Steering Smog Prediction

Robert van Liere[1] and Jarke J. van Wijk[2]

[1] Center for Mathematics and Computer Science CWI, P.O. Box 94097,1090 GB Amsterdam, Netherlands

[2] Netherlands Energy Research Foundation ECN, P.O. Box 1, 1755 ZG Petten, Netherlands

Abstract. The use of computational steering for smog prediction is described. This application is representative for many underlying issues found in steering high performance applications: high computing times, large data sets, and many different input parameters.

After a short description of the smog prediction model, its visualization and steering are described. The amount of computation needed to solve the governing transport equations is alarmingly high. The user has a large number of options for the display of various aspects of the simulation, and also for the interactive control of its input data.

Smooth animation is very important to monitor the evolution of pollutants and for a responsive feedback to parameter changes. Here a performance of least 15 frames per second is required. We discuss techniques that allow the user to steer the numerical solver, such that an optimal tradeoff between computation speed and accuracy can be made.

Keywords: scientific visualization, computational steering, interactive 3D-graphics, atmospheric simulation.

1 Introduction

The word 'smog' is a combination of 'smoke' and 'fog'. It was originally used to describe city fogs containing large amounts of air pollutants. Environment protection agencies, such as the Dutch National Institute of Public Health and Environmental Protection (RIVM), provide detailed forecasts of the expected levels of smog. A variety of smog prediction models have been developed to determine these forecasts. Solving smog prediction models is a very time consuming process and results in large amounts of output data. However, due to the advent of high performance computing and better numerical algorithms, these computational costs are decreasing.

The next challenge is to apply computational steering to smog simulation. Computational steering allows the user to change simulation parameters while the simulation is in progress and its results are visualized. Computational steering enhances productivity by reducing the time between changes to model parameters and the viewing of the resulting output. This simplifies the study of alternative scenarios, and increases insight into the model and the effect of parameter changes. For example, a user may want to investigate the effect that varying emission levels have on the computed pollutants. Another example is to study the effect of varying meteorological conditions, such as alternative wind fields.

In this paper we demonstrate how computational steering is applied to a smog prediction model. The chosen application is of particular interest to the HPCN community because it is representative for many underlying issues found in steering high performance applications: high computation times, large data sets, and varying inputs. The original model was developed at RIVM [1], and the numerical solver was developed at CWI [2]. We have embedded the simulation into the Computational Steering Environment, CSE [3] and parallelized the solver so that it would take advantage of high performance SMP architectures.

In [3], a very simplistic smog simulation was used to illustrate some basic concepts of the CSE. The smog simulation used here is different in many ways: it uses a much more realistic chemical model, it requires orders more computation power, and generates much more output. Also, the visualization is 3D and has many more steering possibilities. One of the goals in this work was to provide smooth animation of the simulation in progress. At least 15 frames per second are required because only then will the user be able to monitor the evolution of a pollutant.

The format of this paper is as follows: first we briefly present the underlying smog model and solver which is used to compute the model. We discuss the complexity of the problem and show why high performance computing is mandatory to interactively solve the governing equations. In section 3, we present the visualization of and interaction with the simulation. In section 4 we discuss implementation and performance issues. Finally, we draw some conclusions.

2 Smog Simulation

2.1 The Physical and Chemical Model

The simulation forecasts the levels of air pollution which is modeled by 15 pollutants, such as ozone (O_3), sulphur dioxide (SO_2) and nitrogen dioxide (NO_2).

For the geographical domain a model of Europe is used. The vertical stratification is modeled by four layers; the surface layer, the mixing layer, the reservoir layer, and the upper layer.

The governing equations of the model are described by a set of partial differential equations that determine the advection, diffusion, emission, wet and dry deposition, fumigation, and chemical reactions. In spherical coordinates, the full model equation is written as

$$
\begin{aligned}
\frac{\partial c}{\partial t} = & -\frac{1}{r\cos\theta}\left[\frac{\partial(uc)}{\partial\phi} + \frac{\partial(vc\cos\theta)}{\partial\theta}\right] && \text{advection} \\
& + \frac{\kappa_{diff}}{r^2\cos\theta}\left[\frac{1}{\cos\theta}\frac{\partial^2 c}{\partial\phi^2} + \frac{\partial c}{\partial\theta}\left(\frac{\partial c}{\partial\theta}\cos\theta\right)\right] && \text{horizontal diffusion} \\
& + \frac{\partial}{\partial z}\left(K_z(z)\frac{\partial c}{\partial z}\right) && \text{vertical diffusion} \\
& + S_d(c) && \text{dry deposition} \\
& + S_w(c) && \text{wet deposition} \\
& + F(c) && \text{fumigation} \\
& + Q && \text{emission} \\
& + R(c), && \text{chemical reactions}
\end{aligned}
$$

in which c denotes the vector of computed pollutants. The pollutant concentration in a layer is interpreted as the average value over that layer in vertical direction.

