Open Environment for 2d Lattice-Grain CA

Guillaume Cottenceau* and Dominique Désérable

INSA – Institut National des Sciences Appliquées,
Laboratoire de Génie Civil & Génie Mécanique,
20 Avenue des Buttes de Coësmes, 35043 Rennes, France
gcottenc@gmail.com, deserable@insa-rennes.fr

Abstract. An open cellular automata (CA) environment applied to the simulation of two-dimensional granular flows is presented herein. The CA belongs to the family of so-called "lattice-grain" (cellular) automata (LGrA) with one particle per cell. The time evolution is governed by a "request-exchange" synchronous mode which simulates a two-stage interaction-advection process. The transition rule follows a simple logic including three physical components: an external field, a set of kinematical exclusion rules and an inertial effect. After a short presentation of the CA logic, this paper describes the open user interface structured onto and emphasizes the versatility of the model.

Keywords: cellular automata, lattice-grain CA (LGrA), granular flow, user-friendly interface, open source.

1 Introduction

A thorough investigation into the behavior of granular matter is of major importance for scientific and industrial applications. The theoretical methods currently used to tackle these problems split up into the three distinct levels of: continuum models, particle dynamics and cellular automata. Cellular automata may capture the essence of physical phenomena resulting from elementary factors and make a suitable and powerful tool to catch the influence of the microscopic scale onto the macroscopic behavior of complex systems. Moreover, an important feature of cellular automata is their capability of handling complicated geometries and boundary conditions, where classical computational methods may fail or involve extra difficulty to model. Characterized by a high number of cells and a low transport of information, cellular automata can simulate a diversity of complex processes as encountered in dynamical systems [1,2]. Known as "lattice-gas" (cellular) automata (LGA) in hydrodynamics, they are an extreme simplification of molecular dynamics and have been widely developed over the last years for a better understanding of turbulence and stability phenomena in fluids, before being applied to the study of a larger diversity of physical systems [3,4,5,6,7,8]. Concerning granular media, there was a number of attempts which in turn make a relative simplification of granular dynamics and which are often

* INSA – Computer Science Department, until 2000.

known as “lattice-grain” (cellular) automata (LGrA); they have yielded some interesting results especially for hopper flows, flows around obstacles, segregation or stratification phenomena during free surface multiphase flows or the formation of density waves in channel flows [9,10,11,12,13,14]. A state of the art for lattice-grain models is given in [15] with references therein.

We focus on our LGrA described in detail in [16] and on the open environment associated with. This software is recognizable under the name “Grany–3” as a freeware [17,18]. Our model, derived from the analytical model of Litwiniszyn-Müllins [19,20], is purely “kinematical”, i.e. it ignores force and acceleration and only takes into account the notion of space occupancy. The transition rule is provided with three original features. Firstly, a two-stage “request-exchange” synchronous logic which simulates a physical interaction-advection process and activates all cells while ensuring a uniform operating mode; secondly, an “exclusion” principle based upon an elementary logic allowing the user to diversify the behavior of the “solid” phase; thirdly a synchronous “propagative” mode which simulates the propagation of the “void” phase: for brevity’s sake, this mode is not tackled in this paper. Beyond its general contribution to the simulation of multiphase granular flows, our model offers a high versatility allowing the user to simulate a diversity of systems by merely adjusting the initial and boundary conditions.

This paper aims to emphasize its versatility and to display the user-friendly interface of the free software associated with. Section 2 describes our LGrA logic, illustrated in our software presentation in Section 3 with a case study. We conclude in Section 4 by emphasizing the advantages that the researcher involved in studies on granular matter could benefit by using, or at least exploring what could bring him this open source software.

2 LGrA Logic

2.1 Topology of the Network and Space Occupancy

The automaton is constructed on the 6-*valent* grid. As for the “FHP” lattice-gas [3], this $2d$ topology offers the greatest number of symmetries for a regular network: herein it maximizes the number of *degrees of freedom* (or directions) for a displacement as well as the upper bound of the *coordination number* (the number of contacts of a particle with its vicinity). The concise notation “$n$df” ($0 \leq n \leq 6$) will be used for a law with n degrees of freedom. Owing to the site-cell duality, a hexavalent site is the center of a hexagonal cell and every site is connected to its six neighbors. We denote by N_s the “size” or number of sites of the network. Since the graph is regular with degree 6, the number of links connecting a pair of adjacent sites is clearly $3N_s$.

