Changing the levels of description of a fluid flow in an agent–based simulation

Pierrick Tranouez, Cyrille Bertelle and Damien Olivier
Laboratoire d'Informatique du Havre
25 rue Philipe Lebon
BP 540, 76058 Le Havre Cedex, France
E–mail: {Pierrick.Tranouez, Cyrille.Bertelle, Damien.Olivier}@univ–lehavre.fr

ABSTRACT

We present in this work an agent–based simulation for hydrodynamic flow in heterogeneous environment. Vortex methods are our distributed formulations allowing a decentralizated computation for the entities interactions. Some processes for the detection of emergent organisations are described. The aim of this work is to be able to manage such computed organisations as new entities which are recursively able to aggregate themselves. A multi–scale computation is so described by this way.

SECONDARY AUXILIARY ENERGY : THE WATER FLOW

Some structures in natural systems appear far from the thermodynamic balance. The thermodynamic of dissipative systems demands that matter and energy flows go through the systems to maintain these structures.
The water flow in aquatic ecosystems is such a flow. Allowing the interactions and communications (in an information theory meaning) of entities in the milieu, it is what is called in ecology a secondary auxiliary energy [Frontier and Pichot–Viale1998].
Our work in estuary ecosystems simulation led us to tackle the representation of fluid flows .

MODELLING A FLUID FLOW

Computer simulations deal with discrete notions. While trying to model a fluid flow, we therefore only have an interest in discrete ways of describing the flow.

Meshes

The most widely used discrete model in hydrodynamics, meshes are based on a grid–like description of the flow, where various variables describing it (such as speed or acceleration) are computed on the nodes of the grid.
One of the main downsides of this method is numerical diffusion : calculations are made on the nodes and the fluid is supposed to behave regularly between them. It can be a problem when this is not the case

Particular description

Other discrete models, somewhat less used, are called vortex methods [Leonard 1980]. Their theoretical basis is to break up the flow in vortex particles. These particles move like the fluid they describe, and are at the same time indicators of hydrodynamic variables such as speed or vorticity [Chorin 1973].
These particles should not be thought as only tiny blocks of moving fluid but also as abstract indicators of the local behaviour of the fluid. Their speed and overall trajectories are computed using the values of all the other particles across the fluid, The effects of numerical diffusion are downplayed through the creation of new vortex particles where non–linear behaviour is expected, such as around obstacles. Increasing thus the number of particles increase the level of detail of description of these more complex phenomena, all the more since the particles move along with them, as opposed to grids that have a harder time adapting their density to the trajectory of the phenomenon.
Furthermore, the calculations are distributed among the particles as opposed to centralised in differential equations, thus allowing the use of new computer tools for simulation. These tools for example handle better the interactions of various heterogeneous entities. Indeed in computer simulation paradigms such as agent–based programming, local models are used to define the behaviour of each agent.

AI APPROACHES FOR FLUID FLOW MODEL

The use of Artificial Intelligence introduces new paradigm in hydrodynamic. One of the main developments in this field is based on artificial neural networks when complexes phenomena appear and no clear mathematical formulation is present. In this case, adequate data sets must be available and some specific parameters are adjusted in given input–output relations [A.E. Mynett 1999]. Genetics algorithms can also be used to optimise such network architectures [R.J. Abrahart 1999].

More recently, Agent–driven simulations are used for runoff, infiltration and erosion processes in the field of hydrodynamical modelling [D. Servat 2000]. Some artificial water drops are modelized in term of agents interacting themselves. They follow various laws of interaction, based on water height computation in term of

neighbouring water drops and then, on the velocity computation of each water drop as result of the water height in the proximity. Organizational entities are able to emerge and modelize pool, corresponding to one−level scale transition. These higher−level entities have different comportment law and the first−level scale transition from drop to pool is not recursively transposable to higher level scale transition.

Our purpose, in this paper, is to be able to recursively change the spatial scale using a generic processus of detection of organisations. These organisations are so able to aggregate themselves, or with other−level entities. This is made with the same scheme that the one used for their formation. For this approaches, an agent−based model is described in the following.

CHANGING THE SPATIAL SCALE

Basic particles organisations

The main structures appearing in a flow are vortexes. A part of changing the level of description will therefore be detecting these vortexes and making them an entity of the model. this creates a higher level of description [Bertelle et al 2000b].

