Efficient Cellular Automata for 2D / 3D Free-Form Modeling

S. Druon
druon@lirmm.fr

A. Crosnier
crosnier@lirmm.fr

L. Brigandat
brigandat@lirmm.fr

Laboratoire d'Informatique, de Robotique et de Microelectronique de Montpellier (LIRMM)
161 Rue Ada, 34392 Montpellier Cedex 5, France

ABSTRACT

This paper presents an approach for efficiently simulating highly deformable 2D substances undergoing viscoplastic deformations in real time. The user deals with objects in the same way as clay works. Based on the work of Y. Takai and H. Arata, we suggest a new approach for the computation of repartition rules. We use a discrete 2D space in which each pixel is given a certain amount of clay at the intialisation of the system. The user's tool only moves some clay from a pixel to another, creating an overload. The repartition of this overload amoung the neighbours is then made according to the laws of plastic deformation thanks to a cellular automaton.

Keywords - Free-form shape modeling, virtual clay, Cellular automata, Margolus neighborhood

1 INTRODUCTION

On the first CAD systems, the modeling techniques were based on strict geometrical and mathematical operators. Designed for mechanical engineering, they require the user to have a very good knowledge of 3D geometry and a good spatial recognition. Furthermore, it is really difficult to model a natural object (animal, human face, etc.) from and with the geometric primitives proposed.

A very useful technique is image metamorphosis, or image "morphing". Given two or more objects, a smooth transition between them is created by generating intermediate objects. [Ler95] [Coh98] [Bei92] [Che95] [He94]. This very popular method needs the final image : it is thus impossible to apply it to the problem of shape modeling from scratch.

Another approach to this problem is the use of parametric patches or primitives to manipulate the surface of the object: Bezier Curves, NURBS, control points, disk Fields, etc. This kind of method is both efficient and applicable: [Wyv97], [Wel92], [Hsu92], [Che96]. The main limit of this technique is that the user cannot deform the objects in an intuitive way.

Journal of WSCG, vol 11, No.1, ISSN 1213-6972
WSCG'2003, February 3-7 2003, Plzen, Czech Republic.

Because those systems are inadapted to artistic creation, Free-form shape modeling is now a key-technique in modern computer graphics. The object is considered as clay which shows well-known deformations. It is then much more simple for the user to interact with the object. In that case, the modelling task is carried out in a natural and user-friendly way, exactly like during clayworks.

Many research works have been carried out on simulation of such deformations using physical laws: Finite Element method [Pen89] [Wie97] [Deb00], methods based on elasticity theory [Deb00] [Ter88] [Don01], particle systems [Ree83] [Des96], etc. So far, none of these methods allows real time interaction with the user : they require much too long computation time for complex shapes and are not suitable for human interaction.

Some researches are undergoing on volume sculpting in a 3D virtual space. The idea is to use a discretisation of the space in voxel (or pixel in the 2D case). Each voxel can be full or empty : the user starts from a basic shape and then removes or adds matter where he wants [Aya01]. Such an approach, combined with a cellular automaton, has been developed by Y. Takai and H. Arata [Ara99]. In this approach, a 3D cellular automaton is used to simulate plastic deformations of clay. To each voxel is allocated a finite state automaton which is given the distribution rules of the virtual clay. Each automaton repeats state transitions according to

the state of their neighbours.

This paper presents an extension of the work of Y. Takai and H. Arata. Several new repartition algorithms have been developed and tested in order to get more realistic results and to evaluate the efficiency and the speed of the method. A parallel implementation of the automaton has also been tested in the 2D case and will be presented here.

2 THE VIRTUAL CLAY MODEL

This section is a very short presentation of the work suggested by H. Arata and al. in [Ara99]. In Arata's method, the deformation is considered as a physical process of equalizing density distribution of the virtual clay in a 2D space. When the density of virtual clay is under a certain threshold everywhere, the virtual clay objects keeps its own shape. The deformation of the object is caused by clay transportation from high density portions to low density portions. In this approach, each pixel is given an integer value corresponding to the mass of clay included in the pixel. A threshold is defined in the pixel space : every pixel over this threshold will try to give some clay to its neighbours. The process is implemented by a cellular automaton using 2D Margolus Neighborhood.

