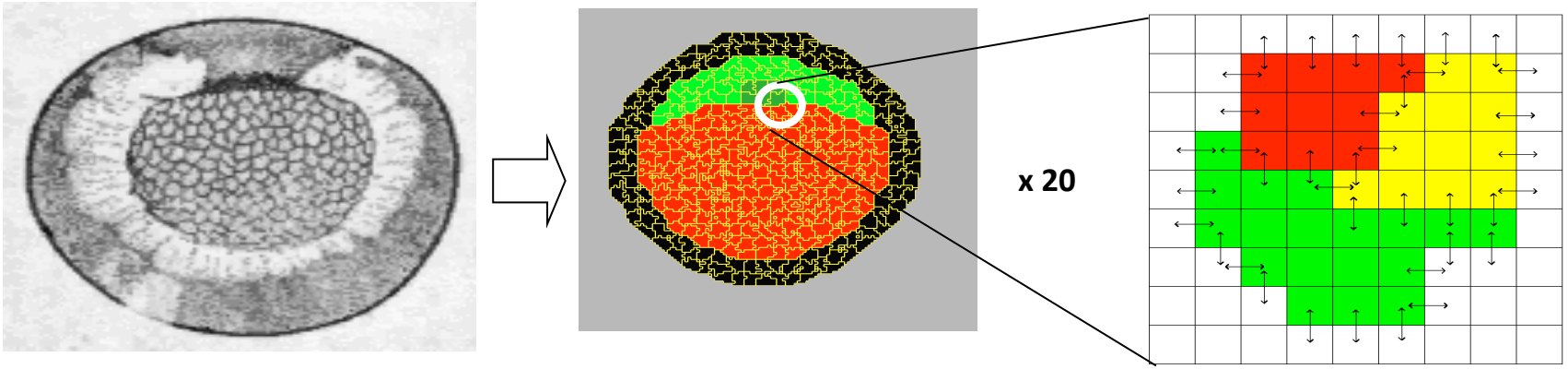






the biocomplexity institute

The GGH Model – an Overview



- Energy minimization formalism
  - extended by Graner and Glazier, 1992
- DAH: Contact energy depending on cell types (differentiated cells)

$$E = \sum_{x,x'} J_{\tau(\sigma(x)),\tau(\sigma(x'))}(1-\delta_{\sigma(x),\sigma(x')}) +$$

$$\lambda_s(s_\sigma - S_\sigma)^2 + \lambda_v(v_\sigma - V_\sigma)^2 +$$

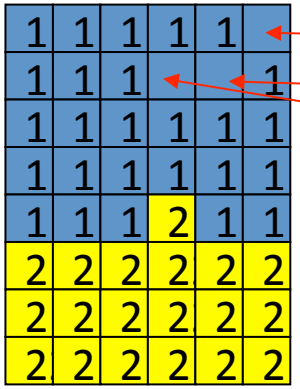
$$E_{chem} + E_{hapt} + \ldots$$

- Metropolis algorithm: probability of configuration change

$$P(\Delta E) = 1, \ \Delta E \leq 0$$

$$P(\Delta E) = e^{-\Delta E/kT}, \ \Delta E > 0$$

Brief Explanation of CompuCell cells



σ(**x**) –denotes id of the cell occupying position **x**. All pixels pointed by arrow have same cell id , thus they belong to the same cell

$\tau(\sigma($**x**$))$ denotes cell type of cell with id σ(**x**). In the picture above blue and yellow cells have different cell types and different cell id. Arrows mark different cell types

Notice that in your model you may (will) have many cells of the same type but with different id. For example in a simple cellsorting simulation there will be many cells of type “Condensing” and many cells with type “NonCondensinig”

Using PIFInitilizer

Use PIFInitializer to create sophisticated initial conditions. PIF file allows you to **compose cells from single pixels or from larger rectangular blocks**

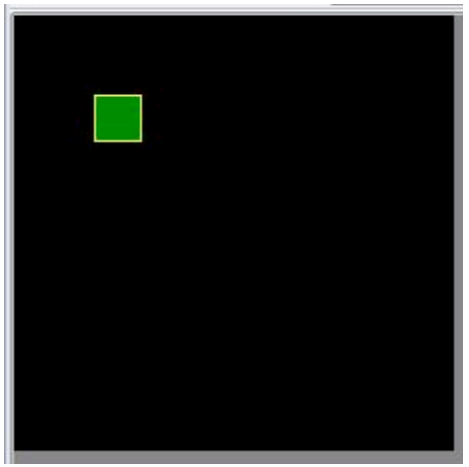
The syntax of the PIF file is given below:

Cell_id Cell_type x_low x_high y_low y_high z_low z_high

1 amoeba 10 15 10 15 0 0

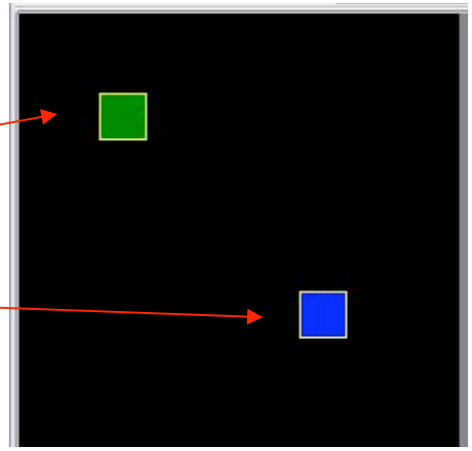
This will create rectangular cell with x-coordinates ranging from 10 to 15 (inclusive), y coordinates ranging from 10 to 15 (inclusive) and z coordinates ranging from 0 to 0 inclusive.

Origin *0,0*



Let's add another cell:
:

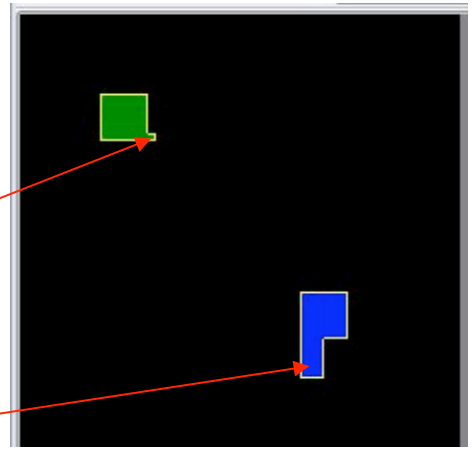
1 Amoeba 10 15 10 15 0 0
2 Bacteria 35 40 35 40 0 0



Notice that new cell has different cell_id (2) and different type (Bacterium)

Let's add pixels and blocks to the two cells from previous example:

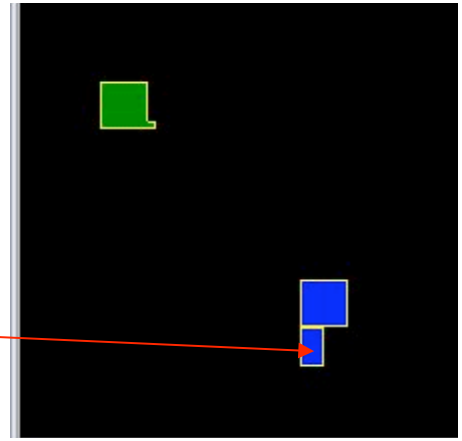
1 Amoeba 10 15 10 15 0 0
2 Bacteria 35 40 35 40 0 0
1 Amoeba 16 16 15 15 0 0
2 Bacteria 35 37 41 45 0 0



To add pixels, start new pif line with existing cell_id (1 or 2 here) and specify pixels.