The chemical model describes summer smog episodes, which are characterized by high levels of ozone concentrations. The summer smog model is defined as 17 reactions between 15 different pollutants. Numerically, these reactions are defined as a set of 17 stiff ordinary differential equations.

There are many parameters that control the input conditions of the model. We name a few:

- Emission fields.

  Emissions are divided into 6 categories: emission due to combustion, space heating, refinery, chemical processes, solvents and traffic. The model distinguishes in emission point sources and emission surface sources.

  Short and middle term predictions can be made by changing emissions; e.g. the effects of shutting down a power plant (modeled as a point source) can be studied.
- Meteorological parameters.

  Examples of meteorological parameters are wind fields, temperature, cloud coverage, humidity, etc. Wind fields are read from a data base provided by RIVM.

  Short term predictions can be made by changing meteorological parameters. For example, a user can study the effect of different weather forecasts.
- Geographical information.

  Geographical information, such as land-sea boundaries and forest densities influence diffusion, wet and dry deposition, fumigation, and the chemical reactions processes.

  Long term predictions can be made by changing geographical information. For example, to study the effect of deforestation on pollution.

2.2 The Solver

The solver uses an adaptive grid refinement technique to improve the efficiency of the model calculations. The tradeoff to be made in local grid refinement is calculation accuracy versus computation speed.

The geographical domain is discretized into a base grid with a resolution of 52×55 cells. When a grid cell is refined it will be split uniformly into 4 smaller grid cells. The base grid is the grid at level 1. A new grid at level N is constructed by refining grids cells at level $N-1$. Figure 1 illustrates grid refinement: 6 cells a1, a2, b1, b2, b3, c3 are refined into 24 new cells. The 6 grid cells define the grid at level 1. The 24 new cells define the grid at level 2.

The governing equations are solved for each grid cell. Hence, for each time step, the maximum amount of work to be done is proportional to the number of grid cells :

$$\sum_{i=1}^{maxlevel} 4^{i-1} \times L \times 52 \times 55$$

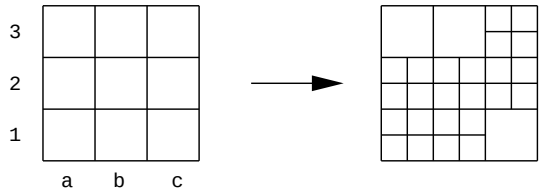

Fig. 1. Adaptive refinement of a grid cell.

in which L are the number of stratification layers. Currently L is set to 4, but this can be increased.

The amount given above is clearly the worst case situation in which the complete grid is refined until *maxlevel* is reached. Fortunately, usually only a fraction of the maximum amount of work is done. For example, if $maxlevel = 4$, test cases have shown the total amount of work to be done is only 13% of the number mentioned above. Higher levels of grid refinement will result in more relative savings.

The solver uses several criteria to automatically determine if and where adaptive grid refinement is to be performed. For example, grid refinement is applied to cells where pollutant concentrations have large spatial gradients. However, these general purpose criteria do not incorporate knowledge about particular interests of the user. Hence, the user has to be enabled to steer the refinement criteria. There are various parameters that control the grid refinement algorithm. First, the tolerance level which determines when a cell should be refined. Second, a region of interest which determines where grid refinement should be performed. Third, *maxlevel* which determines the depth of refinement. The user can select optimal values for all parameters here: the best tradeoff between accuracy and speed of computation, possibly varying per region of interest.

3 Visualization and Steering

The visualization and interaction is designed to be performed on a graphics workstation remote from a compute server. After each time step of the simulation the compute server will send all necessary data over the network for rendering. Upon interaction the graphics workstation will send new parameter values to the compute server.

– *Display*

Figure 2 shows a snapshot of a time step in the simulation of a O_3 concentration field on each of the four layers. A gouraud shaded colored mountain

