Proceedings of the IASTED International Conference
APPLIED SIMULATION AND MODELLING
June 28-30, 2004, Rhodes, Greece

HIERARCHICAL TREES OF UNSTEADY SIMULATION DATASETS

Marek Gayer[1] and Pavel Slavik[2]

[1] Department of Environmental Engineering
Technical University of Crete
University Campus, 73100, Chania
Greece
xgayer@fel.cvut.cz

[2] Department of Computer Science and Engineering
Czech Technical University in Prague
Karlovo náměstí 13, 121 35 Prague 2
Czech Republic
slavik@cslab.felk.cvut.cz

ABSTRACT

We have designed and implemented concept of hierarchical structures consisting of simulation results stored in unsteady (time-varying) datasets. It allows incremental, progressive and easy construction and replaying of various simulation configurations with interactive visualization of results. The ability of modification of simulation boundary conditions is available in every node of the datasets tree. Every interactively selected path in the tree corresponds to one modified simulation solution. We have originally proposed and tested this concept in combustion processes simulation powered by a simple fluid simulator, but it can be easily utilized in general simulation applications, which use unsteady datasets for results storage and replaying.

Key Words

CFD, real-time and interactive simulation, fluid simulation, unsteady (time-varying) datasets.

1. Introduction

Real-time simulation with consequent visualization of various physical and environmental phenomena and processes is a common goal of many research projects and applications. Real-time simulation and visualization offers many significant benefits, such as possibility to quickly obtain the results, possibility to get a good overview of the dynamics of the simulated process and easy manipulation of the simulated model. Also it allows us to make interactive changes to the simulated process with immediate visualization response and visualization of the results in readable and easily understandable form. To achieve these goals, optimization of simulation based on simplifying the physical and computational model and code optimization are being used. Also, the models which can be effectively solved on existing computer architectures and can be easily and quickly implemented are preferred.

1.1 Unsteady datasets

In many complex cases, the above described effort is still not sufficient to obtain real-time simulation or it would lead to significant decrease of the computation preciseness. In such cases, storage of the unsteady (time varying or time dependent) simulation results in datasets, which can be in real-time replayed is used in wide range of simulation applications. Those datasets allow storing of one or more computed characteristics. The interactivity is limited; the additional changes to the already computed simulation setup and simulation configuration are not available. The replaying part is limited to the strict time portion based on the dataset stored on the disk. On the other hand, unsteady datasets offer full interaction in the visualization part of the simulation (e.g. changing of visualized characteristic, zooming to any simulated area, offering various visualization methods including out-of-core visualization, post-processing and others). The typical scientific area of this approach is CFD – where we can easily have insight of spatial and temporal behaviours of various physical processes in any area of interest [1], which is important for applications such as combustion [2]. For a detailed introduction and overview of unsteady datasets, we refer the reader to [3] and [4].

2. Our effort

We have proposed and investigated a new concept of storage representation of unsteady datasets. It is based on hierarchical tree structures. The tree structures consist of complete unsteady datasets with simulated results (in each of the tree node). Storing unsteady datasets can be used in general simulation applications. It allows us use some interesting features, such is for example possibility to increase replaying speed by both saving only one of several simulated frames to dataset (which can result in orders less disk space requirements) and skipping frames when replaying dataset. Instead of using unsteady datasets, we can also use only partial states to accelerate simulation, such as pre-calculated fluid simulator states, which we had developed and used in our previous research [5], but this approach is limited only to simulation based on fluid simulator and solvers and does not allow increase of replaying speed and does not allow saving only selected frames. We had implemented the concept of unsteady datasets tree to our pulverized coal combustion application, but it can also used in other simulation applications.

2.1 The hierarchical datasets concept

In each of the node, the user of the datasets tree can choose which one of the connected simulation datasets should be used. This way, the simulation datasets can be replayed and also created incrementally from any nodes. The replaying of the simulation results is similar to replaying of interactive movies. It allows investigating various simulation configurations and modifications. In each node, one of the sub-node datasets can be selected. It adds very important interactive dimension to the common datasets. We can replay the selected path in the tree from top to bottom. It results in sequential replaying of connected stored states of the simulation with selected modifications of configuration in each of the nodes. After finishing running a dataset at any node, the simulation can easily continue without any dataset (the last computed values from the dataset are used as boundary conditions for further simulation) with optional availability of storing the simulated results to datasets.