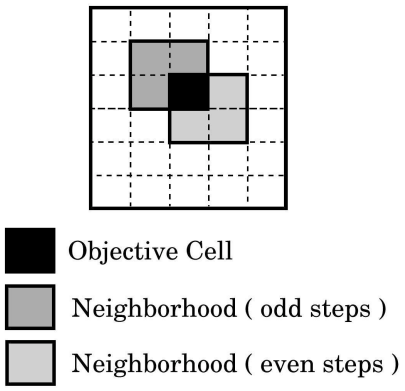

Figure 1 : Margolus neighborhood

Figure 1 shows an example of Margolus neighborhood. In this neighborhood, the nearest 4 cells make one block: the neighborhood can be seen as the block including the cell itself. The transitions of all the cells belonging to a same block will be performed in one step of the algorithm. In addition, the boundaries of the blocks are changed at each step, as shown in fig 1. During even steps, a cell will not belong to the same block as during odd steps : a cell alternates its neighborhood in every transition step.

Figure 2 illustrates the transition rules in the 2D case. Each cell has a binary state depending on whether it is above or under the threshold and a certain amount of clay m_k. A block is composed of four cells, over or under the threshold. Hence, four different block patterns can occurr.

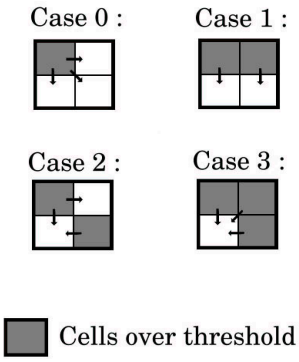

Figure 2 : Block patterns and state transition rules

The repartition of the excess of clay dm_k will be performed with the following algorithm :

For each block
  For each cell k over threshold
    $dm_k \leftarrow m_k * \alpha$
    $m_k \leftarrow m_k - dm_k$
  For each cell j under threshold
    $m_j \leftarrow m_j + (dm_1 + dm_2 + ... + dm_r)/n$

where α is a rate constant for distribution ($0 < \alpha < 1$), r the number of cells over threshold and n the number of cells under threshold. It is important to note that this model can deal with physical obstacles, these obstacles being cells to which clay cannot be distributed. In this model, all plastic deformation is based on a push operation : the clay is transferred from a cell into the adjacent cell along the normal direction of the tool.

This model allows real-time interaction with the user in very good conditions, but because the repartition rules are not physically-based, the behaviour of the virtual clay can sometimes be very surprising and not intuitive. Our contribution to this work will focus on enhancing the realism of the deformation taking into account the time constraint so that real-time interaction remains possible.

3 OUR APPROACH

Even though physics based modeling does not seem to be possible in real time, it is possible to build new

repartition rules that provide results closer to the realistic behavior. The macroscopic behavior of the object obtained with the method described in [Ara99] is not always realistic. The following pictures show a cylinder of clay pressed by a plate submitted to a vertical displacement and horizontally maintained :

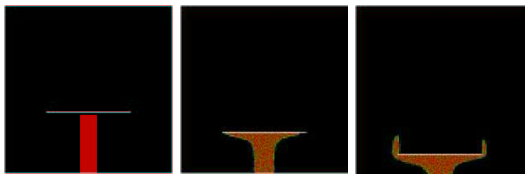

Figure 3.1: An exemple of unexpected macroscopic behaviour of the clay

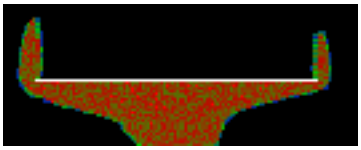

Figure 3.2: The density in the object is not homogeneous

In figure 3.2, the colors of the clay are proportionnal to the amount of virtual clay in a pixel. As we can see, the distribution of density in the object is not homogeneous. To avoid such situation, we introduce three different new repartition rules.

