VII Simpósio Brasileiro de Geoinformática, Campos do Jordão, Brasil, 20-23 novembro 2005, INPE, p. 327-339.

VisFluid: An Analysis Tool for Fluid Flow in Porous Media

Carlos M. de Lima[1], Luiz M. G. Gonçalves[1], Cristiana Bentes[2], Ricardo Farias[3]

[1]Laboratório de Realidade Virtual
Universidade Federal do Rio Grande do Norte - Natal, Brasil

[2]Geomática - Departamento de Engenharia de Sistemas e Computação
Universidade do Estado do Rio de Janeiro - Rio de Janeiro, Brasil

[3]COPPE - Sistemas
Universidade Federal do Rio de Janeiro - Rio de Janeiro, Brasil

{cmagno,lmarcos}@dca.ufrn.br,cris@eng.uerj.br,rfarias@cos.ufrj.br

Abstract. *The study of fluid flow process through porous media has attracted much attention in geographical information systems due to its importance in petroleum and environmental applications, such as: groundwater assessment, soil hydrology and pollution analysis and oil exploration and recovery. In this work we propose an interactive tool, named VisFluid, that uses a simulation engine based on dynamic percolation method to model the fluid flow in porous media, and a visualization engine based on direct volume rendering techniques to provide images and animations of the tri-dimensional data generated. Our main goal here is to develop a complete tool for the study of fluid flow in geographical information systems that integrates the percolation simulation with the visualization, avoiding data conversion and I/O requirements. We apply our tool to the oil exploration and recovery application, obtaining satisfactory results.*

1. Introduction

Recently, a great deal of interest has been focused on the investigation of fluid penetration phenomena into porous media. The understanding of fluid flow process through porous media has attracted much attention in geographical information systems due to its importance in petroleum and environmental applications [27], such as: (i) providing an accurate and reliable groundwater resource information, in order to assess the quantity and quality of groundwater potential zones; (ii) examining the soil hydrology and pollution, in order to analyze the nitrate and pesticide loadings in the soil and the expansion of irrigation activities; and (iii) exploring and recovering of oil, in order to potentially improve the correct determination of the well position and the insertion of water.

One of the most popular models of fluid flow in porous media is the percolation theory. Percolation theory is successfully applied to model the connectivity phenomena, based on the capillary behavior of porous materials. A porous medium can be considered as a system of directly connected capillary tubes. To study the clustering behavior of particles in these systems, the mathematical percolation theory developed in statistical physics [30] was applied by several works [4, 9, 18].

The flow of fluid is a truly 3D phenomenon and for a better understanding of its simulation, percolation results must be visualized as images or animations. Hence, the investigation of percolation simulation results depends on the use of advanced scientific visualization techniques. Scientific visualization has been widely recognized as useful for analyzing and interpreting large-scale volumetric datasets. Due to the vast amounts of data generated, researchers often prefer to view the results in their image representation, than analyzing them in their raw format, or as long lists of numbers. Although a lot of powerful and sophisticated visualization tools have been developed, most of traditional tools and techniques for volume visualization in geographical information systems give only results based on isosurface visualization of 3D datasets [2, 11, 12, 19].

In this work we propose an interactive and helpful tool, named VisFluid, that uses a simulation engine based on dynamic percolation to model the fluid flow in porous media, and a visualization engine based on direct volume rendering techniques to provide images and animations of the tri-dimensional data generated. The use of direct volume rendering technique allows the visualization of datasets, not only of their surface information, but also of their internal structures and details. In the particular case of dynamic percolation, the simulation allows the user to follow the evolution of clusters formation and the interactions between them. As these interactions occur inside the volumetric data, their visualization is only possible with the application of volume rendering, that adds more realism by showing details from the interior of the data. Traditional visualization softwares for geology, like gOcad [3], for example, cannot provide this kind of information, since gOcad is based only on isosurface visualization of 3D datasets. Figure 1 shows an example of gOcad visualization of percolation data generated in [6].

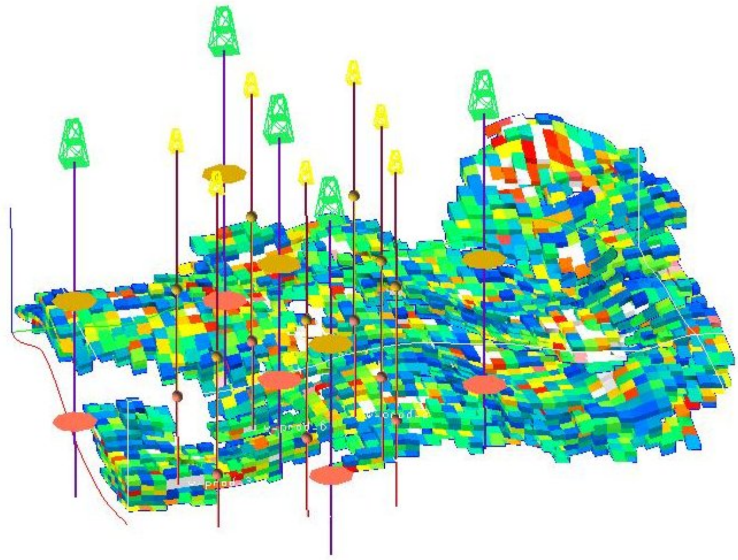

Figure 1. Volume visualization of percolation clusters with gOcad.