This is what happens when you do not reuse cell_id

1 Amoeba 10 15 10 15 0 0
2 Bacteria 35 40 35 40 0 0
1 Amoeba 16 16 15 15 0 0
3 Bacteria 35 37 41 45 0 0



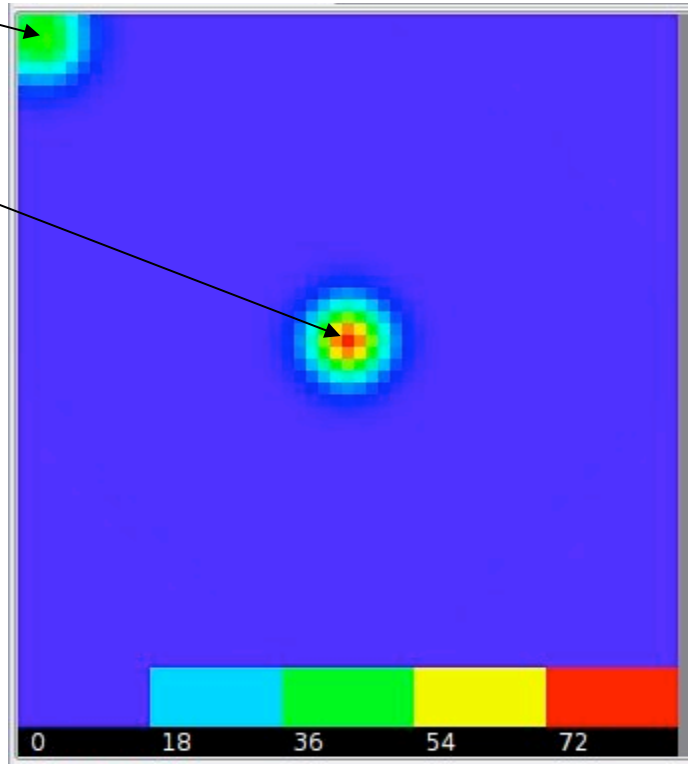
Introducing new cell_id (3) creates new cell.

PIF files allow users to specify arbitrarily complex cell shapes and cell arrangements. The drawback is that typing PIFs is a tedious task, and is not recommended. Typically PIF files are created using scripts.

In the future release of CompuCell3D users will be able to draw on the screen cells or regions filled with cells using GUI tools. Such graphical initialization tools will greatly simplify the process of setting up new simulations.

Two-pulse initial condition

5 5 0 1000.0
27 27 0 2000.0



Using SubCellular Compartments PIFInitilizer

CompuCell allows for **Compartments** of cell to be defined with the key word **Include Clusters**

The syntax of the PIF with Compartments is given below:

Cell_id Compartment_id Cell_type x_low x_high y_low y_high z_low z_high

Include Clusters

1 1 Center 50 54 50 54 10 14

1 2 Top 50 54 55 59 10 14

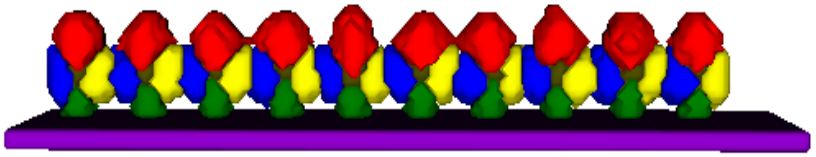
1 3 Bottom 50 54 45 49 10 14

1 4 Side 55 57 47 56 10 14

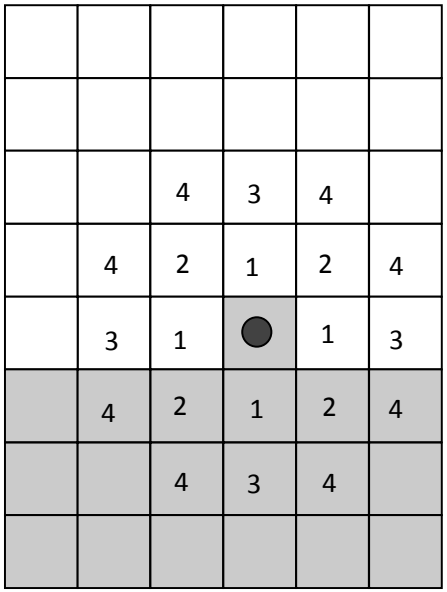
2 1 Center 38 42 50 54 10 14

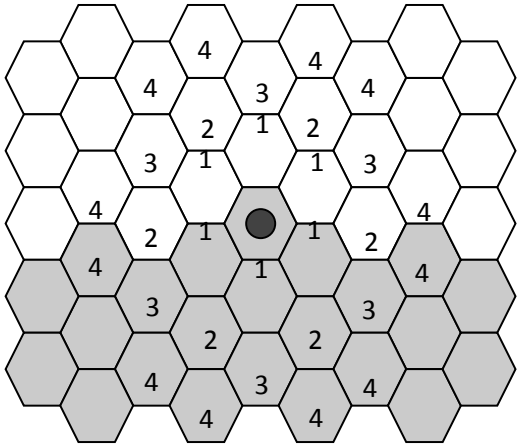
2 2 Top 38 42 55 59 10 14

….