First Repartition Rule

This problem can be solved thanks to a better understanding of the physics involved in the process. One of the models provided studied in rheology for such deformations is the Bingham's fluid. Bingham's fluids are perfect visco-plastic bodies. They act like a newtonian fluid when the constraint is over a certain threshold τ_c. If τ denotes the constraint applied to the object, and ε is the deformation, the behaviour of a Bingham's fluid is described by the following state equation :

$$\begin{cases} \tau < \tau_c & \dot{\varepsilon} = 0 \\ \tau > \tau_c & \tau = \tau_c + \alpha\dot{\varepsilon} \end{cases} \quad (1)$$

From this model comes a new idea for the repartition law. In the previous method, a cell over the threshold gives a certain amount of its clay to its neighbours, without any consideration for the threshold. From this comes the heterogeneity of the density in the object.

Our first new repartition law is inspired from Bingham's model. We keep the idea of a threshold, but this threshold is considered as a local stable state for the cell. The quantity of clay a cell can give away is determined by this threshold: left alone, a cell will tend to go back to the nearest stable state. The new repartition law is the following:

For each block
  For each cell k over threshold T_c
    $dm_k \leftarrow m_k - T_c$
    $m_k \leftarrow m_k - dm_k$
  For each cell j under threshold T_c
    $m_j \leftarrow m_j + (dm_1 + dm_2 + ... + dm_r)/n$

The value of the threshold T_c and the initial quantity of matter in the cells will determinate the behavior of our virtual clay. For great values of the threshold, the virtual clay will be very compressible and will not flow easily. For values near zero, the clay is nearly incompressible and behaves like a newtonian fluid.

Because this algorithm was implemented using integers, we have to take into account the error introduced by the division (the total quantity of clay must be conserved through the deformation). This error is added randomly to one of the neighbours of the cell.

As you can see in figure 4, the visual aspect of the deformation is good and the density is homogeneous. Furthermore, the computation time is about the same as in [Ara99] (cf figure 6 for a comparison).

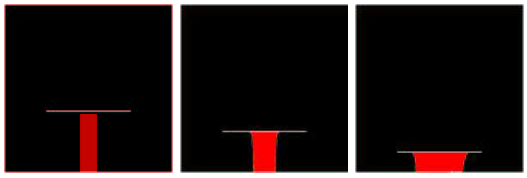

Figure 4.1: Results obtained with repartition law 1

Figure 4.2: The density in the object is now homogeneous

Second Repartition Rule

Even though the results of method 1 are correct, there is something wrong in the way the clay is allocated to the neighbours. If we have a look on the way this excess clay is reparted, we can observe that all the accepting cells will have to accept the same quantity, no matter how much clay they already contain.

This second repartition law introduces a new concept in our algorithm : the concept of absorbance. The absorbance A is the amount of clay necessary to reach the state of maximum compressibility in accepting cells.

For each block
  For each cell k over threshold T_c
    $E \leftarrow E + (m_k - T_c)$
  For each cell j under threshold T_c
    $A \leftarrow A + (T_c - m_j)$

For each block, E is the excess clay in the cells over the threshold, A is the room available in the cells under threshold.

- if $E < A$, the absorbing cells are sorted and we begin by filling the cells with the less clay inside.
  For each absorbing cell j (sorted by quantity of clay)
    $temp \leftarrow min(T_c - m_j, E)$
    $m_j \leftarrow m_j + temp$
    $E \leftarrow E - temp$
- if $E > A$, the clay is equally reparted to the cells :
  For each absorbing cell j, $j\varepsilon\{1 \ldots n\}$:
    $m_j \leftarrow T_c + \frac{E - A}{n}$

The results obtained with this second repartition rule are visually correct, but the computing time is now very long. As shown in figure 6, the evolution of the computing time of each state is not constant with this method and prevents from using it in real time applications. It is obvious that the sort of the absorbing cells is a cpu-consuming operation, this is why we built this third and last repartition rule.