Our main goal here is to develop a complete tool for the study of fluid flow in geographical information systems that generates both simulation results of dynamic percolation on porous media, and the volumetric images of such results. By its more complex nature, volume rendering techniques require higher computing power, so we also include in the VisFluid tool the possibility of visualizing only surface data information using OpenGL [29], that is the industry's foundation for high-performance graphics . This gives the user the possibility of choosing between the delivery of poorer but faster images, or a volumetric and richer in details, but slower viewing of the percolation volume data.

We apply our tool in a dynamic percolation simulation for oil exploration and recovery application, and obtained satisfactory results. Our tool generated images of the interior of the percolation data, bringing valuable contributions for the interpretation and cluster formation process. These contributions can potentially improve the correct determination of the position to insert water in the oil field.

The remainder of this paper is organized as follows. In the next section we relate our work to other geovisualization tools. In Section 3., we briefly describe the percolation model and simulation. In Section 4., it is presented some background on volume rendering. In Section 5., we describe our tool designed to visualize dynamic percolation fluid flow data generated by simulation. In Section 6., we present the results of our experiments. Finally, in Section 7., we present our conclusions and proposals for future work.

2. Related Work

Geovisualization has become an important research area because it provides a better exploration, analysis, synthesis, and presentation of georeferenced information in order to facilitate the decision making process. Visualization of large scientific data for earth sciences [16], oceanography [5, 23, 26], meteorology [32], climate modeling [22], and for environmental decision making [7, 25] has been around for more than a decade. However, the area of fluid flow in groundwater assessment, soil hydrology and oil exploration and recovery has received little attention by the scientists.

There are several works in the area of mathematical models for fluid flow in porous media. They studied the effect of pore structure on relative permeability and capillary pressure [18]; drainage and imbibition [17, 21]; hydraulic conductivity on partially saturated fractured porous [24]; and electrical conductivity of three dimensional pore networks [10]. Nevertheless, all these works deal only with the mathematical models, not with the visualization of the results.

In terms of visual images of porous media, the work by George and Kovscek [13] investigates pore-scale flow phenomena in micromodels, varying pressure and temperature, and in [28], Sagar and Castanier present results of a pore-level visualization study of foam stability in the presence of oil. Their visual investigation, however, lies on high resolution microscopes images, not on three-dimensional images of simulation results. The work by Tomov *et al* [31] follows an opposite way, and uses the graphics hardware for non-graphics numerical simulations like percolation model. Their focus was on the parallel use of programmable GPUs, not on the generation of 3D images. The work by Adler and Lev [1] is the most closely work related to ours. They implemented a percolation model called Bootstrap Percolation and provides visualizations using the AViz software, originally designed for atomistic simulations. The images obtained by Adler and Lev, however, could only show the compact nature of the clusters. They do not provide volume rendering on the porous media system.

3. Percolation Theory

Percolation theory deals with fluid flow (or any other similar process) in random media. It studies the connectivity and transport in geometrically complex systems [30]. Given a square lattice, which represents the region of interest, each site has a random number *t*, that ranges between 0 and 1, associated with it. A site is considered is considered occupied if its value is below a given threshold p, or empty otherwise. Occupied and empty sites may stand for different physical properties. Two occupied sites belong to the same cluster if they are connected by a path of nearest neighbor occupied sites, and a fluid can flow between them. At low p values, the occupied sites are either isolated or form small clusters of nearest neighbor sites, see Figure 2a. At large p values, on the other hand, many sites of the lattice will be occupied, and the occupied sites will form a large cluster that extends throughout the entire lattice, see Figure 2c. In the extreme case, where $p = 1$, all sites are connected to some maximum number of neighboring sites. A large cluster, that connects the system from one side to the other, is called a spanning cluster, as it spans the entire lattice. As p increases, the probability that there is a spanning cluster also increases. There must be a particular intermediate value of *p* for which a spanning cluster will be first formed. Such value of occupancy probability is known as the percolation threshold (p_c), and plays a fundamental role in the percolation theory. The percolation threshold strongly depends on the type and dimension of the grid used, see Figure 3 [14].

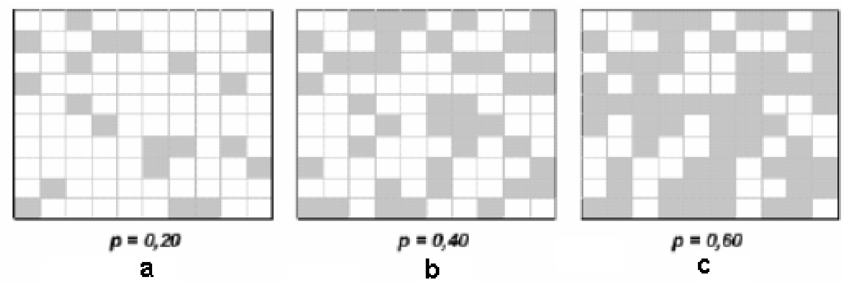

Figure 2. Site percolation scheme.

