

TRAINING GAMES AND GIS

Marcelo G. Metello, Marco A. Casanova
Departamento de Informática – PUC
Rua Marquês de S. Vicente, 225 – Rio de Janeiro, Brasil
{metello|casanova}@tecgraf.puc-rio.br

ABSTRACT

This paper first lists the basic requirements of *geospatial training games*, a class of serious games designed for computer-based training that requires the use of spatio-temporal data. Then, it addresses the implications such requirements have for the development of GIS systems.

1. INTRODUCTION

The widespread adoption of computer games for entertainment purposes, the continuous decrease of hardware cost and the success in military simulations made computer game technologies attractive to some “serious” industries, such as medicine, architecture, education, urban planning, and government (Smith 2007). The term *serious games* (Susi, Johannesson and Backlund 2007) was adopted to denote games used for non-entertainment purposes.

The development of serious games poses specific challenges, since their requirements differ from those designed for entertainment purposes. Serious games usually require simulation models that reproduce certain aspects of reality. By contrast, entertainment games have much more freedom to create and modify their own reality, which is convenient when developers face technical limitations.

We will use the term *geospatial training games* to denote serious games that are designed for computer-based training and that require the use of spatio-temporal data representing real geographic objects. Emergency simulation games, such as those planned to support the InfoPAE system (Carvalho et al. 2001), are a typical example of this class of games.

2. BASIC REQUIREMENTS FOR TRAINING GAMES

This section lists a set of requirements for training games that are not fundamental to other classes of computer games. We note that the four requirements listed below are not entirely orthogonal.

Realistic user experience: Training games require simulation with a minimum acceptable degree of realism.

Interoperability with existing systems: Training games require the ability to use real data, typically stored in existing databases. This requirement may be interpreted as a subrequirement of the first requirement.

Time flow control: Training games require control over the time flow, that is, the ability to stop, accelerate and go back in time.

Player performance evaluation: Training games require player evaluation as part of the learning process (Borodzicz and van Haperen 2002). In many cases, this requirement means that the system should be able to play back a simulation for evaluation purposes, which involves saving simulation data.

3. A BRIEF DISCUSSION ON THE REQUIREMENTS

3.1 Realistic User Experience and GIS

The rendering of spatial data on the computer screen should have enough similarity with the spatial objects depicted so as to help users reason spatially about the real world. Likewise, computer animation of time-varying data should help users reason about dynamic phenomena. Furthermore, just like spatial data, animations may adopt different time scales without impairing how users assimilate the information. For example, if an animation shows in some seconds the changes in land use of a particular region during a period of some years, it must still give the user a sense of what is happening with the land.

However, GIS systems have been traditionally more concerned with displaying spatial data than temporal data. Only recently the major GIS systems, other than military and flight simulators, have started to show some concern with time and user experience similar to those of the gaming industry.

3.2 Interoperability with Existing Systems and DBMS

The technology developed for computer games focuses on displaying data in real time, with a continuous attention on performance to keep an acceptable frame rate for a better user experience. In particular, training games require displaying updated data, typically stored in databases, with a considerably high frequency and potentially to multiple clients.

However, traditional DBMS (database management systems) may not be able to provide the throughput training games require. Therefore, appropriate indexing and caching mechanisms are necessary in order to provide game-compatible frame rates (Metello et al. 2007).

3.3 Time Flow Control and the Storage of Simulation Data

The requirement of playing back simulations also raises problems to DBMS, if it implies that simulation data be recorded. Indeed, storing simulation data must be done without slowing down the simulation itself and without causing the databases to grow too large.

Furthermore, storing simulation data may generate a massive number of update requests. This poses an interesting problem on how much indexing should be used. Indexes can speed up queries, but they necessarily slow down update requests. In particular, spatial feature classification can play an important role with respect to optimizing the amount of indexing, if it can help differentiate features with high update frequencies from static ones or from those with low update frequencies.

Storing simulation data may also force the database to grow beyond acceptable limits. This reinforces the importance of research that seeks alternative ways of storing spatio-temporal data other than storing a new copy every time an update operation is executed.

4. REPRESENTING SIMULATION DATA

The techniques to store simulation data vary from storing a copy of the state of each object at each point in time to storing just the initial state and a complete description of the behavior of each object. In the latter case, it is also necessary to save all external input given to the simulation that possibly alters the state of the world.

The trivial solution to store a copy of the state of each object at each point in time is unacceptably slow and expensive in terms of space, which opens room for an interesting research topic: how to minimize I/O

when recording simulation data in real time. In the context of geospatial training games, the problem is to find a way to represent spatio-temporal data such that it can be updated with as few I/O operations as possible during the simulation.

4.1 Storing Moving Objects

A simple way to store changing objects is to generate a sequence of pairs of the form $\langle t_i, v_i \rangle$, where v_i is the value of the object in the time interval $[t_i, t_{i+1})$. A new pair must be generated whenever the object changes beyond a given limit, as compared to the last generated pair. This representation has at least two drawbacks, which can be a serious problem in the case of large spatial objects: (1) the value has to be rewritten in each pair, even if the changes were small; (2) a new tuple has to be generated every time the value changes beyond a given error tolerance. Furthermore, there is no optimization when the value changes according to some pattern.

The research about moving objects (Wolfson et al. 1998) attempts at improving this situation by finding alternative ways to represent spatio-temporal objects. One such way is to use dynamic attributes. The idea of dynamic attributes is to store triples of the form $\langle t, p, f \rangle$, where t is again a timestamp, p represents the position of a moving object and f , called a *moving function*, models how the position will change until the next update.