Third Repartition Rule

The principle of the method remains the same : for each block, we determinate the quantity of clay E in excess and the total absorbance A. But instead of trying to sort the absorbing cells, the repartition of E will be made at the pro-rata of absorbance :

For each block
  For each cell k over threshold T_c
    $E \leftarrow E + (m_k - T_c)$
    $m_k \leftarrow T_c$
  For each cell j under threshold T_c
    $A \leftarrow A + (T_c - m_j)$
  For each cell j under threshold T_c
    $m_j \leftarrow m_j + E\frac{T_c - m_j}{A}$

This last method presents the avantages of the two previous ones. The deformation is closer to the real one, and the computing time is reduced to its minimum because all the calculation is performed in a simple affectation operation. The result of the implementation is illustrated in figure 5.

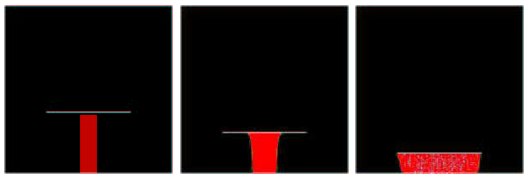

Figure 5: Results with repartition law 3

Comparison of the Methods

There are two aspects in the evalutation of a method: the final aspect of the object and the time elapsed before the stabilisation of the automaton. Figure 6 is a comparison of the stabilisation time for the four algorithms in the example of the cylinder pressed by an horizontal plate. The tests were performed on a dell workstation "Precision 530" (bi-processors Xeon 1.9 Ghz, 1 Gig Ram) running Linux Redhat 7.3 .

Method 2, as described in the previous lines, is not converging : the computation time does not seem to be majorated and thus is not usable for real time applications. Method 1 and 3, are of course longer than the method described by Arata and al. [Ara99] but this is due to the fact that we are looking for more realism in the deformation and need more computing. Anyway, the computing time does not exceed 500 ms without any optimisation in the algorithm nor parallelisation : the conditions for real-time usability are reached.

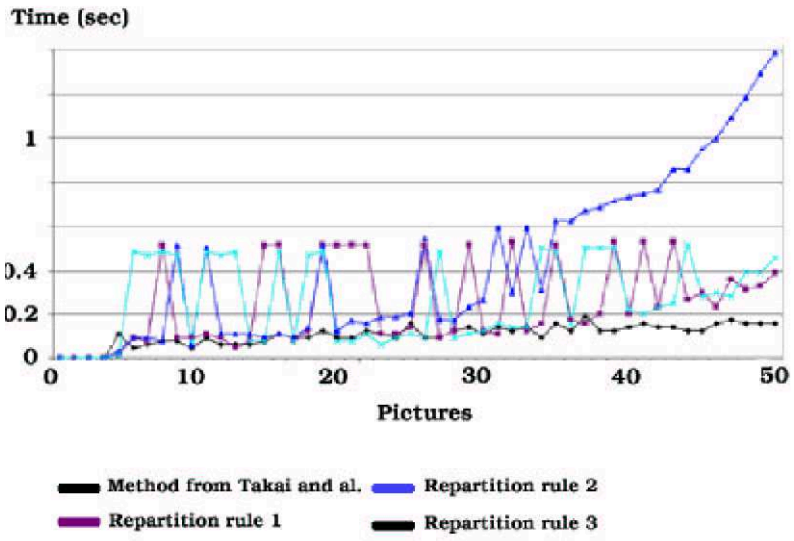

Figure 6: Time comparison

4 PARALLELISATION OF THE ALGORITHM

The problem of working with voxels is the computational complexity. For a voxel space of size n, n^2 operations will be performed for the 2D case and n^3 for the 3D case. Furthermore, the greater n is, and the more precise is the model. This is the reason why cellular automata are interesting : one of their advantage is the fact that it is a highly parallel structure and as a consequence, they allow to work with greater values of n.

One of the methods available for using cellular automata systems in optimal conditions is the use of Posix threads for maximum efficiency. In this article, we will only describe the 2 threads case, but the method can be easily extended to n threads.