Percolation		Lattice
bond - p_c	site - p_c	
$2\sin\pi/18$	$1/2$	Triangle
$1/2$	0.5927	Square
0.2488	0.3116	Cubic

Figure 3. Percolation thresholds.

There are two types of percolation: bond percolation and site percolation. Bond percolation considers the lattice edges as the relevant entities, while site percolation considers the lattice vertices as the basic elements. In the standard percolation models, sites or bonds are static. To correlate physical properties as functions of the probability of percolation *p*, quit often, uncorrelated samples are used for each value of *p*, and averages have to be made over many samples. Some methods and algorithms have been developed to achieve better representation, which expanded the application of percolation theory. Most part of these methods are concerned to the variations of the static percolating. Between them we can find spreading percolation [20], invasion percolation [33], stirred percolation [15].

3.1. Dynamic Percolation

Freitas [6] proposed the model called dynamic percolation where the percolation process is considered time-dependent. The probability p of having one site occupied increases linearly with time. In this way, it is possible to investigate the evolution of physical properties and how the size of the clusters and the number of clusters evolve in time. In standard percolation, clusters of connected sites are static, as sites are occupied with a fixed probability. Hence, dynamic percolation generates more scenarios, than standard percolation.

The dynamic percolation method uses a d-dimensional lattice as input. This lattice contains the physical properties data from the porous media being studied. The method can be applied to site percolation, bond percolation or site and bond percolation. The sites (or bonds) are sorted by their values, and for each step of time t, the non-occupied site with the smallest value is set as occupied. When a site is occupied, if any of its neighbors is also occupied, they are clustered together. Each new cluster receives the label of the larger cluster from which it was formed.

4. Volume Visualization

There are two main approaches for the visualization of volumetric dataset:

- isosurfacing - extracts a surface through the volume;
- volume rendering - shows the interior of the volume, enabling the identification of its inner regions.

The first approach is called *indirect volume rendering*. Indirect volume rendering converts the 3D data into surface elements which are then rendered to form an image. The other approach is called *direct volume rendering* and the information from the entire data set is incorporated into the resulting image.

In volume rendering, the data is represented by a grid of voxels, where each voxel has associated one or more values quantifying some measured or calculated property of the original object, such as transparency, luminosity, density, or flow velocity. Volume rendering techniques sum the contributions of each voxel to compute the final color of a pixel. The faces are considered to be semi-transparent, allowing the ray of light to pass from voxel to voxel, instead of stopping at its surface, preserving this way the integrity of the original data throughout the visualization process. In surface rendering, some interesting features of volumetric data could be lost, embedded in the middle, hidden by outer opaque surfaces. Since the entire data set is preserved in volume rendering, internal structures and details may be viewed.

There has been a lot of works on volume rendering techniques over the past few years, leading to several competing algorithms. The two most common approaches are: (1) ray-casting and (2) cell projection. Ray-casting algorithms are usually called image-space methods, in which rays are cast from the viewpoint through every pixel of the image, computation of the contribution for the pixel color and opacity is made as the ray intersects faces of the data. The ray stops when it reaches full opacity or when it leaves the volume. Cell projection algorithms, on the other hand, generate the image from the object space to the image space and are called object-space algorithms. The projection requires that the cells are first sorted in visibility ordering and then their projection on the screen is performed by a scan conversion process, which results in information equivalent to the intersections obtained by the ray-cast. The same way, these intersections are used to compose the color and opacity for each pixel in the final image. The great advantage of projective methods is that they provide exact results without having to treat degenerate cases which occur in the ray-cast. We use in VisFluid tool the ZSweep volume rendering algorithm [8], because it is a fast and memory efficient algorithm, and the most recently all-software cell projection rendering algorithm. Furthermore, ZSweep handles volumetric and irregular grids data, which are more appropriate to represent seismic data or data generated by fluid dynamic percolation simulation.

4.1. ZSweep Volume Rendering Algorithm

The ZSweep algorithm by Farias *et.al.* [8] is a direct volume rendering algorithm based on the sweeping paradigm. ZSweep was designed to combine accuracy and simplicity with speed and memory efficiency, built over the success of prior sweep approaches. The main idea of ZSweep algorithm is the sweeping of the data with a plane parallel to the viewing plane, in order of increasing z. The vertices are sorted by the z-coordinate, using a heap data structure, and while the plane is swept, ZSweep projects the faces that are incident to the vertices intersected by the plane. During face projection, ZSweep computes the intersection of the ray emanating from each pixel, and store their z-value, and other auxiliary information, in sorted order in a list of intersections for the given pixel, called pixel list. To achieve memory efficiency, ZSweep uses a mechanism called early composition. The composition of the intersections in a pixel list is performed as the *target-Z* is reached. The *target-Z* represents the maximum z coordinate among the vertices adjacent to the first vertex encountered by the sweeping plane.