Interacting organisations : a fractal model

These vortexes interact with both lower level particles but also other vortexes. This latter interaction lead to the creation of vortexes of a higher level, combining various smaller vortexes or even basic particles. These bigger vortexes constitute yet other scales. These scales all work along the same rules : we have a fractal organisation.

Changing the scale : a two−way process

These changes of level of description happen in the two ways. If lower−level entities get structured, they may be substituted for a more global entity, but when these organisation entities interact, be it among themselves or with elementary entities or other species entities, it is always a solution to break them up in the smaller members from whom they emerged, so as to handle these interactions as close as possible to what the physical model imposes.

USING A MAS TO IMPLEMENT THIS MODEL

Describing the physical model : Active Objects

In vortex methods, fluid flow is split up into N fluid particles located at $(\vec{X}_i)_{1\leqslant i\leqslant N}$ and carrying vorticity $\vec{\Omega}_i$ which characterises their ability of rotation round themselves. A dense set of such artificial vortex particles can represent a fluid flow with accurate precision.
In two dimensional space, at each time t of the simulation, the algorithm computes the part of particles velocity that is deduced from other particles rotation, using regularised Bio−Savart formula:

$$\vec{U}_p(\vec{X}_i,t)=\frac{1}{2\,\Pi}\sum_{j\neq i}\vec{\Omega}_i\wedge\frac{(\vec{X}_i-\vec{X}_j)}{(\|\vec{X}_i-\vec{X}_j\|^4+\epsilon^4)^{1/2}}$$

where ϵ is a regularisation parameter whose value can be set approximatively to $h^{3/4}$ and h the minimal distance between two neighbouring particles.
However, if the particle set is convected by a global and irrotational flow, with an average vorticity $\vec{U}_\infty(t)$, then the total velocity of each particle is the sum of this global irrotationnal flow and the velocity provided by all the other particles rotation:

$$\vec{U}(\vec{X}_i,t)=\vec{U}_p(\vec{X}_i,t)+\vec{U}_\infty(t)$$

The position of each particle is computed with its own velocity integration:

$$\frac{d\,\vec{X}_i}{dt}(t)=\vec{U}(\vec{X}_i,t)$$

If the fluid is considered viscous, with viscosity ν and V_i the i^{th} particle volume, the vorticity evolution is computed, integrating differential equation, following [Choquin and Huberson 1988]:

$$\frac{d\,\vec{\Omega}_i}{dt}=\frac{\sum_j(V_i\vec{\Omega}_j+V_j\vec{\Omega}_i)\exp(-\frac{\|\vec{X}_i-\vec{X}_j\|^2}{4\nu\Delta T})}{4\pi\nu(\Delta T)^2}$$

Each of these particles will be implemented by active objects.

Describing structures and organisations : MAS.

Detecting the basic structures
We detect coherent and stable vortex structures which are composed of particles of the same rotation direction. We've so far used a minimum spanning tree computation over a domain decomposition based on Delaunay's triangulation.
We're currently experimenting other clustering methods, mainly Kohonen maps.
Reifying the structures by an agent
Once these structures are detected, we dismiss the particles from whom it emerged and reify an agent that embodies said structure. It is in part possible as hydrodynamic knowledge teaches us that one particle cannot belong to two different stable structures. The environment of this agent can be broken up in an inert one and a social one.
The inert environment will be the flow particles. The proaction of the agent will be of trying to incorporate particles having the right hydrodynamical variables in its vicinity. It will in the other hand be weakened by particles whose trajectory clash with his and of variables too different for him to incorporate. "Weakening" for the agent means shrinking and releasing aggregated particles.
His social environment will be the other agents. When near other agents of opposite rotation direction, it will transform its outer layer in entities of a lower level, be

them particles of smaller agents. Changing thus its granularity, it allows for a finer management or the interaction. When near agents rotating in the same direction, it will try to aggregate to form a bigger vortex, as described in the next paragraph.

Detecting and handling advanced structures

The vortex agents themselves can, if of a similar vorticity, aggregate with one another to constitute vortex of a higher level. These vortex will be reified by agents very similar to those described earlier, if of a somewhat higher volume. This process of automated generation is therefore recursive, property relying on fractal properties of vortexes in fluid flows.