The *space occupancy* principle allows one and only one particle per site, whether it is a solid, liquid or gaseous one, the term “particle” being a purely formal denotation. Multiphase flows are considered, where a phase ϕ_i, indiced in the set $N_\phi = \{1, 2, \ldots, n_\phi\}$ for a system of n_ϕ phases, denotes a set of particles provided with identical properties. Note that a “phase” refers either to different states of the same material or to different kinds of materials. A working feature of the automaton is that any cell of whatever content performs the same action synchronously, that leads to a uniform operating mode over the whole network.

2.2 A Two-Stage Interaction-Advection

The interaction-advection process is performed by a two-stage transition according to an original "request-exchange" mechanism. We will prefer this two-stage term to the "propagation-collision" one often used in the lattice-gas terminology in order to avoid confusion with a long-range propagative interaction. In the *request* stage, each cell autonomously performs a computation composed of a precalculation followed by a random choice. The result is a *potential* direction of displacement which becomes the direction of request. In the *exchange* stage, a test is performed for each link of the network in order to detect whether an agreement has been reached between the potential directions yielded by both adjacent sites.

Request-Exchange Synchronous Logic. It is useful to define a "request" graph as well as an "exchange" graph on the hexavalent lattice. The cell performs a request for displacement in one direction *at most.* A request is represented by an oriented arc, say $x \rightarrow x'$, from one site x to another site x' (Fig. 1a). Whenever two arcs $x \rightarrow x'$ and $x' \rightarrow x$ coexist, they are replaced by the edge $x \leftrightarrow x'$ in the exchange graph (Fig. 1b). The exchange graph is clearly a non-connected graph of isolated edges and where each component stands for a binary reaction. (The fact that each cell performs one request *at most* ensures thereby a conflict-free process. So whenever two particles compete with one another to fill a common site, arbitration is handled by the particle filling the addressed target.)

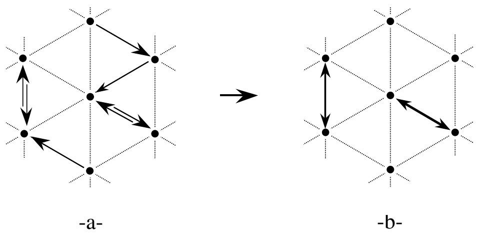


Fig. 1. The two-stage interaction-advection: (a) request graph and (b) exchange graph

Scheduling the Particle-Particle Exchanges. Requests are independently performed on each of the N_s sites whereas the particle-particle exchanges must be reviewed on each of the $3N_s$ links. In order to achieve the exchange stage without gap nor overlap a consistent scheduling is set up according to the, say, isotropic orientation $N \rightarrow S$, $SW \rightarrow NE$, $SE \rightarrow NW$ in Fig. 2a. This convention yields a N–SW–SE input pattern as well as a S–NE–NW output pattern uniformly distributed on any site. In practice, the cell sends a copy of its own request to its *S–NE–NW* neighbors while receiving a copy of the requests from its *N–SW–SE* neighbors (Fig 2b). Then the occurrence of an exchange through the *N–SW–SE* pattern is analysed. Finally the exchange with the *N* or *SW* or *SE* neighbor (if any) is performed.

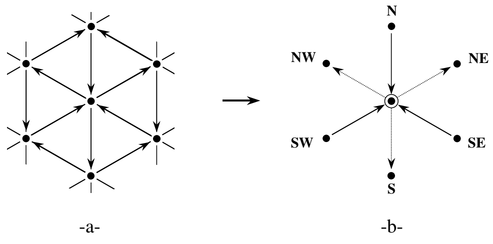


Fig. 2. (a) Orientation of the network and (b) exchange control

2.3 Phase Components and Interaction Law

The behavior of a *phase* in a multiphase system will be defined by three "physical" components: an external field, a set of exclusion rules and an inertial effect. It should be pointed out that it is not so much the autonomous behavior law of a given phase that must be taken into account but the *interaction* law with its local neighborhood. This fact is illustrated hereafter. In the sequel, a set $K = (0, 1, 2, 3, 4, 5)$ will be assigned to the six directions NE, N, NW, SW, S, SE.