5. The VisFluid Tool

The VisFluid tool was developed in C++ on top of Linux and IRIX operating systems. The front-end interface was developed using QT API, with widgets and others tools. VisFluid is composed by two modules: the percolation engine and the visualization engine. The percolation module is responsible for the simulation of the fluid dynamic percolation. The visualization module provides three-dimensional images of the data generated by the simulation, using two different visualization techniques: surface rendering and volume rendering. Surface rendering is implemented based on OpenGL and provides poor, but faster images. Volume rendering is implemented based on the ZSweep algorithm and provides volumetric and more complete images, but at a slower speed.

Our idea in VisFluid is to integrate these two modules in order to avoid intermediate files and data conversion. Usually these two processes, percolation simulation and volumetric visualization, are done by different tools requiring the conversion of the data into the dataset format suitable for the rendering algorithm.

5.1. Percolation Simulation

The percolation simulation module uses the dynamic percolation model explained in section 3.1.. We choose this model because it is simple, and allows the study of the evolution of clusters distribution as a function of time. Another important feature of dynamic percolation is that it has a low computational cost. As dynamic percolation varies p with

t, it produces much more scenarios than standard methods, but without requiring more computational power.

We have implemented the method proposed by by Freitas [6] with two improvements. The first improvement was on the lattice sorting, we introduced a heap structure in order to improve sort performance. The heap also reduces the number of visits in the lattice sites, when different sites have the same permeability value. Another feature we introduced in the simulation was the format of the output data. As the final goal is to visualize the clusters, we have changed the structure of the output data, in order to conform with visualization tools input. In this way, we avoid data conversion between the simulation and visualization module.

5.2. Visualization

The visualization module divided in two modules: surface visualization and volumetric visualization modules. The surface visualization module uses OpenGL directives from the three-dimensional data generated by the simulation to create surface images of the data. These images are useful when the user does not need details of the interior of the data and wants the images results in critical time. The surface rendering module provides an animation of all t_i instants of percolation, where the user can observe the cluster formation evolution.

To better analyze the results of the percolation simulation, however, VisFluid also provides volumetric visualization of the data at a slower rate. This volumetric visualization is accomplished by the a modified version of the ZSweep rendering algorithm. The idea is to have the user deciding whether to observe just the evolution process or the inner properties and correlation of the clusters.

The integration between the simulation module and the volume rendering module is done by the irregular grid data file, the grid unifies the simulation output and the volumetric visualization input, avoiding any kind of data conversion. As the dynamic percolation studies a time-dependent problem, it increases the amount of data to be rendered. Therefore, the integration between the simulation and the volume rendering module is crucial for the visualization performance.

6. Experimental Results

ZSweepVR was tested on a database from a mature field. This database was generated by the IMEX flow simulator at CMG through geostatistic studies including data interpolation from wells profile and geophysical parameters interpretation, such as permeability, porosity, and water and oil saturation. The database is a $36 \times 51 \times 27$ grid, where each cell corresponds to an region of 25m $\times$ 25m $\times$ 0.5m. It was used as the input to the simulation sever and we analyze the images generated by the two different modules of the visualization server: surface rendering and volume rendering. The main goal of our experiments was to investigate how the properties are distributed over the field and how these properties contribute for oil exploration and recovery.

Figure 4 shows the interface of VisFluid. As the results of dynamic percolation are time-dependent, the interface allows the user to play different instants of percolation results. The widgets implement forward and backward operations, with the option of choosing the percolation instant of time.

