An Event-Driven Algorithm for Fractal Cluster Formation

S. González, A. R. Thornton, S. Luding

Multi Scale Mechanichs, TS, CTW, UTwente
P.O.Box 217, 7500 AE Enschede, Netherlands

Abstract

A new cluster based event-driven algorithm is developed to simulate the formation of clusters in a two dimensional gas: particles move freely until they collide and "stick" together irreversibly. These clusters aggregate into bigger structures in an isotropic way, forming fractal structures whose fractal dimension depends on the initial density of the system.

Keywords: fractals, event-driven, granular matter, simulation

1. Introduction

Cluster formation is an important subject in various areas of physics. In astronomy, ice clusters are believed to aggregate into planetesimals [1], the base of what we know as planets today. In granular materials, our area of research, tiny nano-Newton forces are responsible for macroscopic clusters in free falling jets [2] – similar to the ones that appear in nano jets from plasma physics [3]. Clusters are also found in granular avalanches [4], or air-driven granular beds [5].

Motivated by nanoaerosols [6], a cluster based event-driven algorithm is developed to simulate the formation of clusters in a $2D$ gas with periodic boundary conditions: particles move freely until they collide and "stick" together irreversibly, moving as one cluster. The dynamics of the clusters is utterly simplified in our model, conserving linear momentum and disregarding angular momentum in the system. These clusters evolve and aggregate into bigger structures. The fractal dimension obtained with this algorithm is found in the range $1.5 < d_f < 2$. In contrast to the case of diffusion-limited aggregation (DLA), where $d_f = 1.71$ is found [7], we keep track of the dynamics of the clusters instead of adding particles one by one. This procedure is similar to the one occupied in Ref. [8], where the coalescence of particles is studied.

Implementing clusters in an event-driven algorithm has two advantages: On the one hand, defining clusters of particles avoids the need to predict the events between particles of the same cluster. Since particles in a cluster move together as a rigid solid, they cannot collide. This alone decreases the computational effort required in simulating the clusters, where most of the collisions of the system occur [9]. On the other hand, the concept of clusters appears in a wide range of particulate physics: granular structures develop long correlations in space and time, see e.g. Ref. [5] where is found that particles move in one-dimensional paths ("strings") that aggregate into clusters.

In the next section we explain the algorithm that we use and how it is related to the classical event-driven model. After that, the experiments that we realized are presented. Concluding remarks and plans for future work end the paper.

2. Algorithm

By event-driven we mean that the state of the system is evolved in time from one event to another, predicting the time for the next (future) event after a present event has taken place. For the details of the algorithm we refer the reader to standard books, see e.g. Ref. [10]. For the moment, it is enough to say that the algorithm consists of:

1. Given the instantaneous positions and velocities for all the particles in the system.
2. Predict the time of the next collision.
3. Update the velocities of the particles that collide with a given collision rule.
4. Advance the time of the system to that instant, update the positions and velocities, and repeat from 1.

The event-driven algorithm presented here comes in the line of previous work. When the static phase in dense granular systems is simulated with a different dynamics, the performance of the simulation is improved [11]. This is a necessary step towards a multiple-scale event-driven simulation for granular matter, where each cluster can have its own dynamics and collision rules.

The kind of clusters we are interested in at the moment, are suspensions of nanoparticles in a gas, which stick together at contact due to Van der Waals forces (as e.g. Ref. [12]). In reality, clusters of particles conserve angular momentum when they collide, which results in rotating clusters. For the sake of the simplicity, and since –at the moment– we are mainly interested in the algorithm rather than in recovering the right physics, we will disregard the rotational motion of the clusters and will consider only translational motion.

In normal event-driven algorithms one has to predict the next event – a collision – between all two-particle pairs. In this version, we introduce a new object called cluster (which may consist of just one particle), and only collisions between these objects have to be computed. Since a cluster consists of a finite number of particles, the position of a particle within a cluster is given by

$$\vec{r}_{\text{cluster}}(t) = \vec{r}_0 + \vec{v}_{\text{cluster}} t,$$

where $\vec{v}_{\text{cluster}}$ is the linear velocity of the cluster. The time is measured since its last collision, so r_0 is the center of mass of the cluster at that instant.