Modeling the External Field. The action of an external field (in general the gravity) is modeled by assigning a 6-fold vector W to each phase. For simplicity, let us consider an isolated particle p_1 of phase ϕ_1, immersed in a fluid of phase ϕ_2 and where $\rho_1 > \rho_2$ are the respective densities. To each phase ϕ_i is assigned a distribution of weights $W_i = (w_i^{(k)})_{k \in K}$ with non-negative integers, according to the orientations in Fig. 2. We adopt a "3df–3df" law with $W_1 = (0, 0, 0, 1, 1, 1)$, $W_2 = (1, 1, 1, 0, 0, 0)$ where the pattern *SW–S–SE* means "downwards". One request of ϕ_1 downwards will then result from W_1 and a swap may occur with a fluid particle below. Assume now p_1 immersed in a new fluid of phase ϕ'_2 with density $\rho'_2 > \rho_1$. Moving p_1 upwards requires *now* a distribution of the form $W_1 = (1, 1, 1, 0, 0, 0)$, $W'_2 = (0, 0, 0, 1, 1, 1)$.

From this observation, it follows that the external field is represented neither by W_1 nor by W_2 but by the set $\{W_1, W_2\}$. In a system of n_ϕ phases, the external field will be represented by a set $\{W_1, W_2, \ldots, W_{n_\phi}\}$, where W_i is the distribution related to phase ϕ_i.

Exclusion Rules and Modes. The role of *exclusion rules* is firstly to ensure "kinematical compatibility" of the model. For instance, a usual rule consists in prohibiting the exchange of two solid particles lying in adjacent cells through their common link. More generally, their role is to differentiate the laws governing the interactions between neighboring particles. They thus help to model some morphological (size, particle geometry...) or mechanical fuzzy notion (roughness, stiffness...) by introducing a binary parameter according to which a particle could either appear as "rough" or "smooth" in a monodisperse medium or "large" or "small" in a polydisperse medium... other physical scenarios are possible.

An *exclusion mode* is assigned to an exclusion rule and defined as follows.

– *pre-exclusion*: the rule is applied from the *request* stage onwards; to select a displacement direction, the cell is aware of the state of its six neighbors.
– *post-exclusion*: the application of the rule is postponed until just before the *exchange* stage; the cell performs a "blind" request with no consideration of the local vicinity.

As a matter of fact, the *pre* (resp. *post*) mode aims to affect a less (resp. more) frictional –or viscous– behavior to the phase which is assigned to. We define a set of three exclusion rules R_1, R_2, R_3, namely "frontal", "interstitial", "constrictive", more and more *coercive* when taken in this order (Fig. 3). Physically, assigning R_3 to phase ϕ implies R_2 which in turn implies R_1 to it. This sequence is shortly denoted by the implication $R_3(\phi) \Rightarrow R_2(\phi) \Rightarrow R_1(\phi)$. For consistency, a "neutral" rule R_0 will denote no exclusion for the assigned phase. The implementation of this exclusion logic is quite simple.

Inertial Effect. Modeling an "*inertial effect*" consists in saving the *memory* of the particle's displacement direction at the previous timestep, to reintroduce it with a user-defined weight into the new weighted distribution for the current timestep. An *inertial* coefficient c_i, which takes on a positive integer value, is assigned to each phase ϕ_i and $c_i = 1$ means no inertial effect for this phase.