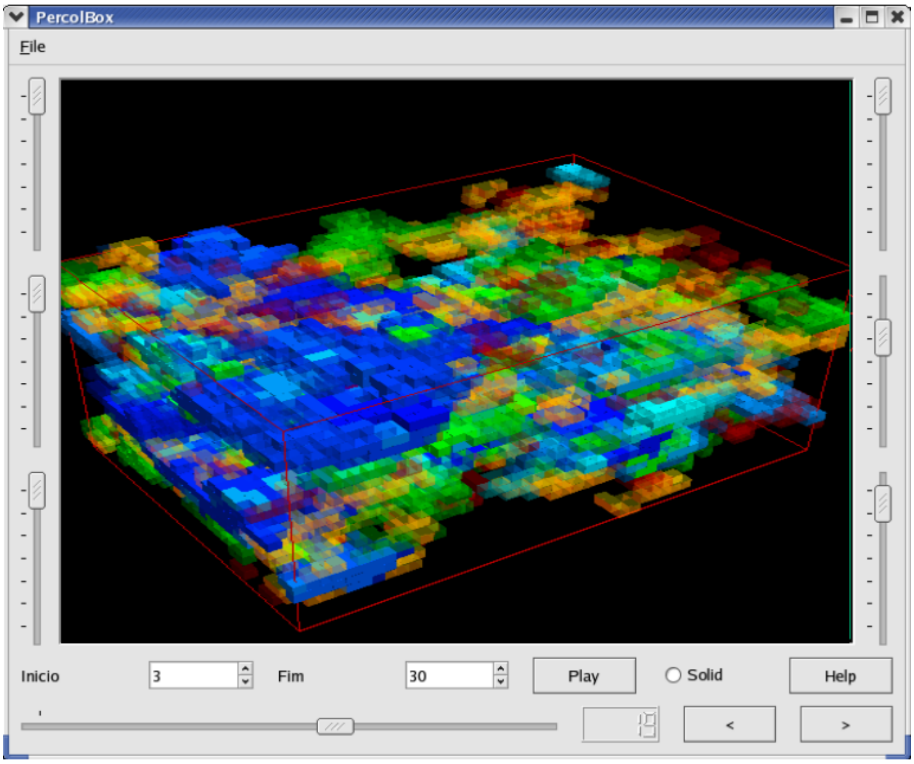

Figure 4. VisFluid interface.

In Figure 5 we show the results generated by the surface visualization module that comprises images from different moments of the percolation. With these images we can observe cluster formation and we can also observe local and individual information about this evolution process. Since reservoirs are complex structures, containing geological heterogeneity ranging from centimeters to kilometers, this kind of visualization allows the proper analysis of the the percolation evolution, improving the security decisions regarding oil recovery.

The images in Figure 6, were generated from the last image of Figure 5, using the OpenGL transparency. Using transparency, the user can visualize the correlation between clusters in the evolution process. Regions with less percolation sites are made more transparent, highlighting the spanning clusters.

The volumetric visualization module of VisFluid generates much more detailed images than the ones created by the surface visualization module. Figure 7 shows the images corresponding to the same percolation moments shown in Figure 5, but from a different point of view. As the time advances, frontal clusters occludes the ones in the back. Playing with the transparency in the volumetric visualization module, the back clusters are still visible. Furthermore, as the percolation evolutes, it is possible to better visualize the sites which connect the clusters.

7. Conclusions

In this paper we presented the VisFluid tool for studying fluid flow in porous media. Our tool can be used in different areas of geographic information systems, such as: groundwater assessment, soil hydrology and pollution analysis and oil exploration and recovery.