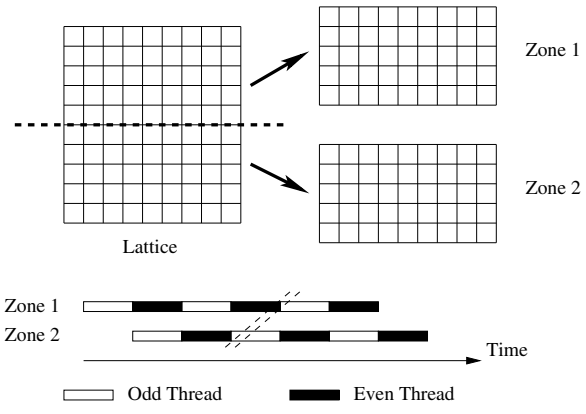

Figure 7 : Two threads working on the same lattice

If we take a closer look to the algorithm, we can observe that it is not necessary for a step to be finished before beginning the next one. Even and odd steps can be performed by different threads as long as we make sure that they will not interact on the same cells concurrently and as long as they remain in the same order.

In order to make sure that these conditions are respected, we have cut our working grid in two zones. Each one is seen as an unsharable ressource. Figure 7 illustrates the use of the lattice by two threads :

Two threads are used, one for the odd steps and one for the even steps of the algorithm. The synchronization between the two threads is made thanks to a simple petri network, described in figure 8.

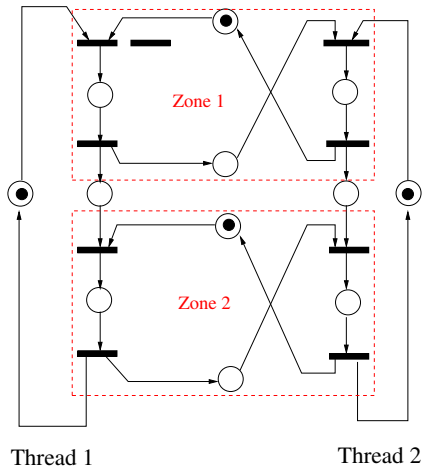

Figure 8 : Petri Network used for threads synchronisation

Thanks to this use of the Posix threads, the computation time before stabilisation is nearly divided by two. An implementation on a workstation using shared cache memory should allow us to make this time even shorter.

5 EXTENSION TO THE 3D CASE

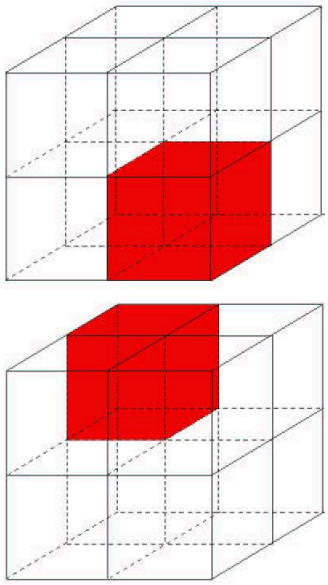

Figure 9 : 3D Margolus neighborhood (odd and even steps)

The repartition rules presented in the previous pages are general and can be applied to the 3D case without any change. The margolus neighborhood, of course, was adapted and is represented in figure 9.

The parallalelisation of the algorithm in the 3D case is also based on what we presented in the previous sections. For p threads, the voxel space is cut into p horizontal layers.

6 RESULTS AND PERSPECTIVES

As we saw in the previous sections, it is possible to use cellular automata for interactive and realistic simulations of deformations. The method we have presented in this paper allows real time interaction and a natural behaviour of the clay in the 2D case. The different repartition rules and the parallelisation of the algorithm can be extended to the 3D case with no real difficulty. The following pictures show some examples of deformations in a 3D voxel space. This work is a first attempt to make a realistic modelisation of viscoplastic deformations in real time. The aim of our research work is now on enhancing the realism of the deformation and the interactivity between the user and the 3D system.

References

[Ara99] H. ARATA, Y. TAKAI, N. K. TAKAI, and T. YAMAMOTO. Free-form shape modeling by 3D cellular automata. In *International Conference on Shape Modeling and Applications*, pages 242–247, 1999.

[Aya01] J. AYASSE and H. MULLER. Interactive manipulation of voxel with free-formed voxel tools. In *Vision, Modelling and Visualization (VMV01)*, 2001.