It may be asked what physical meaning could be attributed to the artifact giving inertia to the void. Whenever a high value is assigned to the "memory"

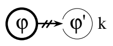
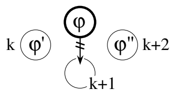
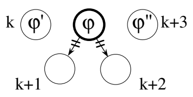
Rule	**Rule name**	**Configuration**
R_1	Frontal	
R_2	Interstitial	
R_3	Constrictive	

Fig. 3. Exclusion rules R_1, R_2, R_3: applied to phase ϕ, a rule works when the neighboring sites occupied by ϕ', or ϕ' – ϕ", contain equally or more coercive phases. The barred arrow stands for the exclusion of the target site.

of the void phase, this tends to induce, when the void moves in a dense packing, an "indraught" to the particle located in the active direction. During a sequence of transitions, this void will move a row of grains one at a time but in the same direction. Although the row only moves at a rate of one particle per timestep, a sort of effect of void *propagation* may occur. Let us recall that a phase component is meaningful, only when embedded into the interaction law.

2.4 Time Evolution Equations

Modeling a Phase. For a given timestep, a cell contains a particle of phase ϕ_i $(i \in N_\phi)$ characterized by the three following physical components.

– the action of an *external field*, depicted by a 6–fold vector

$$W_i = (w_i^{(k)})_{k \in K} \tag{1}$$

where weights $w_i^{(k)}$ are non-negative integers.

– the action of *exclusion rules*, precluding some direction or other depending on the state of the local vicinity and acting according to a *mode* from which the exclusion will be applied before (*pre*-exclusion) or after (*post*-exclusion) the request. This action is depicted by the 6–fold binary vector

$$\tilde{\mathcal{E}}_i = (\tilde{\varepsilon}_i^{(k)})_{k \in K} : \quad \tilde{\varepsilon}_i^{(k)} = r_i \; \varepsilon_i^{(k)} + (1 - r_i) \tag{2}$$

where $\varepsilon_i^{(k)} = 0$ (or 1) whenever the site in direction k is excluded (or not) and where $r_i = 1$ (resp. 0) for a pre (resp. post) mode assigned to the phase.

– the action of a "memory", or *inertial effect*, depicted by the 6–fold vector

$$\mathcal{M}_i = (\mu_i^{(k)})_{k \in K} \tag{3}$$

where $\mu_i^{(k)} = c_i$ if k was the displacement direction at the previous timestep and $\mu_i^{(k)} = 1$ otherwise. Coefficient c_i takes on positive integer values and $c_i = 1$ means no inertial effect for the phase.

Transition Algorithm. Prior to computing a request, a precalculation yields the corrected distribution

$$W_i^* = (w_i^{*(k)})_{k \in K} : \quad w_i^{*(k)} = \mu_i^{(k)} \; \tilde{\varepsilon}_i^{(k)} \; w_i^{(k)} \tag{4}$$

and according to the exclusion mode, the probability of sending a request in direction k is given by

$$p_i^{*(k)} = \varepsilon_i^{(k)} \frac{w_i^{*(k)}}{\sum_K w_i^{*(k)}} \tag{5}$$

on condition that the sum of the corrected distribution be positive ($p_i^{*(k)} = 0$ otherwise). Direction k is selected at random by a pseudorandom sequence generated from a user-defined seed.

Let $(p_j^{*(k)})_{k \in K}$ be now the distribution of probabilities of the neighboring particle of phase ϕ_j in direction k and let X_k be the representative event of a displacement in direction k for the current particle. The probability of this event is finally

$$P(X_k) = p_i^{*(k)} p_j^{*(k+3 \bmod 6)} \tag{6}$$

according to the exchange protocol.

3 Software Presentation

Grany-3 is a C++ application implementing a simulator of our LGrA logic described in the previous section. It is organized around the following modules:

- a core engine which executes the two-stage request-exchange transition algorithm,
- a graphical interface which allows the user to define scenes from a vector-based geometry, to parameter the granular system from a user-friendly window set and to visualize the simulation steps,
- a file manager based on a text-based interface which saves the data of the scene as well as the simulation in progress for further external use.

3.1 A Case Study

We illustrate the presentation through a simple granular system simulating a silo emptying process. The silo is a container with a hopper at the bottom, provided with an outlet[1] through which bulk grain falls down.

A two-phase system $N_\phi = \{1, 2\}$ is considered, where ϕ_1 denotes the "grain" phase and ϕ_2 the "void" phase. The initial state is a silo fully filled with grains.