Now that we have defined the evolution of particles within a cluster, collisions between particles in different clusters can be detected. This is a massive time saving as collision between particles within the same cluster do not have to be checked for, and as the size of the clusters increases the number of checks decreases. Once the collision of two clusters has been detected the colliding particles "stick" together and the two cluster are combined into a single larger one. The velocity of the newly formed cluster is calculated by considering the conservation of linear momentum only. This process is repeated until the system only consists of a single cluster.

In order to find the time of collision between two particles i, j we need to find the first positive root of

$$|\vec{r}_i(t) - \vec{r}_j(t)|^2 = d^2,$$

with d the diameter of a particle. Classical event-driven model just need to deal with a quadratic equation, both in the case with or without gravity. If we consider static particles in coexistence with normal particles under gravity, the equations to be solved are quartic or cubic [11].

The inclusion of rotating clusters in the simulation makes the equation to find the collision time highly nonlinear. In general, this means to find the intersection of two finite volume helices in $3D$. This is why we will consider a toy model where just linear momentum is conserved, making the equation to solve easier.

Summarizing, the simulation procedure can be described as follows:

1. Start with an initial configuration of particles.
2. Find the time for the next collision in the system.
3. Advance the system to that instant and merge the two particles (clusters) into a single cluster. The post-collisional velocity of the new cluster is such that linear momentum is conserved.
4. Predict the new events for this cluster (only).
5. Repeat until all the energy is dissipated and a single cluster is present in the simulation.

Three snapshots of a simulation are shown in Fig. 1. At the beginning of the simulation (a), particles are arranged in a square lattice with random velocities. The color code represents different clusters in the simulation. In this case, every particle correspond to a cluster of size one. At a later time, clusters of different size coexist in the simulation (b) and aggregate as soon as they are in contact. Finally, the system contains only two clusters (c) that will collide in the next event of the simulation, ending the aggregation process.

A note on the speed of the algorithm must be added. The classical event-driven simulation is fast compared to soft-particle molecular dynamics. Among other reasons, because the collision time between two particles can be found analytically. Solving non-linear equations will make the algorithm more time consuming. This drawback can be compensated with a multi-core implementation of the algorithm. In this case, the calculation of the collision time between one given particle and its neighbors can be done in parallel. This approach is not viable in classical event-driven simulations due to the big data transfer overhead associated and the relatively small amount of computation that is needed. In the classical case, the collision time is given by solving a quadratic equation. For the rotational case, a non-linear equation must be solved, making a more expensive computation. Parallel implementations of classical event-driven simulations have been done, but with suboptimal speedup, see [13, 14]. We are currently working on this issue.

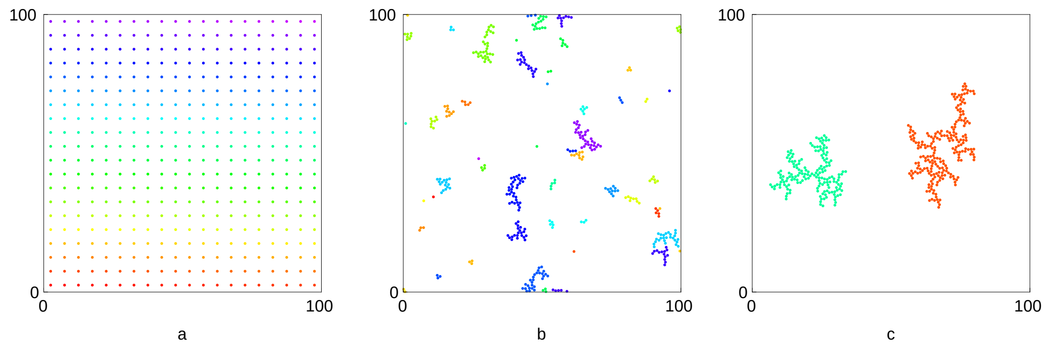

Figure 1: Three snapshots for the evolution of a system of $N = 400$ particles in a box of size $L = 100d$, a packing fraction of $\nu \simeq 0.125$. Each color represents a different cluster. Time increases from left to right.