[Bei92] T. BEIER and S. NEELY. Feature-based image metamorphosis. *ACM/SIGGRAPH Computer Graphics*, 26(2):35–42, 1992.

[Che95] M. CHEN, M.W. JONES, and P. TOWNSEND. Methods for volume metamorphosis. Image Processing for Broadcast and Video Production, 1995.

[Che96] M. CHEN, W. JONES, and P. TOWNSEND. Volume distortion and morphing using disk fields. *Computers and Graphics*, 20(4):567–575, 1996.

[Coh98] D. COHEN-OR, D. LEVIN, and A. SOLOMOVICI. Three-dimensional distance field metamorphosis. *ACM Transactions on Graphics*, 17(2):116–141, 1998.

[Deb00] G. DEBUNNE. *Animation multirésolution d'objets déformables en temps réel*. PhD thesis, INPG, 2000.

[Des96] M. DESBRUN and M-P GASCUEL. Smoothed particles : A new paradigm for animating highly deformable bodies. In *6th Eurographics Workshop on animation and Simulation '96*, pages 61–76, 1996.

[Don01] K. T. MCDONNELL, Hong QIN, and R. A. WLODARCZYK. Virtual clay: a real-time sculpting system with haptic toolkits. In *Symposium on Interactive 3D Graphics*, pages 179–190, 2001.

[He94] T. HE, S. WANG, and A. KAUFMAN. Wavelet-based volume morphing. In *IEEE Visualization'94*, pages 85–92, Washington DC, 1994.

[Hsu92] W. M. HSU, J.F. HUGHES, and H. KAUFMAN. Direct manipulation of free form deformation. In *Computer Graphics*, pages 177–184, 1992.

[Ler95] A. LERIOS, C. GARFINKLE, and M. LEVOY. Feature-based volume metamorphosis. In *SIGGRAPH'95*, pages 496–456, Los Angeles, August 1995.

[Pen89] A. PENTLAND and J. WILLIAMS. Good vibrations: Modal dynamics for graphics and animation. In *Computer Graphics*, pages 215–222, 1989.

[Ree83] W. T. REEVES. Particle systems – a technique for modeling a class of fuzzy objects. In *Computer Graphics*, pages 359–376, 1983.

[Ter88] D. TERZOPOULOS and K. FLEISHER. Deformable models. In *The Visual Computer*, pages 306–331, 1988.

[Wel92] W. WELCH and A. WITKIN. Variational surface modeling. In *Computer Graphics*, pages 157–166, 1992.

[Wie97] C. WIENERS. The implementation of adaptive multigrid methods for finite elements. Technical report, Universitat Stuttgart, SFB 404 Preprint 97/12, 1997.

[Wyv97] G. WYVILL, D. MCROBIE, and M. GIGANTE. Modeling with features. In *IEEE Computer graphics and applications*, pages 40–46, 1997.

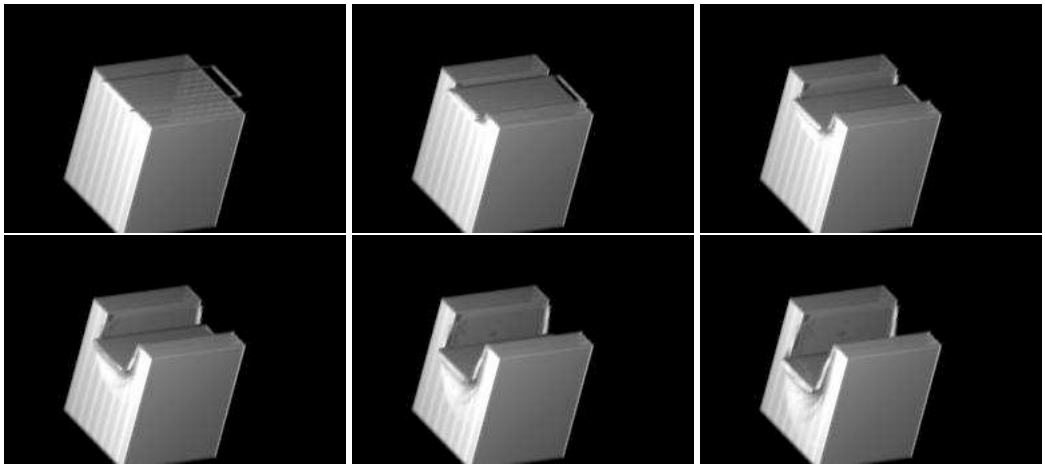

Figure 10 : Cube pressed by a small plate (repartition rule 3)