2.2 Incremental, reusable solution of the simulation task

This representation also adds easy possibility of modification the simulation configuration for both the simulation solver (during solution) and simulation user (during replaying). Because each configuration uses the already computed simulation results in ancestral sub-trees, it results in less disk requirements, which would be needed for all of the cases, if solved from the scratch. The structure and the content of the tree can be in any time extended and modified easily and interactively by simply storing another dataset. This allows incremental and step-by-step solution and creating hierarchy of re-playable results of the simulated problem. This is namely advantageous for solution of complex tasks. The new dataset uses as boundary conditions results of state, from which it was derived (corresponds to the ancestor node). Each of the nodes has its unique ID of the tree where it belongs and unique number of nodes in the tree (e.g. 4/15 in Fig. 1 and Fig. 2). This way, the sequences of connected simulated task modifications are defined in every node.

2.3 Interaction

The whole tree can be visualized and implemented to accept interactive user actions and modifications – e.g. extending the tree with nodes for another part of simulation, deleting parts of the partial simulation solution etc. An example of a datasets tree is shown in Fig. 1. Each node there represents one stored simulation dataset. Every interactively selected path in the datasets tree corresponds to one modified simulation solution. The current selected path of simulation with configurations modifications is being highlighted, see Fig. 1. In Fig. 2, the ability of changing the simulation configuration (boundary conditions) is being demonstrated. In the first sub-node, the simulation continues with the original inlet position. In the second sub-node, we have changed the direction and spread angle of the combustible injection from the inlet. Simulation continues with changed parameters with acceleration of visualization being kept regardless the interactive change of simulation configuration in the node.

Selected nodes can be made read-only, to for example disable accidental modification and deleting of the base tree structure, which is intended to be used by all users.

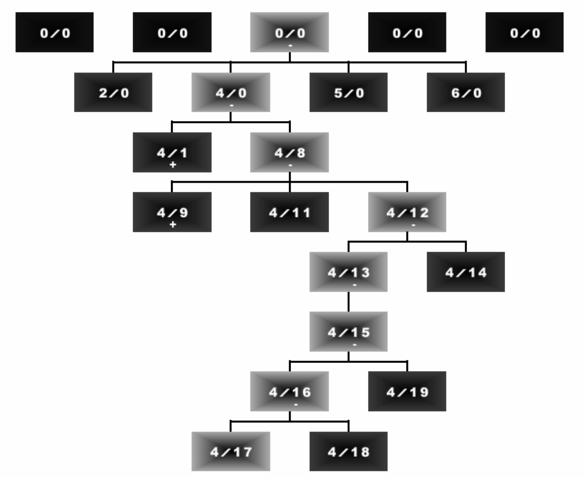

Figure 1: Hierarchical tree of simulation datasets

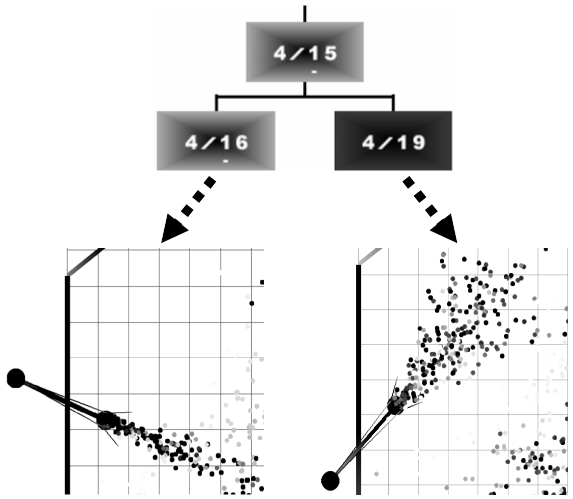

Figure 2: Illustration of interactive change of parameters of simulation in different sub-node datasets

2.4 Visualization of simulation data

Our concept allows subsequent real-time visualization of the datasets. Wide range of general visualization methods and approaches suitable for unsteady datasets such as LIC, contour and colour plate visualization [6], volume rendering [7], flow volumes [8] various types of streamlines [9] and other techniques [10], [11] can be used. Pre-computation based visualization for other techniques such is the motion maps [12] are also available. In such case, the pre-computed structures can be placed together with other data to datasets for selected nodes. This way, our solution could be possibly extended also for saving whole particle structures for special visualization methods such as [13] and for various encoding and data compression techniques such as [14]. For a detailed list of flow visualization possibilities, which of them most are also applicable to unsteady simulation datasets, we recommend [15].

