Cellular Automata Modeling of Snow Transport by Wind

Alexandre Masselot * Bastien Chopard †

December 12, 1995

Parallel Computing Group
CUI, University of Geneva
24, rue du Général Dufour
1211 Genève 4 Switzerland

Abstract

We present a lattice gas model to simulate snow transport by wind and its deposition on a given ground profile. Our approach is very well suited to a fine grained massively parallel computing.

keywords: cellular automata, lattice gas, massively parallel computing, wind modeling, snow-drift, snow deposit.

*email : masselot@cui.unige.ch
†email : chopard@cui.unige.ch

1 Introduction

Massively parallel computing offers new approaches to many scientific problems by allowing the processing of a large volume of data in an acceptable amount of time. Here, we present a numerical simulation of snow transport by wind, which is a phenomena involving many complex processes.

Predicting the location of snow deposit after a wind period is a crucial question to protect roads against snow drift or control the formation of wind-slab (responsible for about 80% of accidently caused avalanches). Placing snow fences at appropriate locations can substantially modify the shape of the deposit, by screening out the wind. Snow fences are also used to store snow on some ski trails. Thus, a reliable simulation technique of snow transport has a large impact in exposed areas such as a mountain environment.

The numerical research on this subject is still not very developed and we proposed a very intuitive 2-D approach to transport, deposition and erosion of snow particles by wind. Our model considers a description at the particles level: wind and snow "molecules" are the basic constituents and the dynamics takes into account only the essential microscopic phenomena, namely those still relevant at a macroscopic scale of observation.

Note that the transport of sand and the question of dune formation is a closely related problem which can also be addressed in the present framework.

Our approach is based on the cellular automata techniques to simulate a fluid motion: the so-called FHP lattice gas model[1]. This method is ideally suited to massively parallel computations and constitutes a promising alternative to the more traditional numerical solutions of partial differential equations when complex boundary conditions are involved.

In a first step, we briefly review the cellular automata method of fluid modeling. Then, we explain how snow particles in suspension can be added and how they interact with wind and pile up on the ground. Finally we discuss the implementation of this model on a Connection Machine CM-200.

2 The model

2.1 The Cellular Automata Approach

In a cellular automata approach, the real physical world is represented as a fully discrete universe. Space is discretized as a regular lattice and particles move *synchronously* according to discrete time steps with a finite set of possible velocities. Mutual interactions are local.

In order to guarantee space isotropy, it has been shown that modeling a fluid flow[1] requires a hexagonal lattice (with the six directions $\vec{c_i}, i \in \{0, \ldots 5\}$), as shown in figure 1.

The description of the dynamics considers an site occupation variable n_i. At each site r and time t, there is $n_i(r,t) \in \{0,1\}$ particle traveling along the i^{th} direction (with velocity unity).

Each time step is made of two phases:

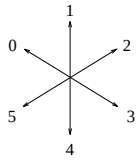

Figure 1: *The six lattice directions*

- Collision: when several particles meet at the same lattice site r, they interact so as to modify the directions of motion

$$n_i'(r,t) \leftarrow f_i(n_0(r,t), \ldots n_5(r,t)) \tag{1}$$

- Propagation: then, the particles move to a nearest neighbor site, according to their new direction of motion

$$n_i(r + \vec{c_i}, t+1) \leftarrow n_i'(r,t) \tag{2}$$

On a massively parallel computer, the spatial domain is partitioned and each part assigned to a different processor. In the data-parallel language, the strategy is to allocate a virtual processor for every lattice sites. All the processors can therefore simultaneously compute step (1). For step (2), each processor communicates with its six neighbors (north, west, south-west, south, east, north-east).

To build an hexagonal lattice from the processor interconnection topology, we transform the square lattice as shown in fig. 2.

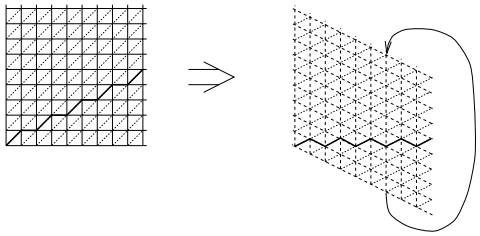

Figure 2: *transformations to get an hexagonal lattice*

2.2 Wind modeling

We shall represent the wind as a cellular automata fluid composed of many elementary particles. From the fundamental laws of physics, collisions between wind particle must satisfy mass and momentum conservation. This is the key ingredient in a microscopic model. Thus, we impose

- the mass ($\sum_{i=0}^{5} n_i$) is locally conserved
- so is the momentum ($\sum_{i=0}^{5} n_i \vec{c_i}$)

- collisions rules are the same modulo $\frac{\pi}{3}$ rotations.