3. Experiments

The simulation consists of a system of N particles in a $2D$ square box of size L with periodic boundary conditions. Particles are monodisperse with radius $d/2$ and mass m. The packing fraction of the system is given by $\nu = N\pi d^2/(4L^2)$. In order to start with a random configuration, we let the system equilibrate: starting from a square lattice, each particle collides at least 10 times elastically until a homogeneous regime is reached. The initial average speed is $0.36v$. This velocity will set the time scale of the system.

Once thermalized, the clustering algorithm is switched on, and the simulation runs until one big cluster is formed and all the energy of the system is dissipated (the simulations are run in the center of mass reference frame).

3.1. Temporal Evolution

The temporal evolution of the mean cluster size is shown in Fig. 2. The system reaches the final stage after approximately $\log(N)$ collisions, which happens around a time $t \sim 10^7[d/v]$. With this example we can appreciate the advantage of making an event-driven simulation: since the system advances through events, the computational cost does not scale with the physical time of the system, but just with the number of collisions. Since the energy is dissipated in the system, the collision rate decreases, increasing the physical time between events [15]. If done with molecular dynamics, the computational cost would be enormous due to the usually fixed integration time step.

The scaling behavior of the energy was also studied. We found that for the most dilute system, the mean kinetic energy per particle follows a power law $\langle E_K \rangle \propto t^{-\delta}$ with $\delta = 1.26$. This results is similar to the one from Ref. [8], where a scaling of $\delta = 1.12$ was found.

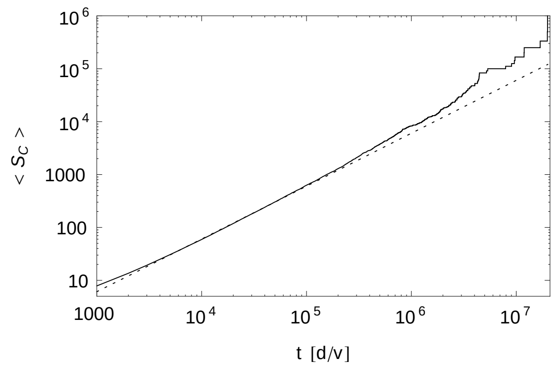

Figure 2: Mean cluster size $\langle S_C \rangle$ as a function of time for a system with $N = 10^6$ and $\nu = 0.0078$. The dotted line shows a slope of 1 and best fits in the middle part of the simulation.

3.2. Fractal dimension

With the final configuration from the simulation, we count the number of particles present in a circle of radius r around ten randomly chosen particles of the cluster. We do this to obtain the number distribution, which exponent is the dimension of the system. We confirmed that the fractal dimension was independent of the points selected by choosing the points in the inner third or in the outer third of the fractal: booth measurements lead to the same results.

The results for a system with $N = 10^6$ and $\nu = 0.0078$ are shown in Fig. 3 where points are the experimental data, and the line is the power law fit. Due to the finite size effects, to fit the data we disregard the last two points. The slope of the best fit is given by $d_f = 1.56 \pm 0.01$.

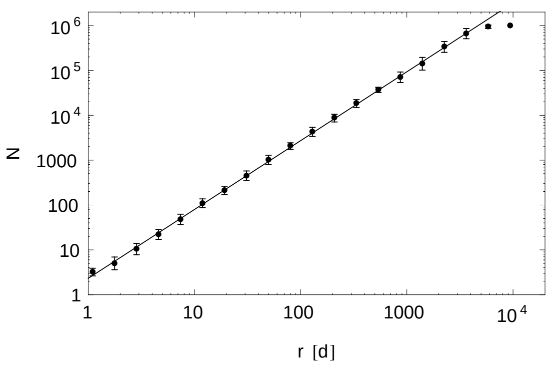

Figure 3: Number of particles in a circular given region as a function of the radius for a system with $N = 10^6$ particles and $\nu = 0.0078$. The line indicates a slope of 1.56.

3.3. Role of density

The fractal dimension we obtain is dependent on the density of the system. For example, if we start with a very dense system, there is no arrangement possible and the final state will practically coincide with the initial state. Due to this, an integer dimension of $d_f = 2$ is expected for dense systems. On the other hand, we expect an asymptotic fractal dimension for vanishing density.