IMPLEMENTATION

Basic choices

The programming of this system has started. The chosen language is Java, because of its relative ease of use for prototypes, of its good integration in many agent platforms and of its numerous implementations of CORBA.

The agents either are based on the MADKIT platform or cater to the specific needs of the system.

What's been done so far

Implementing the basis entities

The flow of particles as outlined before is coded. The flow is realistic and its computation fast.

Managing the simulation space

A system of NSP trees of grid–like structures has been implemented. Each of these grids is in charge of a part of the simulation space.

The leaves of this tree of grids are in charge of the simulation entities that have a spatial localisation and whose said localisation falls within the part of the simulation space they are in charge of. The nodes of the tree are grids divided in *n* sub–grids. This division is dynamic and depends on the number of entities that fall within the borders of the grid : above a given threshold, the grid gives up its leaf status to turn into a node pointing to *n* leaves. The node knows the size of the total population of its leaves; if this size falls below another threshold, the leaves are destroyed, the node resumes its leaf status and takes responsibility of the entities formerly managed by its children.

The root grid encompasses the whole space.

This tree allows the simulation entities a fast access to a knowledge of their surroundings : the time it takes for knowing who belongs to the same grid as a given entity is directly link to the height of the tree, which is in most practical cases relatively limited. Knowing who belongs to neighbouring grids takes a bit more time, but it is still negligible if compared to the computing of the movement of the particles themselves. It was not the case before the grids were implemented.

Finally, the grids will now allow the use of *fast vortex computation methods* [REF].

Detecting structures

The detection method using Delaunay's triangulation has been implemented. It has been detailed in [Bertelle et al 2000a]. It is efficient but slow. In our tests it detected all the vortex it should and none more. It is a $O(n^2)$ algorithm. The program takes a couple of minutes to analyse 1000 fluid particles.

What will come soon

- Fast vortex methods : the computing of the particles trajectories is currently of a $O(n^2)$ order. It can be improved to $O(n \log n)$ through the use of the grids and of these methods.
- Reification of the detected structures in agents, thus starting the recursivity of our structure detections.

CONCLUSION

Transforming emerging structures into agents will allow the following development

Sharing the computing through CORBA

Each agent will know its immediate surroundings, thanks among other to the tree of grids, and with whom it interacts most. This knowledge will be used to compute a distribution of the computing where the simulation entities that need more communications between one another will be gathered together in the same node, as outlined in [Bertelle et al 2000b].

Adapting the time scale

The simulation entities being of various size, they do not all need to act at the same time or even as often. A heuristic is that the larger the structure reified, the more stable it is, which means a lesser sensitivity to small changes.

Nonetheless, close to a turbulence , this stability will be questioned, and the entity will have to update more often.

All in all, the entities will have a basic timeline linked to its size, and this timeline will be altered to match those of the entities it interacts most with.

REFERENCES

R. J. Abrahart 1999, « Using pruning algorithms and genetics algorithms to optimise network architectures and forecasting inputs in a neural network rainfall–rainoff model » *Journal of Hydroinformatics*, 01.2

C. Bertelle, D. Olivier, P. Tranouez and V. Jay. 2000a. «Agent–Based Simulation of Water Flow for Environmental Modelling in Estuaries » In *Proceedings of the Agent–Based Simulation Workshop 2000* (Passau, Germany, May). SCS, 115–120.

C. Bertelle, D. Olivier, V. Jay, P. Tranouez and A. Cardon. 2000b. «A multi–agent system integrating vortex methods for fluid flow computation », In *Proceedings of the 16th IMACS World Congress* (Lausanne, Switzerland, August)

J.P. Choquin and S. Huberson. 1988. « Particle simulation of viscous flow » *Journal of Computer and Fluid*, n°2 : 397–410.

A.J. Chorin. 1973. « Numerical study of slightly viscous fluid » *Journal of Fluid Mechanic* 57 : 785–796.

S. Frontier D. Pichod–Viale 1998. *Ecosystèmes Structure – Fonctionnement – Evolution* Dunod Paris

A. Leonard. 1980. « Vortex methods for flow simulation » *Journal of Computational Physic* 37 : 289–335.

A. E. Mynett. 1999. « Hydroinformatics and its applications at Delft Institute » *Journal of Hydroinformatics* 01.2.

D. Servat. 2000. « Modelisation de dynamiques de flux par agents », PhD Paris VI University (France).