We adopt a "3df–3df" law with $W_1 = (0, 0, 0, 1, 2, 1)$, $W_2 = (1, 2, 1, 0, 0, 0)$ where the pattern *SW–S–SE* of Fig 2b means "downwards".

The frontal exclusion rule R_1 is assigned to ϕ_1 with the pre-exclusion mode. The neutral rule R_0 is assigned to ϕ_2.

No inertial coefficient is assigned neither to ϕ_1 nor to ϕ_2, namely, $c_1 = c_2 = 1$.

The walls of the container will be created by means of a graphical editor. The cells in the outlet region will play a special role of "void generator" by generating a void when a grain exits. The instantaneous flow rate is the number of voids generated per timestep.

This parameter set defines the local interaction law at the microscopic level. Macroscopically, it will involve a kind of last-in first-out pattern of "funnel flow". In this example, we are not concerned by any physical interpretation of the observed phenomena: we are just interested by a presentation of the graphical interface.

[1] This term is preferable to "open source" in this context.

3.2 The Graphical Interface

Preparing a Scene. When starting the application, a *Scene Editor* should be open. After loading an existing scene, the scene editor activates the first `Borders` tab as shown in the front window of Fig. 4. Normal boundaries are drawn in blue, generating boundaries in red (the outlet generating "void" particles)[2]. To create a new `border` file, to open an existing one or to modify the existing borders, the buttons on the right side should be used accordingly.

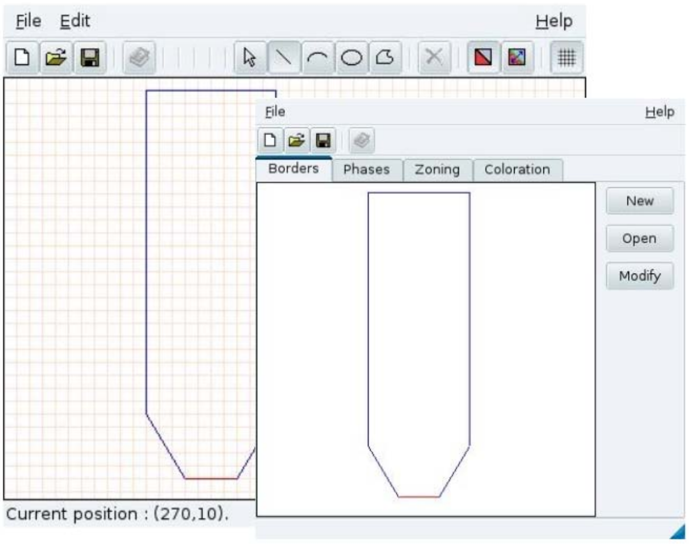
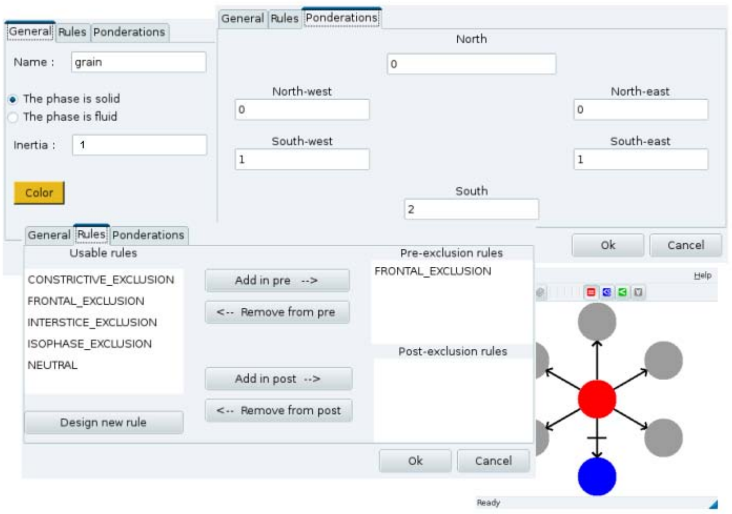


Fig. 4. The *Scene Editor* window, with its `Borders` tab activated. The *Border Editor* backwards, with its vector-based graphical tools displayed on the right of the toolbar.