One of the simplest cellular automata model of fluid is the FHP model proposed in 1986[1]. If only one particle is entering a site, it goes straight to its neighbor in the direction of motion. If many particles are entering a site, they collide according to the rules given in fig. 3

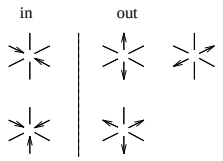

Figure 3: *FHP collisions rules; in the first case, we must choose randomly between the two outcomes. All the other configurations remain unchanged.*

A FHP fluid has a built-in viscosity, determined by the above collision rules. For many practical purposes, the viscosity of the FHP model is to large to simulate flows at high enough Reynolds numbers. In the so-called FHP-III model, rest particles and more sophisticated collision rules (for example, see fig 4) are considered to lower the viscosity.

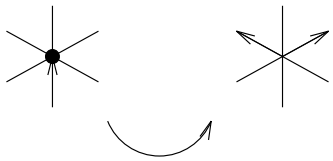

Figure 4: *One of the FHP-III new rules*

Ultimately, we would like that our cellular automata fluid obeys the physical laws of fluid motion. The Navier-Stokes equation is the standard equation describing the flow of a real fluid. In the macroscopic limit and within some approximations, it can be shown[1] that the FHP dynamics indeed evolves according to the Navier-Stokes equation.

2.2.1 Boundary Conditions

For our simulation, we consider a two-dimensional hexagonal lattice of size $L_x \times L_z$, corresponding to a vertical slice of the real three-dimensional space. Wind is blowing from left to right. Since we have only a finite system, we must find appropriate boundary conditions to simulate an infinite system in the x-axis and semi-infinite in the z-axis.

Ground boundary: Ground is modeled by "solid" lattice sites. We either apply the slip (bounce forward) or the no-slip (bounce back) condition when a wind particle collide with the ground (or with a snow deposit).

Sky boundary: On the upper boundary, we simply inverse the vertical speed of a particle, without modifying its horizontal speed.

Left and right boundaries: Wind particles are injected, at a given speed from the left side. At the right extremity, they should behave as if the system were infinitely long. This problem is solved by considering a zero-gradient method: we add particles from the left to the right, and from the right to the left, according to three criteria:

- the density (number of particles) is conserved over the tunnel.
- The input speed has a given value.
- The vertical density profile of particles inserted backward is proportional to the profile observed a few sites upwind (zero-gradient method).

2.3 Snow modeling

Snow particles are also represented as particles moving on the same lattice as wind particles. Without a wind flow, snow particles fall due to gravity. When they reach the ground, two kinds of phenomena can be considered

- There is a cohesion force and each particle sticks to the deposit, whatever the local configuration is.
- There is no cohesion and, at the landing, a particle rolls down until it finds a stable configuration (see fig 5).

Figure 5: *situation (1) is stable, but the (2) is not; the flake must go down.*

2.4 Wind-snow interactions

The real wind-snow interactions are quite complex and still not fully understood. According to the spirit of our model, we consider only simplified interactions. Two main phenomena are taken into account in the present version of the model:

Transport: Flying snow is subject to gravity, but also to wind forces. On each lattice site, we compute the new direction of a snow particle by computing the mean wind speed in an hexagonal neighborhood. This driven force is combined with gravity. This "force" acts statistically, due to the restrictions imposed by a cellular automata approach.

Erosion: Up to 80% of snow transport takes place due to the phenomena of saltation [3]: a micro-eddy pulls out a snow flake and, sometimes, a larger one makes it fly away. We model this phenomenon by assigning to each landed snow particle a counter of wind impacts; after a given number of impacts the snow particle can take off and, perhaps, fly if the wind is locally strong enough.

3 Computer Implementation

3.1 From boolean...

On each site r, the wind configuration is given by $b = 6$ bits (7 with rest particles), which can be stored as an integer $n(r) = \sum_{i=0}^{b-1} 2^{n_i}$. Our two-phase algorithm (collision and propagation) discussed in section 2.1 becomes:

- $n(r) \leftarrow f(n(r))$ where $f : \{0, 2^b - 1\} \mapsto \{0, 2^b - 1\}$; one notices that the function f can be computed once, and the results for all the values $f(i), i \in \{0, 2^b - 1\}$ stored in an array (lookup table) on each processor,
- we stream the i^{th} bit of $n(r)$ in the direction $\vec{c_i}$

The wind and snow configurations are both fully determined by 6 or 7 bits. The solid configuration also requires 7 bits per site (6 of them to know whether there is a solid site in direction $\vec{c_i}$ and the last one to know if the site itself is solid).