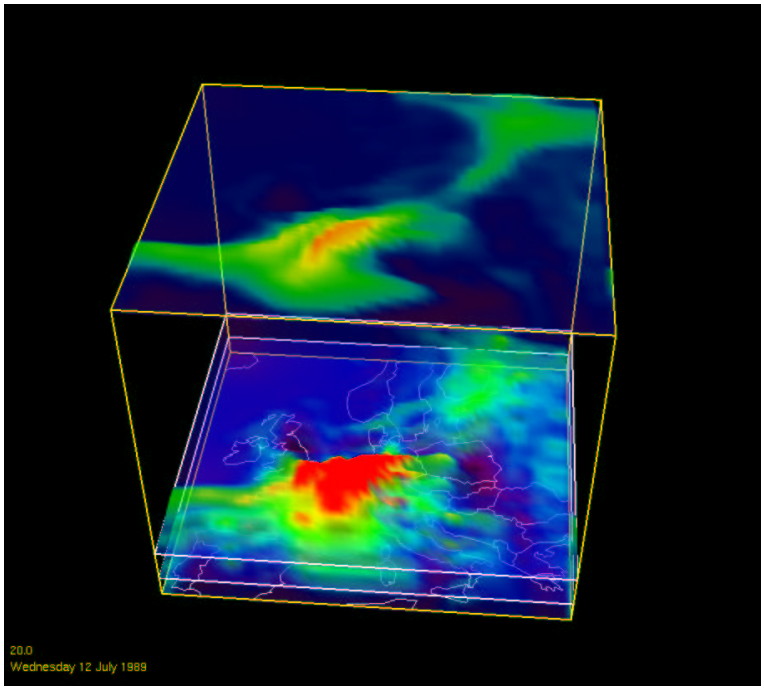

Fig. 2. Four layers of ozone concentration over Europe.

plot is used to display the concentration of a pollutant. Color and height are assigned according to the concentration at the corresponding grid cell. Transparency is used for the simultaneous display of all four layers; the amount of opacity is proportional to the concentration.

The wind field is shown via small vectors per grid cell. This representation will clearly display the changes of the wind field if the frame rate is high enough. Each level of grid refinement is displayed as a uniquely colored mesh. Successive grid levels are rendered on top of the previous level.

Figure 3 shows a different view of a time step. Only two of the four layers shown. Three levels of the grid refinement are shown in the surface layer. Level 2 grids are shown in red, level 3 grids in green, and level 4 in blue. In

this way the user can clearly see how the grid is refined. The wind field is shown in the upper layer.

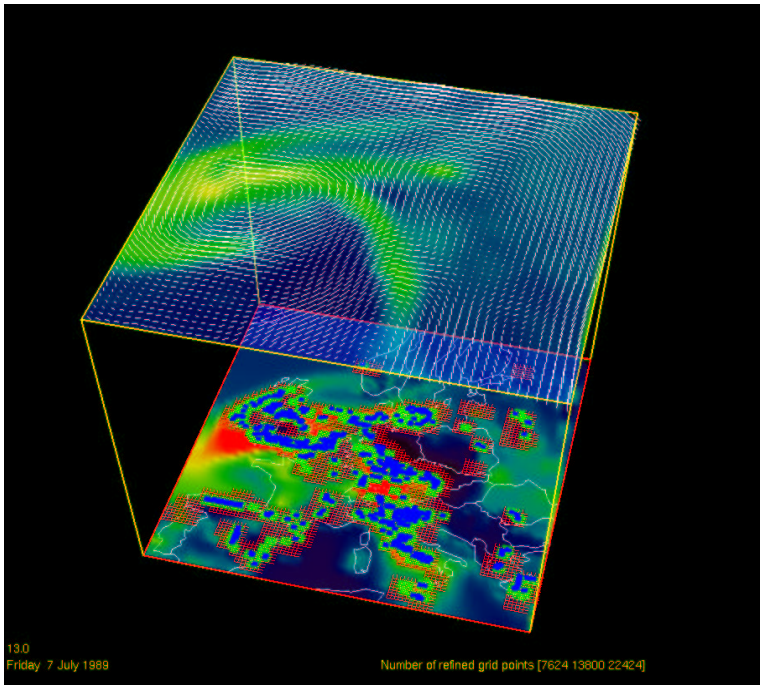

Fig. 3. Ozone with grid (surface layer) and wind fields (upper layer).

Animated sequences of these visualizations reveal several important features. The evolution of individual pollutant concentrations over time can be monitored. Also, monitoring the grid refinement algorithm over time can provide valuable information for the simulation developer. The user can monitor these features only if the frame rate is high enough. We experimentally found that here at least 15 frames per second were required to give the illusion of smooth motion. Below this threshold only discrete images were observed, and the motion was much less obvious.

The user can perform interaction for viewing purposes: selectively display one or more pollutant concentration layers, a wind field or a grid mesh. The mouse is used for zooming and rotation. Finally, a color map editor for mapping the value range onto color and transparency is available.

– *Steering*

The user can steer various aspects of the model and solver. First, emission field editors have been developed that allow the user to control the emissions. The editors provide a direct manipulation interface to model emission parameters, in which geometric representations are bound to simulation data. The user may manipulate the geometry resulting in mutations of simulation data. The user can change a emission point source by dragging its position or changing the emission values. Also, filters may be defined over a geographical region which damp or amplify the emission surface sources.