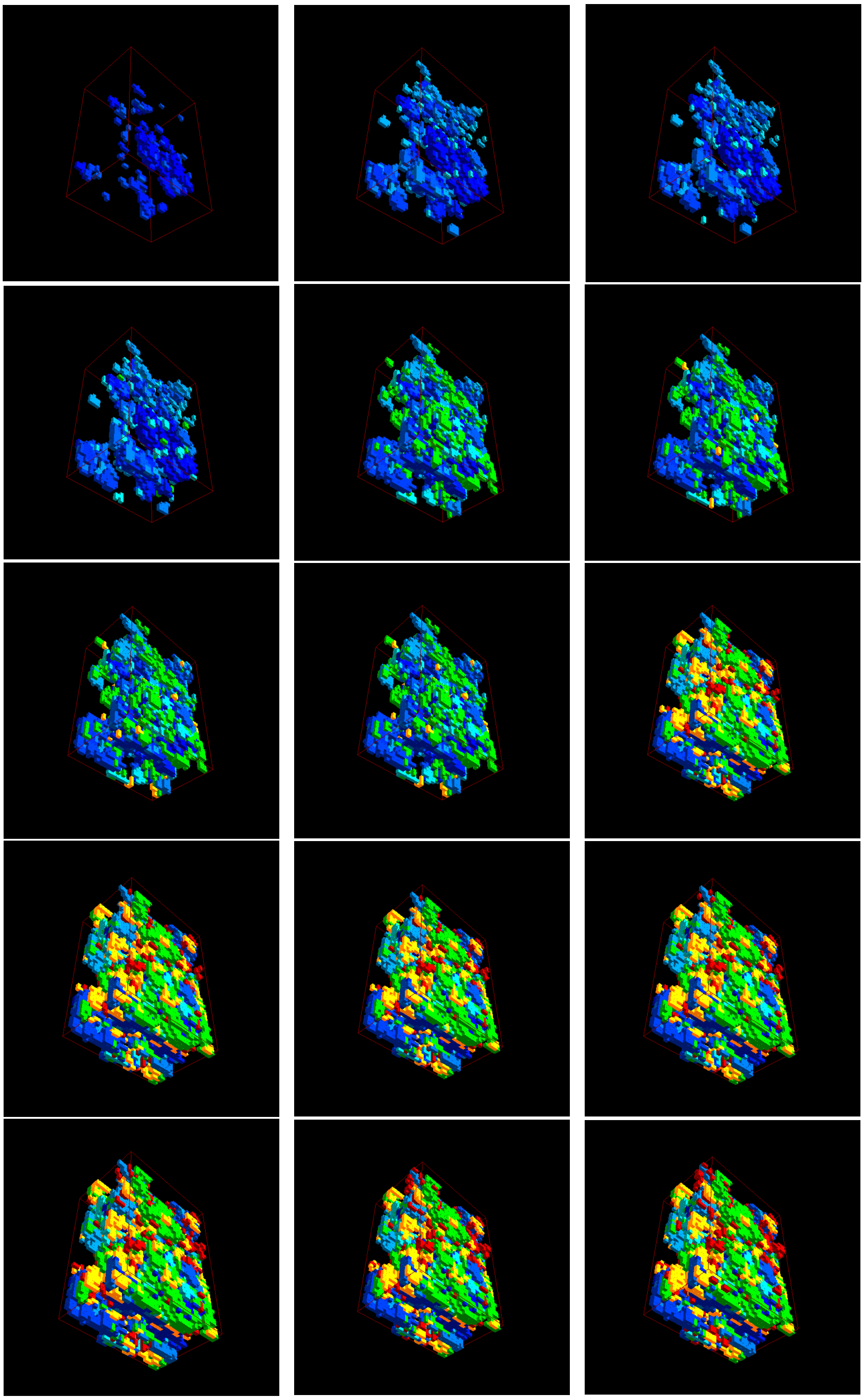

Figure 5. Images of cluster formation generated by the surface visualization module.

The main idea of VisFluid is to provide a simulation environment that uses percolation theory in order to model the fluid transport and to provide a visual interpretation of the data generated by the simulation.

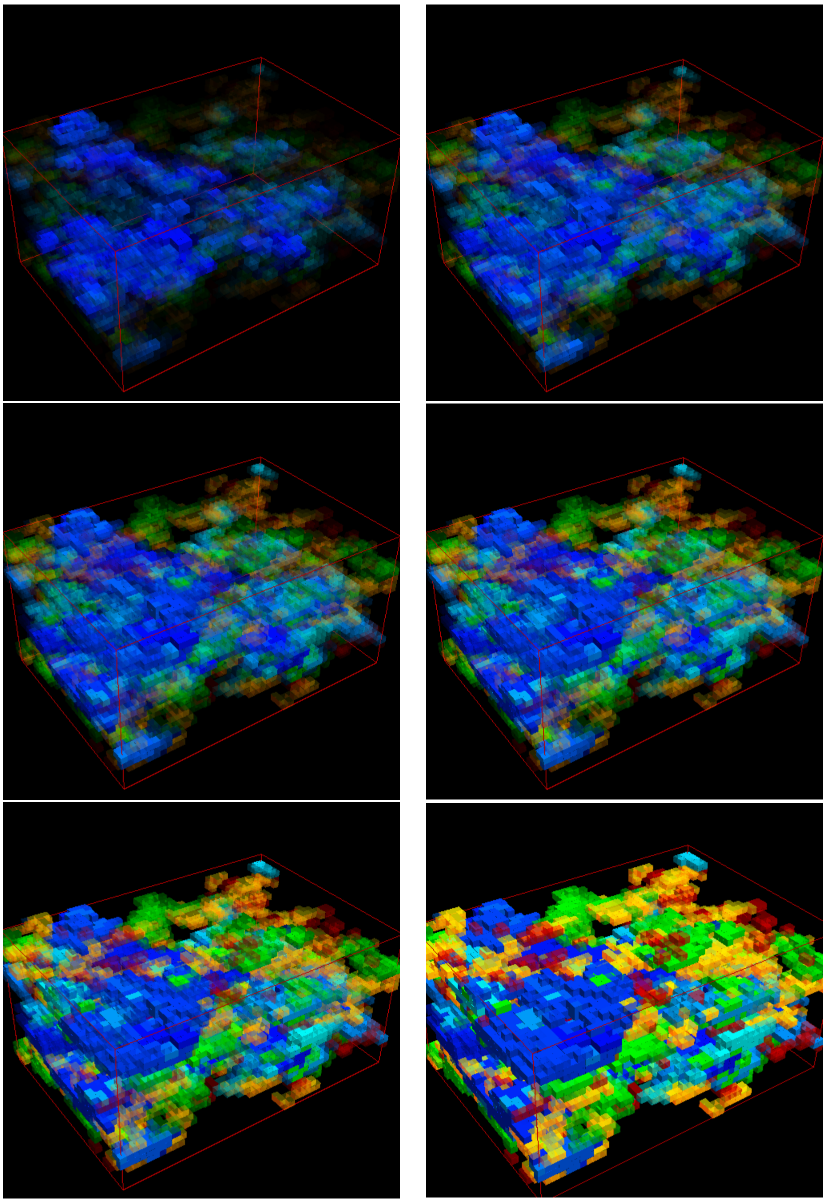

Figure 6. Using transparency to highlight the spanning clusters.

The VisFluid tool is composed by two modules: the percolation engine and the visualization engine. The visualization module provides three-dimensional images of the data generated by the simulation, using two different visualization techniques: surface rendering and volume rendering. The two modules are integrated, avoiding any kind of data conversion and I/O requirements between the modules, in order to improve performance.

We apply our tool in a oil exploration and recovery application, and obtained satisfactory results. Our tool generated dynamic percolation simulation results and images of the percolation data, bringing valuable contributions for the interpretation and cluster formation process. These contributions can potentially improve the correct determination of the position to insert water in the oil field.