Therefore, the configuration at a given site is fully determined by $7 + 7 + 7 = 21$ bits and we could make a general evolution function $\{F : \{0, 2^{21} - 1\} \mapsto \{0, 2^{21} - 1\}$. Problems occur when we try to store 2 mega-words on each processor, which cannot be done on our CM-200 (although each lookup table copy is shared among 32 processor). We must split the evolution in four interactions (wind-wind, wind-snow, wind-solid, snow-solid).

3.2 ...to real valued quantities

Instead of describing the dynamics in terms of the absence $n_i = 0$ or presence $n_i = 1$ of a wind particle in every direction, we can also look at the probability $< n_i > \in [0, 1]$. That is the *Boltzmann approximation* which neglects N-body correlations and replaces the boolean operators .or., .and. & .not. by the floating point operations $+, *, 1 - ..$ A direct computer simulation of the evolution equation for the $< n_i >$ is a technique known as the Lattice Boltzmann Method (LBM)[2]. It reduces statistical noise, and allows much more flexibility when adjusting the system parameters (such as density, entry speed, viscosity...).

4 A few results

The simulations we have performed so far with our model give quite promising results. Good qualitative agreement[4] is obtained with *in-situ* observations and wind tunnel experiments. The next two figures illustrate the snow deposit behind a fence and the wind behavior around it. Depending on the type of numerical experiment, the execution time typically ranges from several minutes to a few hours, for a system of size 128×1024 on a 8k processors CM-200.

In collaboration with the Swiss Institute for Snow and Avalanche Research, we are currently improving the model toward a more effective LBM scheme and a better modeling of snow-wind interactions.

Acknowledgments

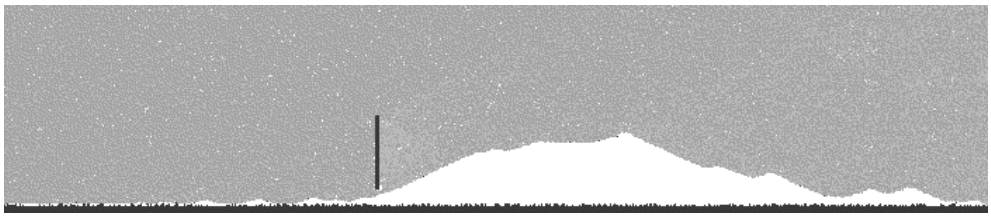

Figure 6: *Cellular Automata model. Wind blows from left to right trough a tunnel. The fence has a ground clearance. Snow deposit is qualitatively close to reality.*

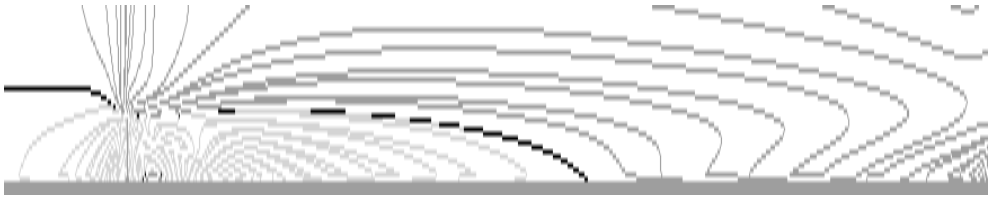

Figure 7: *Reattachment wind isolines, when the wind is modeled by LBM. Same reattachment distances are observed in real wind tunnel.*

This research is supported by the Swiss National Science Foundation.

References

[1] U. Frish B. Hasslacher et Y. Pommeau. *Phys. Rev. Lett. 56 (1986) 1505*; "Lattice gas method for partial differential equations," G. Doolen Edt., Addison-Wesley, (1990).

[2] R. Benzi S. Succi M. Vergalossa. The lattice Boltzman equation : theory and application. *Physics Reports 222 No 3 (1992) p. 145-197*

[3] Thierry Castelle. Transport de la neige par le vent en montagne: approche expérimentale du site du col du Lac Blanc. *PhD thesis, EPFL Lausanne, (1995)*

[4] J.K. Raine D.C. Stevenson. Wind protection by model fences in a simulated atmospheric boundary layer. *Journal of Industrial Aerodynamics, 2 (1977) p.159-180*