To measure the effect of the density, we vary the size of the system for a given number of particles $N = 10^6$. The system sizes chosen are in the range $1000d \leq L \leq 10000d$, corresponding to densities between $0.0078 \leq \nu \leq 0.78$.

Figure 4 shows the fractal dimension obtained as a function of the density of the system. As expected, for high densities the fractal dimension approaches 2, namely for $d_f(0.78) = 1.97 \pm 0.01$, that is, the cluster resembles a two dimensional solid. On the other hand, for vanishing densities it is found that $d_f(\nu) \rightarrow 1.5$ for $\nu \rightarrow 0$. This fractal dimension is smaller than the one found, for example, for the diffusion-limited aggregation process [7], where the fractal dimension is $d_{\mathrm{DLA}} = 1.71$.

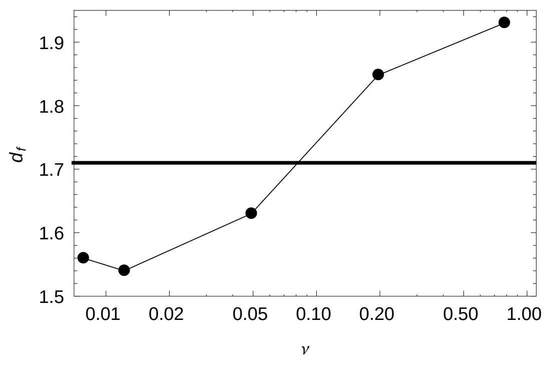

Figure 4: Fractal dimension as a function of packing fraction for systems with $N = 10^6$ particles, together with the DLA fractal dimension (thick solid line). The dots are the simulation results while the solid line is just a guide to the eye. The error bars here are smaller than the symbols.

4. Conclusions

In this paper we have presented event-driven simulations of irreversible aggregating clusters in a $2D$ system. These clusters have non-physical dynamics but represent a toy model that permits us to understand how to make cluster simulations in an event-driven algorithm. The formation of fractals was studied, and the exponent found depends on the initial density of the system. Depending on the density, the resulting fractal structure has a dimension in the range $1.5 < d_f < 2$. The denser the system, the closer to a two dimensional structure the fractal is. The inclusion of more realistic dynamics and collision rules for the clusters is currently investigated. This will allow us to use a multicore implementation of the event-driven algorithm, as the subject of a future paper.

References

[1] C. Dominik, A. Kilns, ApJ, **480**, 647 (1997).

[2] J. R. Royer, D. J. Evans, L. Oyarte, Q. Guo, E. Kapit, M. E. Möbius, S. R. Waitukaitis, H. M. Jaeger, Nature **459**, 1110-1113 (2009).

[3] M. Moseler, U. Landman, Science **289** (5482), 1165 (2000).

[4] D. Bonamy, F. Daviaud, L. Laurent, M. Bonetti, J. P. Bouchaud, Phys. Rev. Lett. **89**, 034301 (2002).

[5] Keys, A. S., Abate, A. R., Glotzer, S. C. & Durian, D. J. Nature Phys. **3**, 260264 (2007).

[6] A. Schmidt-Ott, Appl. Phys. Lett. **52**, 954 (1988).

[7] T. A. Witten, L. M. Sander, Phys. Rev. B **27**, 56865697 (1983).

[8] E. Trizac, J. P.Hansen, Phys. Rev. Lett. **74**, 4114 (1995).

[9] S. Miller and S. Luding, Phys. Rev. E 69(3), 031305 (2004).

[10] T. Pöschel and T. Schwager, *Computational Granular Dynamics: Models and Algorithms*, Springer (Berlin 2005).

[11] S. González, D. Risso, R. Soto, Eur. Phys. J. Special Topics **179** 33-41 (2009).

[12] J.H. Werth, H. Knudsen, H. Hinrichsen, D.E. Wolf, Phys. Rev. E **73**, 021402 (2006).

[13] S. Miller, S. Luding, J. Comp. Phys. **193**(1), 306-316 (2004).

[14] M. Marín, Comp. Phys. Comm., **102**, Issues 1-3, 81-96 (1997).

[15] S. Luding, H. J. Herrmann, Chaos **9**(3), 673-681 (1999).