Nowadays, real-time visualization of unsteady data is also maintained by using techniques and approaches regarding to new programmable abilities of current graphical accelerators, such as described in [16], [17] and [18]. We are also currently working on accelerated, real-time b-spline interpolation based grid visualization of our simulation results, implemented using vertex and pixel shaders [19].

2.5 Zooming features and time navigation

We can set any zoom level and change the current view position to any area of the simulated task under investigation in our simulation model. Using the unsteady datasets, we can furthermore arbitrary seek to selected time position in dataset. We can easily change the replaying speed (slower, faster) of the simulation or even back-reverse the combustion process. It is important because we can easily seek to a particular part of the simulation as easy as we can set an area of concern in space. We can control the time resolution and minimum speed of simulation. For example, when we are creating an interactive hierarchical unsteady datasets for simulations based on unstable fluid simulators [20], [21] we can increase the time step to make the simulation more precise and at the same time store only selected frames to be able to replay the simulation at the original simulation speed without unnecessary frames. This, way the common drawback of unstable fluid simulation, that forces at the interactive level to slow computation and interaction due to too small time level required can be easily avoided.

3. Implementation recommendations

We had implemented our concept to the 2D datasets and only for those cases, when all of the frames can fit in main computer memory. But our concept can also be used for large datasets, even those which would need out-of-core processing. In our case, we stored the data from the structured grid and particle system. The general implementation of the concept depends on the type of the solved task.

It is important of choosing an optimal storing representation of the whole structure. Here are many solutions possible. In our test implementation for combustion simulation, we use a straightforward approach, when all the dataset for nodes are stored in separate binary files.

3.1 File representation and conventions

The binary dataset file consists of a header at the beginning of file, where fields such as number of total frames are stored. Also, index of all frames at the end is present for enabling faster seeking possibility. After the header, the single frames are successively stored. The frames consist of arrays of characteristics of structured grid and all particles, which are needed to restore the particle system. We use 64-bit file access pointers to be able to work with files larger than 2 gigabytes.

For each node, we also keep a plain text file with basic configuration, which contains the initial simulated task boundary conditions. Applying these boundary conditions, we prepare the simulation first frame. We can either continue the whole computation, or we can use the dataset to replay the stored data.

For the simulation tree nodes stored in our proposed file organization, we have specified the following file naming convection. Each file name consists of a number of the tree and number of the node. In the beginning, we restore the whole tree by reading the list of files, sorting them and then recreating the tree from the top to the leaves. Extending the tree with new nodes is based on saving other files with appropriate names to the disk and extending the tree with new dataset. The example of a name would be test-002-007, where test is the name of the solved case, 2 is the ID number of the tree and 7 is ID number of the node.

3.2 Other implementation possibilities

The above described concept is only one possible solution, not a strict definition. It is also possible use different ones, which would fulfil the needs of various ranges of solved tasks. For example, the list of the nodes, frames and the tree structure and boundary conditions and configuration could be stored in XML files and binary datasets of the whole tree nodes could be stored in a one file instead of many. Any other storage such as a database based solution is applicable too.

3.3 Unsteady datasets tree applications

Using the above described concept, we can implement interactive and fast simulation applications. The unsteady datasets tree has been originally implemented and tested for our fluid application for combustion processes simulation. But it can be straightforward reused for others simulation and visualization applications, that either use or can be extended to use unsteady datasets for storing the results. This could possibly include for example also computer graphics field, where tasks such are animation of liquids and water [22] and gaseous phenomena [23] are being currently solved.

4. Tests and measurements

We have successfully implemented our concept of hierarchical datasets to a structured grid based, unstable fluid simulator for simulation and visualization of combustion processes. This simulator has been described in [21]. We have simulated the beginning of pulverized coal combustion in a 2D approximation of a power plant boiler. We have compared interactive features of direct simulation and visualization for both the performance and interactive features and requirements.

We have measured 2 cases. In the first case, the grid was set to 20 x 40 cells. In the second case, the grid was set to 50 x 100 cells, and the computation code was more precise due to a decreased time step.

As a storage drive for a tree of unsteady datasets we used a 120GB Seagate Barracuda Ultra Ata V. All results and measurements were performed on a commodity AMD Athlon 1333Mhz system equipped with 512MB SDRAM PC 133 memory.