Representing dynamic attributes by an initial value and a function is clearly helpful. Most research in this area is concentrated on simple functions, such as linear and polynomial (Erwig et al. 1998), which can be represented by a small number of parameters. The use of simple functions in a fixed format allows the use of specific indexing techniques (Wolfson et al. 1998).

Another interesting way to represent moving objects is to use a multidimensional spatio-temporal space (Siebeck 2004). As an example, a moving point in 2D space is represented as a line in a 3D space, where the third dimension represents time. This has the potential advantage of translating spatio-temporal queries into simpler spatial queries.

4.2 Storing Complex Behavior

Simulations and other dynamic models in general often involve objects with complex behavior which can be too complex to be expressed as mathematical functions. In addition, the behavior of many objects is

likely to depend on the state of other objects involved in the simulation. This suggests that it may be an interesting alternative to store complex behavior descriptions, if simulation recording is to be optimized. It is worth mentioning that complex and real time simulations are likely to use specialized spatial indexes structures, instead of the more general alternatives provided by DBMS's.

5. AN EXAMPLE FROM EMERGENCY SIMULATIONS

This section describes a simple example to clarify the idea of storing specific behavior of an object in an emergency simulation system.

An oil leak that contaminates a water body usually requires placing contention barriers to keep the oil from reaching the coast. At the moment an order is given to place a barrier, the team already knows where the barrier should be put and how long it should take. Therefore, it may be a good idea to store the final expected barrier location as a polyline along with a function that will give the expected percentage of the barrier that has already been placed at a given time. If everything happens as planned in the simulation, it would need just one I/O operation for the whole barrier placement operation. Only in the case of exceptions (e.g. the order to place the barrier was cancelled because the wind has changed direction) more I/O's would be necessary.

However, modelling and monitoring the oil dispersion in the water is far more complex. In fact, in a real application, data representing possible oil dispersion patterns accounts for over 90% of the database.

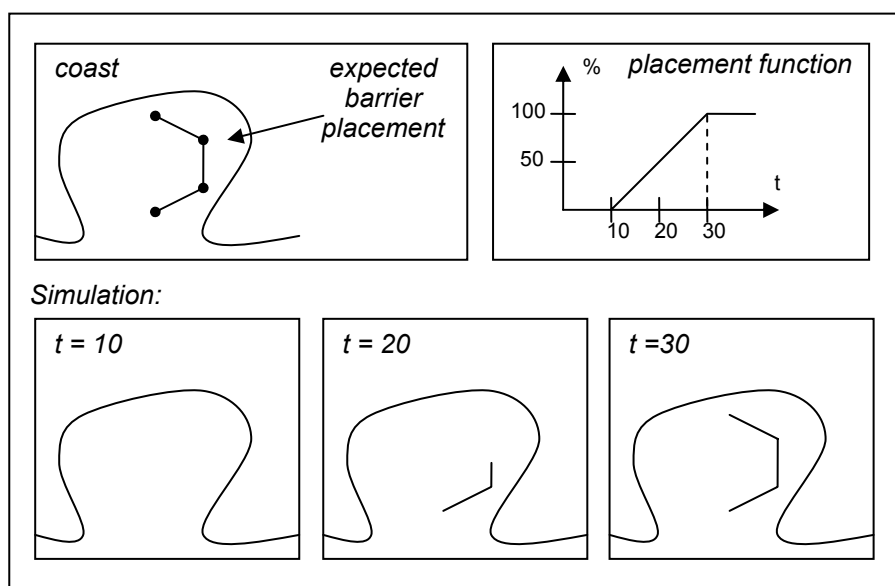


Fig. 1. Barrier positioning example.

6. CONCLUSION

We addressed in this paper some of the challenges geospatial training games pose for the development of GIS and spatio-temporal DBMS. Among the problems, we focused on the storage of simulation data, as a variant of the general problem of representing complex object behavior. The major motivation comes from emergency simulation games, as part of the effort to design emergency response information systems.

REFERENCES

- Borodzicz, E. and van Haperen, K. (2002) "Individual and Group Learning in Crisis Simulations". *Journal of Contingencies and Crisis Management* 10 (3), 139–147 doi:10.1111/1468-5973.00190
- Carvalho, M.T.; Freire, J.; Casanova, M.A. (2001) "The Architecture of an Emergency Plan Deployment System". Proc. III Workshop Brasileiro de GeoInformática, Rio de Janeiro, Brasil, Oct. 2001.
- Erwig, M., Guting, R.H., Schneider, M. and Vazirgianis, M. (1998). "Spatio-temporal Data Types: An Approach to Modeling and Querying Moving Objects in Databases". Proc. ACM GIS Symposium.
- Metello, M, et al. (2007). "Continuous Interaction with TDK Improving the User Experience in Terralib". Proc. IX Brazilian Symposium on GeoInformatics.
- Siebeck, J. (2004). "Concepts for the Representation, Storage, and Retrieval of Spatio-Temporal Objects in 3D/4D Geo-Information-Systems". Rheinische Friedrich-Wilhelms-Universität Bonn, Diss.
- Smith, R. (2007). "Game Impact Theory: Five Forces That Are Driving the Adoption of Game Technologies within Multiple Established Industries". *Games and Society Yearbook*.
- Susi, T., Johannesson, M., Backlund, P. (2007) "Serious Games – An Overview". Technical Report HS-IKI-TR-07-001, School of Humanities and Informatics, University of Skövde, Sweden
- Wolfson, O., Xu, B., Jiang, L., Chamberlain, S. (1998) "Moving Objects Databases: Issues and Solutions". SSDBM, p. 111, 10th International Conference on Scientific and Statistical Database Management