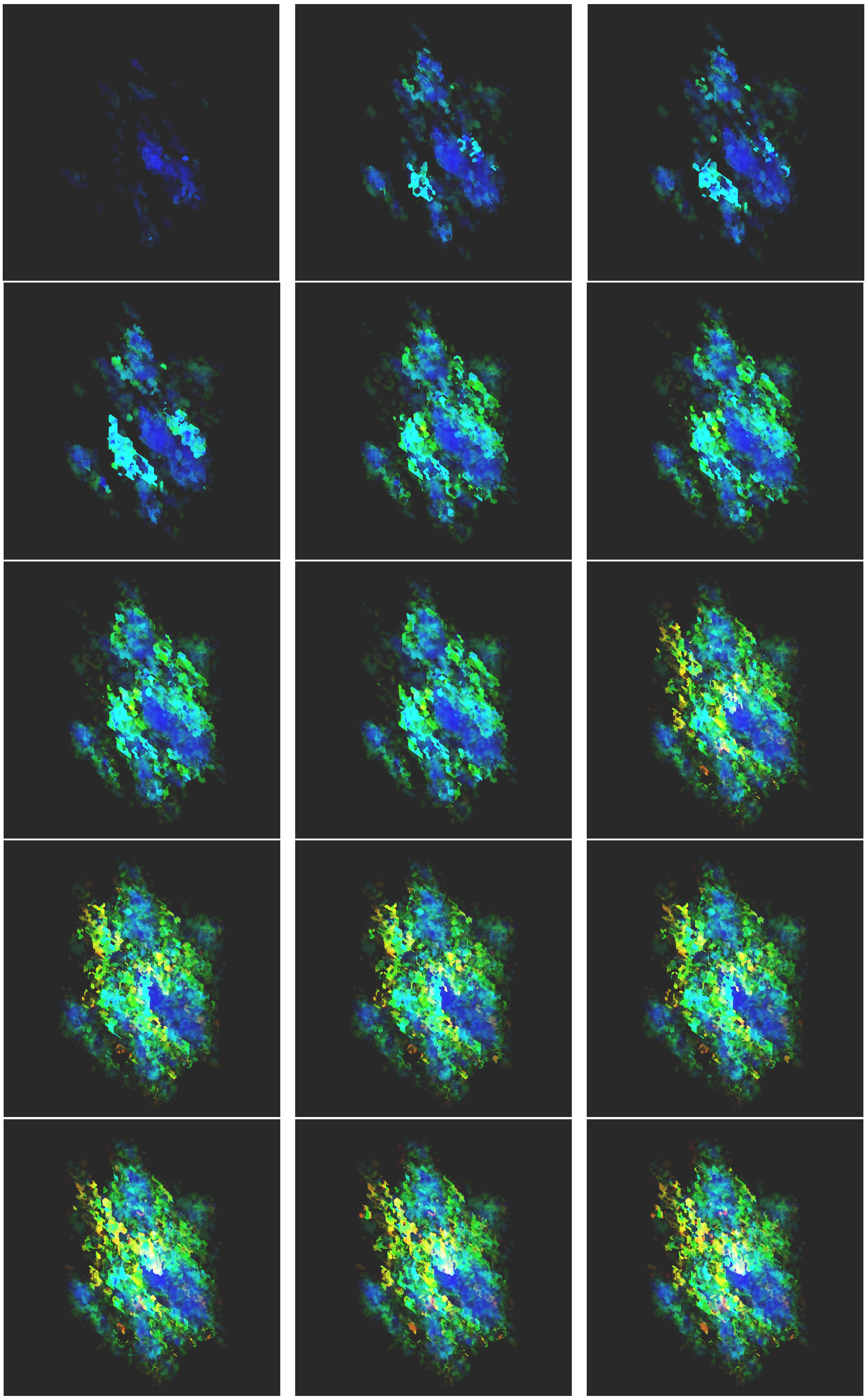

Figure 7. Images of cluster formation generated by the volumetric visualization module.

References

[1] J. Adler and U. Lev. Bootstrap percolation: visualizations and applications. *Braz. J. Phys.*, 33(3), 2003.

[2] G. Andrienko, N.Andrienko, J.Dykes, M.Gahegan, D.Mountain, P.Noy, J.C.Roberts, P.Rodgers, and M.Theus. Creating Instruments for Ideation: Software Approaches to Geovisualization. In *Exploring geovisualization*. Pergamon, December 2004.

[3] V. Bucha. Displaying 3-d seismic models through the vrml and gocad. *Seismic Waves in Complex 3-D Structures*, 1(11):337–355, 2001.

[4] I. Chatzis and F. A. L. Dullien. The modelling of mercury porosimetry and relative permeability of mercury in sandstones using percolation theory. *Int. Chem. Eng.*, 1(25):47–66, 1985.

[5] A. Coelho, M. Nascimento, C. Bentes, M.C. Castro, and R. Farias. Parallel volume rendering for ocean visualization in a cluster of pcs. In *Geoinfo 2004*, pages 291–304, November 2004.

[6] J. E. de Freitas. *Estudo de Algums Sistemas Complexos: Percolacao dependente do tempo, aplicacao a problemas de Petroleo, percolacao de longo alcance e modelo de reacao e difusao*. PhD thesis, Universidade Federal do Rio Grande do Norte, 2002.