The goal of all of these cases was to provide an interactive simulation reflecting changes of every 0.05 seconds in a boiler. The SIM case shows the direct simulation without of using hierarchical datasets. The UDT is the same case but with interactivity based on datasets trees. Because of using unstable type of fluid simulator, the simulation time step had to be set to 0.005s, which results in more frames being computed. Choosing less value would lead in the test case to computation instabilities and errors. The basic features of the task are summarized in the Table 1.

Features	SIM	UDT
Interactive change of the simulation	Full	In every tree node
Simulation time step set	0.005s	0.05s
Total played frames	10000	1000
Space shift and zoom	Yes	Yes
Time zoom features	No	Yes

Table 1: Setup and features comparison of direct simulation (SIM) and unsteady datasets tree (UDT)

Each of the cases contained about 10000 particles per animation frame. The grid size has been set to 20 * 40 and 50 * 100 cells. The results are summarized in the Table 2. In UDT cases, we received acceleration of 24 and 70 of the total simulation replaying time compared to direct simulation and immediate visualization. Storing the data to disk drive consumed virtually no additional time cost compared to the version without storing the data (with fast Ultra DMA 5 data transfer mode used).

Store method / Grid size	SIM / 20*40	UDT / 20*40	SIM / 50*100	UDT / 50*100
Simulation time	1089s	1103s	4617s	4665s
Write [MB/s]	N/A	0.7	N/A	0.4
Replay time	1062s	49s	4659s	68s
Read [MB/s]	N/A	15.9	N/A	26.4
Drawing speed [FPS]	**9.4**	**22.5**	**2.1**	**14.7**
Disk space [GB]	**N/A**	**0.78**	**N/A**	**1.8**
Total acceleration	**N/A**	**x 24**	**N/A**	**x 70**

Table 2: Results gained using direct simulation and full datasets

The amount of data stored in unsteady datasets is in our case limited to the specific storage drive used. When running demanding tasks with large cell grids, multiple characteristics and very long simulation time or frames, this method would suffer from both capacity and performance limitations of the drive. Then due to the speed limit of the drive, a performance bottleneck could appear when replaying the simulation. This could only be fixed by storing only selected frames (e.g. only 1 from 10 or even only 1 from 100 and more), but with losing the precision and resolution of the data stored. But in many cases, such as in commonly used unsteady fluid simulators, it is not very important, because the computation step differs from the simulation step and saving only selected frames is often fully sufficient.

5. A demonstration example of our results

The following figures display example simulation results of our pulverized coal combustion application based on unsteady datasets trees. Our application can arbitrarily select and switch among a total of about 50 cell and particle characteristics and statistics in the visualization phase. Fig. 3 shows a captured frame of real-time visualization of cell temperatures together with floating virtual coal particles inside the boiler chamber area. The results were directly simulated using our simple fluid simulator [21]. The same results and features (including switching to all other characteristics and statistics, see Fig. 4) can be gained using the unsteady datasets tree, with interactive possibilities of changing the simulation configuration in each of the node. In such a case, the system runs in orders faster than the original simulation. Because of ability of saving only necessary frames in fluid simulation, the disk space demands are acceptable.

In our test it consumed 1.8GB of disk space for fluid simulator states per 68 seconds animation.

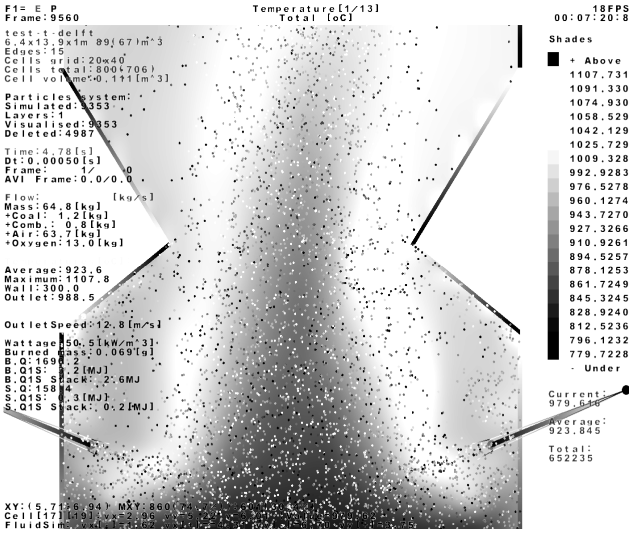

Figure 3: Sample visualization output of our coal combustion simulation application.