When modifying the boundaries, a vector-based *Border Editor* is used as shown in Fig. 4 backwards. Available geometric forms (lines, arcs of ellipses, ellipses and polygons) are provided as buttons in the toolbar, and normal borders –or generating borders– can be switched from there too. A light pink grid is shown to facilitate the drawing of straight lines.

The second `Phases` tab of the scene editor will open a window to create the list of the two phases in the N_ϕ set. This main list is not displayed here. Phases will be edited with a `Properties` button and they can be set as `Propagative` and-or `Generated` by clicking an appropriate button. We create a first phase named `grain` and a second phase named `void` to which the `Generated` function will be assigned, that will involve a generation of voids in the red outlet region previously set up from the *Border Editor*. Neither the `grain` nor the `void` is defined as `Propagative`.

[2] We apologize for this coloured presentation in a black and white context.



Name : grain

● The phase is solid
○ The phase is fluid

Inertia : 1

	North	
	0	
North-west		North-east
0		0
South-west		South-east
1		1
	South	
	2	

Usable rules:
CONSTRICTIVE_EXCLUSION
FRONTAL_EXCLUSION
INTERSTICE_EXCLUSION
ISOPHASE_EXCLUSION
NEUTRAL

Pre-exclusion rules:
FRONTAL_EXCLUSION

Post-exclusion rules:

Fig. 5. The window environment to set up a selected phase. From back to front: the `General` window, the `Ponderations` window, the `Rules` window. A *Rule Editor* can be displayed to customize a new rule from the "`Design new rule`" button.

When the `grain` phase is selected in the `Phases` list, the `Properties` button opens a window with a `General` tab activated, as shown in the upper back window on the left side of Fig. 5. The `grain` is distinguished as a `solid` phase with no `Inertia` and an associated user-defined `Color`. The `Rules` tab displays a window with a list of standard or user-defined existing rules. The `FRONTAL_EXCLUSION` rule is selected in pre-exclusion by the "`Add in pre`" button. New rules can be customized with a *Rule Editor* activated by the "`Design new rule`" button and displayed on the right side in a window containing a 6-neighborhood template; since both frontal and neutral rules exist in the library, this tool will not be used in this example. Finally, the `Ponderations` tab displays a window to define the weighted vector W_1. A similar environment will be activated by selecting the `void` phase in the `Phases` list.

Activating the third `Zoning` tab of the *Scene Editor* opens a *Zoning Editor* to create the initial state of a silo fully filled with grains. The back window in Fig. 6 shows a relative mix of phases specified by the sliders on the right, with 100% for the `grain` phase and 0% for the `void` phase. The "filling" is carried out from one of the geometric frames available in the toolbar. Note that the zoning rectangle does not need to fit perfectly into the frame of the container: it suffices to roughly cover it. The colour is meaningful: it is the colour chosen in Fig. 5 by the user for the `grain` phase.

The last `Coloration` tab of the *Scene Editor* shows an additional layered coloration in the front window. This artefact is not physically meaningful but very useful: it merely overlays the normal color of the grain phase, in order to better visually track the movement of individual grains during the simulation.

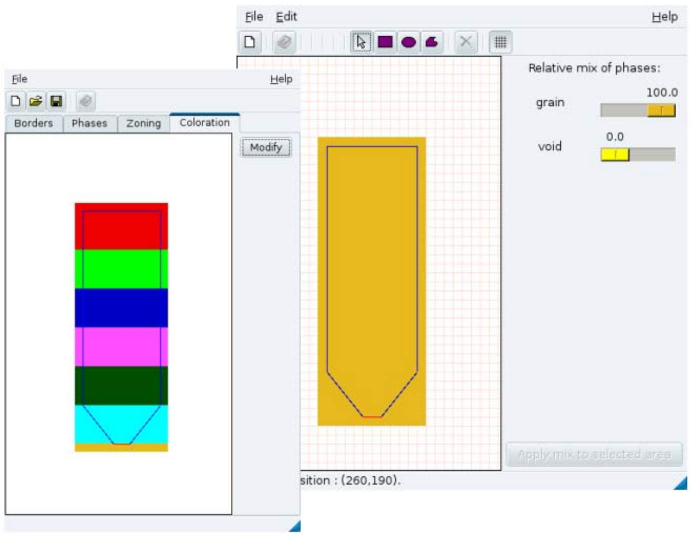


Fig. 6. The *Zoning Editor* window backwards, with its vector-based graphical tools on the toolbar to define various zoning regions. The *Scene Editor* in front, with its `Coloration` tab activated.