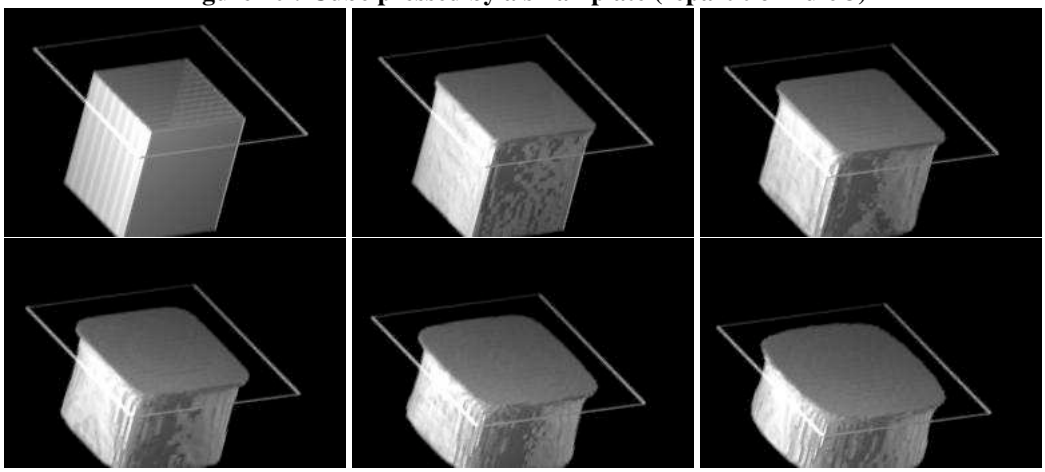

Figure 11 : Cube pressed by a big plate (repartition rule 3)

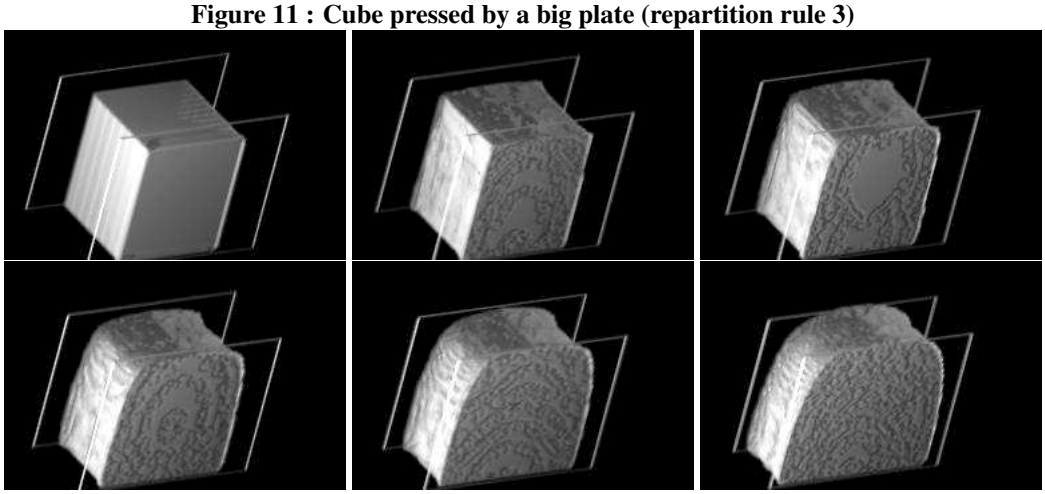

Figure 12 : Cube pressed by two big plates (repartition rule 3)