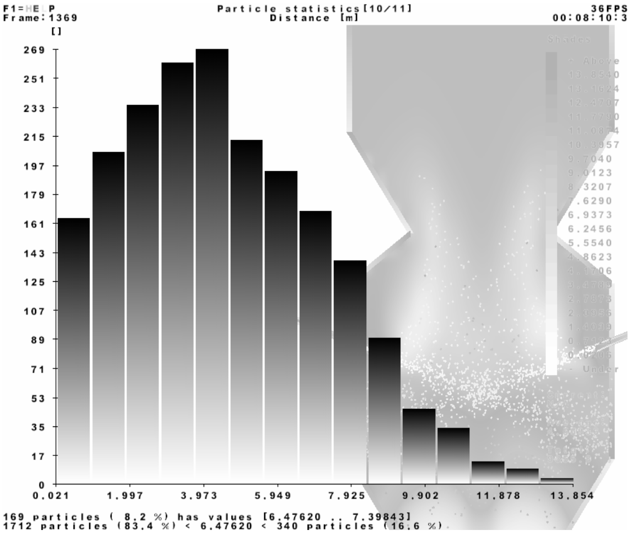
F1= E P
Frame:1369
[]
Particle statistics[10/11]
Distance [m]
36FPS
00:08:10:3
269
251
233
215
197
179
161
143
125
107
89
71
53
35
17
0
0.021
1.997
3.973
5.949
7.925
9.902
11.878
13.854
169 particles (8.2 %) has values [6.47620 .. 7.39843]
1712 particles (83.4 %) < 6.47620 < 340 particles (16.6 %)

Figure 4: Our application runs using unsteady datasets tree enabled in orders faster then without them

6. Future work

By implementing optimal existing compression and encoding methods to unsteady datasets files storage, we plan decrease both the disk space and traffic requirements for complex tasks.

We plan testing our system on very large or even 3D datasets and perform measurements for these cases.

Currently, we are also working on several techniques based on interpolation, which would considerably decrease the disk space requirements, while keeping the precise and smooth visualization of the simulation data's stored in unsteady datasets tree structures.

7. Conclusion

We have proposed, fully implemented and tested the original concept of hierarchical datasets tree structures in combustion processes simulation and visualization powered by unstable fluid simulator. Choosing unsteady dataset helps to avoid the common drawback of slow simulation of unstable fluid simulators due to small time steps.

Our concept overcomes the common drawback of the commonly used unsteady datasets, which results in loosing of interactivity on the simulation side. Proposed solution of hierarchical structures consisting of simulation results datasets over classical datasets allow incremental, progressive and easy construction and replaying of various simulation configurations with interactive visualization of results. The modifications of simulation boundary conditions are available in every node of the datasets tree. Every interactively selected path in the datasets tree corresponds to one modified simulation solution.

Our results can be easily utilized in general simulation and modelling applications, which use unsteady datasets for storage of the results. Users of simulation applications can now extend the prepared and pre-calculated simulation results with their own modifications with possibility to use the results of previous solutions stored in tree while keeping a high speed of replaying, because of utilization of unsteady datasets.

8. Acknowledgment

This project has been partially supported by the Ministry of Education, Youth and Sports of the Czech Republic under research program No. Y04/98: 212300014 (Research in the area of information technologies and communications).

References

[1] Bryson, S., and Johan, S. Time management, simultaneity and time-critical computation in interactive unsteady visualization environments. In *Proceedings of the 7th conference on Visualization '96* (1996), IEEE Computer Society Press, pp. 255-ff.

[2] Gillespie, J. The use of CFD in design of combustion equipment. *Progress in Computational Fluid Dynamics 1*, 1/2/3 (2001), 80-90.

[3] Cox, M. Large data management for interactive visualization design. In *Proceedings of the SIGGRAPH '99 System Designs for Visualizing Large-Scale Scientific Data course notes* (1999), ACM Press, pp. 5-29.

[4] Bryson, S., Kenwright, D., Cox, M., Ellsworth, D., and Haimes, R. Visually exploring gigabyte data sets in real time. *Communications of the ACM 42*, 8 (1999), 82-90.