The scene is now ready, so we save it from the file manager toolbar of the *Scene Editor* as a “scene” file named `silo.scene`.

Creating a Simulation. To create a simulation, the scene file `silo.scene` should be reopened from the browser of a small dialog box, not shown herein. In this box, the user enters the three following parameters: the `Network size` of the cellular automata, the required `Number of steps` and the `seed` of the pseudorandom sequence.

Once the network size is known, the actual cells are generated to match the borders, phases, zonings and colorations of the scene as shown in the initial state of Fig. 7. The simulation toolbar is provided with a `Stop-Play-Pause` button set. The simulation starts when clicking on the `Play` button. Snapshots of the time evolution of the silo emptying process are displayed for $T = 0$, $T = 750$, $T = 1500$. From the toolbar, the user has a full control on the simulation steps and can save selected states from the file manager. Unless required otherwise, the simulation will stop once the number of steps is reached.

`Network size` and `seed` are two important parameters. The user can adjust the size of the model from coarse-grain (e.g. 16 x 16 = 256 cells) to fine-grain (e.g. 512 x 512 = 262144 cells) to study scale effects and to investigate scaling laws and critical phenomena for homogenization procedures. On the other hand, running several simulations with different seeds of the pseudorandom sequence allows the user to perform statistical investigations.

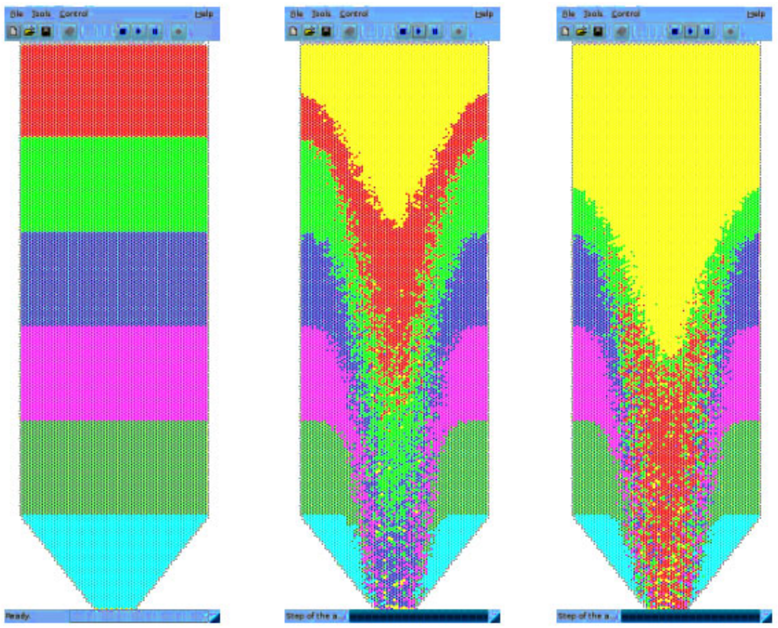


Fig. 7. Simulation of a silo emptying process. The toolbar displays the file manager tools to save data and the `Stop-Play-Pause` button set controlling the simulation. The cursor in the bottom displays the current step of the simulation in progress. The network contains 71289 cells. Snapshots at time $T = 0$, $T = 750$, $T = 1500$.

4 Conclusion

This software is very flexible and easy to use. In this case study, it is not difficult to modify a given parameter: for example, changing the angle of the hopper or the diameter of the outlet and observing their influence on the flow pattern could be readily performed, and so forth... The data produced by the file manager during a simulation process are available in a post-processing context for external use. In particular, flow patterns, flow rates, velocity profiles, density profiles, gradients, pressures, power laws and other macroscopic entities could be investigated. The aim of this paper was to present this open CA environment as a freeware that the researcher, involved in the study of the complex behavior of granular matter, may explore and customize for his own. Comparative studies with other computational methods like finite elements or granular dynamics appear as promising challenges which come within these objectives.