Representations of Square and Hexagonal Lattice in 2D



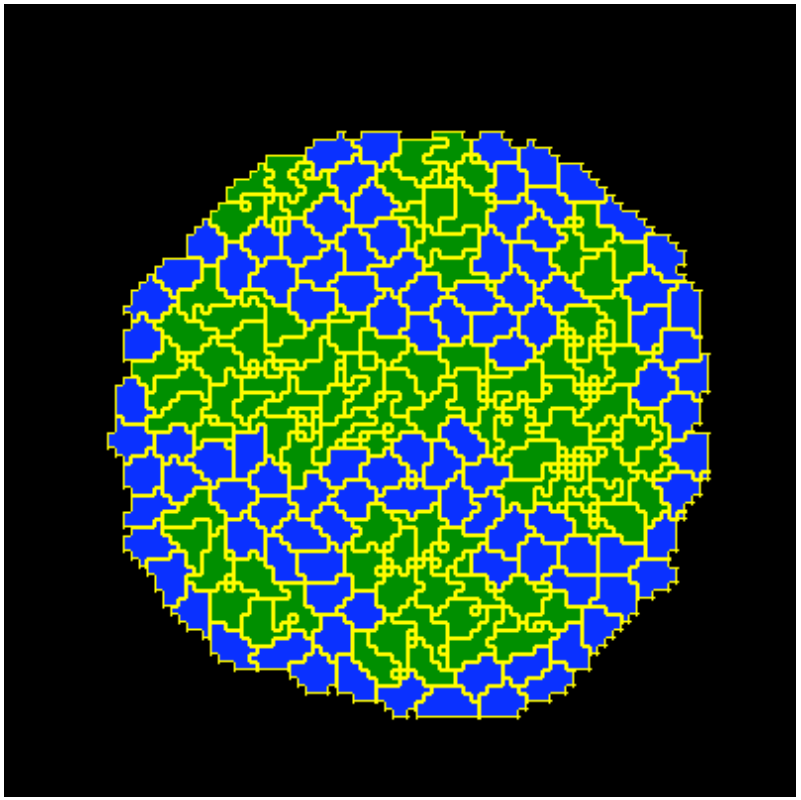


	2D Square Lattice		2D Hexagonal Lattice	
Neighbor Order	Number of Neighbors	Euclidian Distance	Number of Neighbors	Euclidian Distance
1	4	1	6	$\sqrt{2/\sqrt{3}}$
2	4	$\sqrt{2}$	6	$\sqrt{6/\sqrt{3}}$
3	4	2	6	$\sqrt{8/\sqrt{3}}$
4	8	$\sqrt{5}$	12	$\sqrt{14/\sqrt{3}}$