[5] Gayer, M., and Slavík, P. Pre-calculated fluid simulator states tree. In *Proceedings of 12th IASTED International Conference on Applied Simulation and Modelling* (2003), IASTED, Anaheim: Acta Press, pp. 610-615.

[6] Robbins, K. A., and Gorman, M. Fast visualization methods for comparing dynamics: A case study in combustion. In *Proceedings of the 11th IEEE Visualization 2000 Conference (VIS 2000)* (2000), IEEE Computer Society.

[7] Crawfis, R., Shen, H.-W., and Max, N. Flow visualization techniques for CFD using volume rendering. In *Proceedings of the 9th International Symposium on Flow Visualization* (2000).

[8] Becker, B. G., Max, N. L., and Lane, D. A. Unsteady flow volumes. In *Proceedings of the 6th IEEE Visualization Conference* (1995), G. Nielson and D. Silver, Eds., IEEE Computer Society Press, pp. 329-333.

[9] Jobard, B., and Lefer, W. Unsteady flow visualization by animating evenly-spaced streamlines. In *Computer Graphics Forum (Proceedings of Eurographics 2000)* (2000), vol. 19 of *3*, Blackwell Publishers.

[10] Lum, E. B., Ma, K. L., and Clyne, J. Texture hardware assisted rendering of time-varying volume data. In *Proceedings of the conference on Visualization 2001* (2001), IEEE Computer Society Press, pp. 263-270.

[11] Bauer, D., Peikert, R., Sato, M., and Sick, M. A case study in selective visualization of unsteady 3D flow. In *Proceedings of the conference on Visualization '02* (2002), IEEE Computer Society, pp. 525-528.

[12] Jobard, B., and Lefer, W. The motion map: efficient computation of steady flow animations. In *Proceedings of the 8th conference on Visualization '97* (1997), IEEE Computer Society Press, Los Alamitos, US, pp. 323-328.

[13] Guthe, S., Gumhold, S., and Straßer, W. Interactive visualization of volumetric vector fields using texture based particles. *Journal of WSCG 10*, 3 (2002).

[14] Ma, K., Smith, D., Shih, M., and Shen, H. Efficient encoding and rendering of time-varying volume data. ICASE Report No. 98-22 NASA/CR-1998-208424, Institute for Computer Applications in Science and Engineering, NASA Langsley Research Center, 1998. Presented at SIGGRAPH '99.

[15] Hauser, H., Laramee, R. S., and Doleisch, H. State-of-the-art report 2002 in flow visualization. Tech. Rep. TR-VRVis-2002-003, VRVis Research Center, Vienna, Austria, 2002.

[16] Weiskopf, D., Hopf, M., and Ertl, T. Hardware-accelerated visualization of time-varying 2D and 3D vector fields by texture advection via programmable per-pixel operations. In *Proceedings of the Vision, Modeling, and Visualization VMV '01 Conference* (2001), T. Ertl, B. Girod, G. Greiner, H. Niemann, and H.-P. Seidel, Eds., pp. 439-446.

[17] Telea, A., and van Wijk, J. J. 3D IBFV: Hardware-accelerated 3D flow visualization. In *Proceedings of the IEEE Visualization '03* (2003), ACM Press.

[18] Weiskopf, D., and Ertl, T. GPU-based 3D texture advection for the visualization of unsteady flow fields. In *WSCG Short Communication Papers proceedings* (2004), UNION Agency - Science Press, Plzen, Czech Republic.

[19] Kadlec, P., Gayer, M., and Slavík, P. Visualization using hardware accelerated spline interpolation. In *WSCG Short Communication Papers proceedings* (2004), UNION Agency - Science Press, Plzen, Czech Republic.

[20] Foster, N., and Metaxas, D. Realistic animation of liquids. *Graphical Models and Image Processing 58*, 5 (1996), 471-483.

[21] Gayer, M., Slavík, P., and Hrdlička, F. Interactive system for pulverized coal combustion visualization with fluid simulator. In *Proceedings of the Visualization, Imaging, and Image Processing Conference* (2002), IASTED, Anaheim: Acta Press, pp. 288-293.

[22] Enright, D., Marschner, S., and Fedkiw, R. Animation and rendering of complex water surfaces. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (2002), ACM Press, pp. 736-744.

[23] Stam, J. Stable fluids. In *Proceedings of Computer Graphics and interactive techniques* (1999), vol. 33, ACM Press, pp. 121-128.