[7] R. Denzer. Graphics for environmental decision making. *IEEE Comput. Graph. Appl.*, 13(2):58–64, 1993.

[8] R. Farias, J. Mitchell, and C. Silva. Zsweep: An efficient and exact projection algorithm for unstructured volume rendering. In *2000 Volume Visualization Symposium*, pages 91 – 99, October 2000.

[9] L. A. Ferrand and M. A. Celia. The effects of heterogeneity on the drainage capillary pressure: Saturation relations. *Water Resour. Res.*, 3(28):859–870, 1992.

[10] S. P. Friedman and N. A. Seaton. Critical path analysis of the relation-ship between permeability and electrical conductivity of three dimensional pore networks. *Water Resour. Res.*, 7(34):1703–1710, 1998.

[11] M. Gahegan. Four barriers to the development of effective exploratory visualisation tools for the geosciences. *International Journal of Geographical Information Science*, 13(4):289–309, 1999.

[12] M. Gahegan, M. Wachowicz, M. Harrower, and T. M. Rhyne. The integration of geographic visualization with knowledge discovery in databases and geocomputation. *Cartography and Geographic Information Science*, 28(1):29–44, June 2001.

[13] D. S. George and A. R. Kovscek. Visualization of solution gas drive in viscous oil. Technical report, 2001. Stanford University, CA, USA.

[14] H. Gould and J. Tobochnik. *An Introduction to Computer Simulation Methods*. Addison-Wesley, 1996.

[15] G. S. Grest, I. Webman, S. A. Safran, and A. L. R. Bug. Dynamic percolation in microemulsions. *Phys. Rev. A*, 4(33):2842–2845, 1986.

[16] W. L. Hibbard and D. A. Santek. Visualizing large data sets in the earth sciences. *IEEE Computer*, 22(8):53–57, 1989.

[17] M. Hilpert and C.T. Miller. Pore-morphology-based simulation of drainage in totally water wetting porous media. *Advances in Water Resources*, 1(24):243–255, 2001.

[18] G. R. Jerauld and S. J. Latter. The effect of pore-structure on hysteresis in relative permeability and capillary pressure: pore-level modelling. *Transp. Porous Media*, 1(5):103–151, 1990.

[19] D. Koller, P. Lindstrom, W. Ribarsky, L. F. Hodges, , N. Faust, and G. Turner. Virtual GIS: A real-time 3D geographic information system. In *Proc. of the 6th Conference on Visualization'95*, 1995.

[20] P. L. Leath. Cluster size and boundary distribution near percolation threshold. *Phys. Rev. B*, 1(14):5046–5055, 1976.

[21] M. I. Lowry and C. T. Miller. Pore-scale modeling of nonwetting-phase residual in porous media. *Water Resour. Res.*, 1(31):455–473, 1995.

[22] N. Max, R. Crawfis, and D. Williams. Visualization for climate modeling. *IEEE Comput. Graph. Appl.*, 13(4):34–40, 1993.

[23] A. McPherson and M. Maltrud. Poptex: Interactive ocean model visualization using texture mapping hardware. In *Proc. of the 9th IEEE Visualization Conference (Vis1998)*, pages 471 – 474, October 1998.

[24] D. Or and M. Tuller. Hydraulic conductivity of partially saturated fractured porous media: flow in a cross-section. *Advances in Water Resources*, 1(26):883–898, 2003.

[25] T. M. Rhyne. Scientific visualization in the next millennium. *IEEE Comput. Graph. Appl.*, 20(1):20–21, 2000.

[26] L. J. Rosenblum. Visualizing oceanographic data. *IEEE Comput. Graph. Appl.*, 9(3):14–19, 1989.

[27] R. Rosslerova and V. Kodes. The use of gis in percolation study of soil porous system. In *Proc. HydroGIS 96, Application of geographic information systems in hydrology and water resources management*, pages 259–265, 1996.

[28] N. S. Sagar and L. M. D. Castanier. Oil-foam interactions in a micromodel. Technical report, 1997. Dept. of Petroleum Engineering, Stanford University, CA, USA.

[29] OpenGL Architecture Review Board/Editor: Dave Shreiner. *OpenGL Reference Manual: The Official Reference Document to OpenGL, Version 1.4, 4/E*. Addison-Wesley, 2004.

[30] D. Stauffer and A. Aharony. *Introduction to Percolation Theory*. London: Taylor & Francis, 1992.

[31] S. Tomov, M. McGuigan, R. Bennett, and J. Smith, G.and Spiletic. Benchmarking and implementation of probability-based simulations on programmable graphics cards. *Computers & Graphics*, 29(1):71–80, 2005.

[32] L. A. Treinish. Visualization of scattered meteorological data. *IEEE Comput. Graph. Appl.*, 15(4):20–26, 1995.

[33] D. Wilkinson and J. F. Willemsen. Invasion percolation: a new form of percolation theory. *J. Phys. A*, 1:3365–3376, 1983.