References

1. Wolfram, S.: Statistical Mechanics of Cellular Automata. Rev. Mod. Phys. 55, 601–644 (1983)
2. Kirkpatrick, S., Swendsen, R.H.: Statistical Mechanics and Disordered Systems. Comm. ACM 28(4), 363–373 (1985)

3. Frisch, U., Hasslacher, B., Pomeau, Y.: Lattice-Gas Automata for the Navier-Stokes Equation. Phys. Rev. Lett. 56(14), 1505–1508 (1986)
4. Wolfram, S.: Cellular Automata Fluids 1: Basic Theory. J. Stat. Phys. 45, 471–526 (1986)
5. McNamara, G.R., Zanetti, G.: Use of the Boltzmann Equation to Simulate Lattice-Gas Automata. Phys. Rev. Lett. 61(20), 2332–2335 (1988)
6. Rothman, D.H., Zaleski, S.: Lattice Gas Cellular Automata. Cambridge University Press, Cambridge (1997)
7. Chopard, B., Droz, M.: Cellular Automata Modeling of Physical Systems. Cambridge University Press, Cambridge (1998)
8. Talia, D., Sloot, P.: Cellular Automata: Promise and Prospects in Computational Science. Special issue of Fut. Gen. Comp. Sys. 16, 157–305 (1999)
9. Baxter, G.W., Behringer, R.P.: Cellular Automata Models of Granular Flow. Phys. Rev. A 42, 1017–1020 (1990)
10. Peng, G., Herrmann, H.J.: Density Waves of Granular Flow in a Pipe Using Lattice-Gas Automata. Phys. Rev. E 49, R1796–R1799 (1994)
11. Károlyi, A., Kertész, J., Havlin, S., Makse, H.A., Stanley, H.E.: Filling a Silo with a Mixture of Grains: Friction-Induced Segregation. Europhys. Lett. 44(3), 386–392 (1998)
12. Ktitarev, D.V., Wolf, D.E.: Stratification of Granular Matter in a Rotating Drum: Cellular Automaton Modelling. Granular Matter 1, 141–144 (1998)
13. Désérable, D., Masson, S., Martinez, J.: Influence of Exclusion Rules on Flow Patterns in a Lattice-Grain Model. In: Kishino, Y. (ed.) Powders and Grains 2001, Balkema, pp. 421–424 (2001)
14. Cisar, S.E., Ottino, J.M., Lueptow, R.M.: Geometric Effects of Mixing in 2D Granular Tumblers Using Discrete Models. AIChE Journal 53(5), 1151–1158 (2007)
15. Désérable, D., Dupont, P., Hellou, M., Kamali-Bernard, S.: Cellular Automata Models for Complex Matter. In: Malyshkin, V.E. (ed.) PaCT 2007. LNCS, vol. 4671, pp. 385–400. Springer, Heidelberg (2007)
16. Désérable, D.: A Versatile Two-Dimensional Cellular Automata Network for Granular Flow. SIAM J. Applied Math. 62(4), 1414–1436 (2002)
17. Cottenceau, G., Crunchant, S., Garcia, P., Le Guelvouit, G., Michard, X., Padioleau, Y., Zemali, Y.: Grany–3 Project: Design of a Cellular Automaton Simulator Dedicated to Granular Media. Technical Report, INSA Computer Science Department, Supervisors: Désérable, D., Rozé, L (1999)
18. Cottenceau, G.: Grany–3, `http://freshmeat.net/projects/grany3`
19. Litwiniszyn, J.: Application of the Equation of Stochastic Processes to Mechanics of Loose Bodies. Archivuum Mechaniki Stosowanej 8(4), 393–411 (1956)
20. Müllins, W.W.: Stochastic Theory of Particle Flow under Gravity. J. Appl. Phys. 43, 665–678 (1972)