Nearest neighbors in 2D and their Euclidian distances from the central pixel

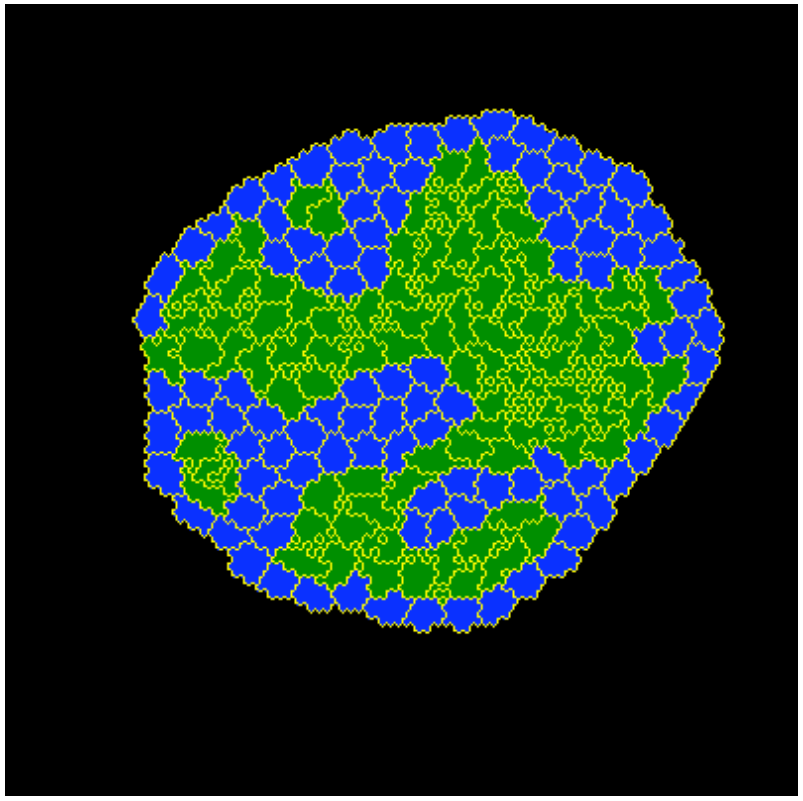
Hexagonal lattice reduces anisotropy

Square Lattice



1000 MCS

Hexagonal Lattice

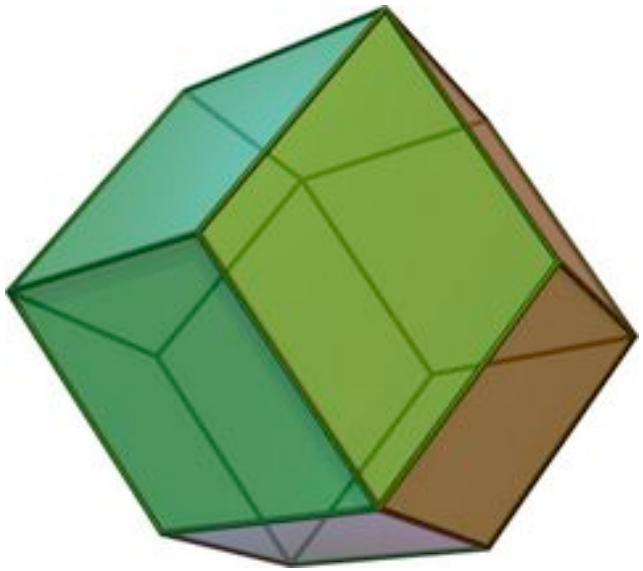


1000 MCS

the biocomplexity institute

3D Hexagonal Lattice Representation Rhombic Dodecahedron

Single Cell of Lattice



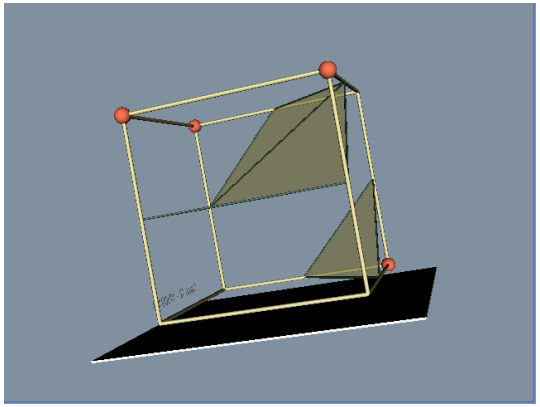
Lattice Representation

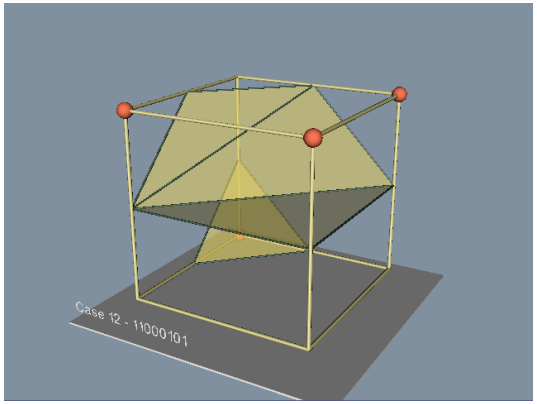
Visualization of 3D Cells in CompuCell

Currently use Qt for 2D Imaging and VTK Discrete Marching Cubes for 3D
Next generation player will use VTK for both 2D and 3D

Example of VTK Discrete Marching Cubes

Pick a cell and draw contour through it





Different File Formats?

If we are going to be using VTK for imaging maybe we should switch to .vti instead of .pif

Pros:

- Open source
- Widely used
- Smaller files for complex large simulations

Cons:

- Large Files for simple simulations
- Non Human Readable


```
<VTKFile type='ImageData' version='0.1' byte_order='LittleEndian'>
 <ImageData WholeExtent='0 99 0 99 0 159'
  Origin='0 0 0' Spacing='1 1 1'>
 <Piece Extent='0 99 0 99 0 159'>
  <PointData Scalars='field1'>
   <DataArray type='Int32' Name='field1' format='ascii'>
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
...
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 2 3 2 3 3 3 3 3 3 3 2
...
 </DataArray>
 </PointData>
 </Piece>
 </ImageData>
 </VTKFile>
```