Second, several parameters that control the grid refinement algorithm are accessible through a direct manipulation interface. The user can change the tolerance of the grid refinement with a slider. Also, a region of interest defines the area in which grid refinement is to be performed. Finally, the maximum number of levels that determine the grid refinement can be set.

Figure 4 shows a snapshot of the three editors. Users can directly manipulate various aspects and immediately view the effects. The emission field editors are used for cause/effect analysis; e.g. what would be the effect on the ozone concentrations when an emission changes? Steering the grid refinement process allows the user to make a tradeoff between speed and accuracy of the simulation. For example: when nothing "interesting" is happening in the simulation, the user may wish to compute with large grid resolutions.

– *Record and Playback*

A record and playback facility is available which the user may turn on or off. In record mode a selection of pollutants, wind fields and grid levels will be dumped into a file. In playback mode the file will be read and rendered. The big advantage of the playback mode is that smooth animation frame rates can easily be realized.

The playback facility may also be used in lock step with the simulation. In this case two renderers are set up to display the recorder and the simulation. Combining the playback facility with steering allows the user to experiment with alternative scenarios and compare results simultaneously.

4 Implementation and Performance

4.1 Computational Steering Environment

The Computational Steering Environment [3], CSE , is a client/server based architecture for computational steering. It provides many general purpose tools

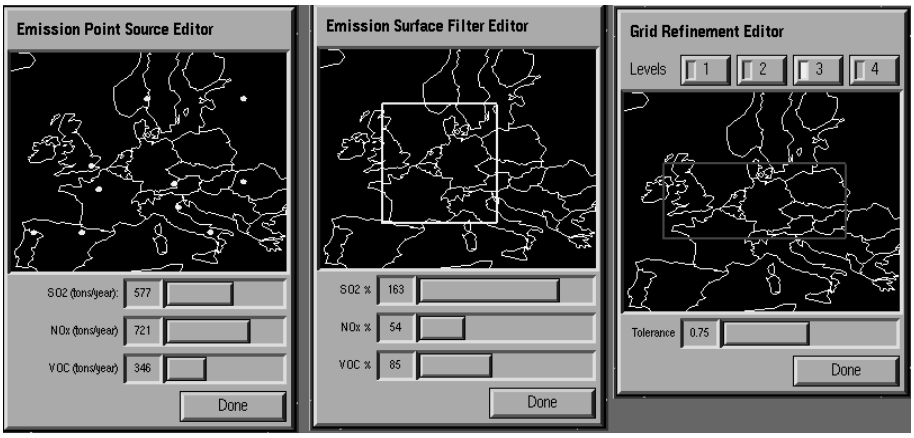

Fig. 4. Three editors; point source (left), surface source (middle), grid refinement (right).

that allow the user great flexibility in defining visualizations.

An overview of the architecture of the CSE is shown in figure 5. The architecture is centered around a *Data Manager* that acts as a blackboard for communicating values. Separate processes (*satellites*) connect to the Data Manager and exchange data with it. The simulation, record/playback and renderer are all packaged satellites.

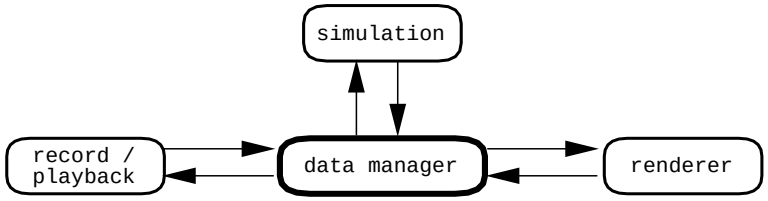

Fig. 5. The CSE architecture

The purpose of the data manager is twofold. First, it manages a database of variables. Satellites can create, open, close, read, and write variables. For each variable the data manager stores a name, type, and value. Second, the data manager acts as an event notification manager. Satellites can subscribe to state

changes in the data manager. When such a state change occurs the satellite will receive a notification from the data manager. For example, when a satellite subscribes to mutation events on a particular variable, the data manager will send a notification to that satellite whenever the value of the variable is mutated.

The CSE is implemented as a network transparent environment, which uses TCP/IP for data movement over machine hosts. The user may choose to execute a satellite on any host. However, the most natural configuration would be to run the render satellite on a graphics workstation and the simulation satellite with the data manager remotely on a high performance compute server.

The simulation, record/playback and renderer satellites are custom satellites implemented specifically for this smog simulation. Performance was the main reason for us to choose for implementing custom satellites. The standard satellites provided by the CSE were not able to cope with the requirement of 15 frames per second.

4.2 Performance

Performance is analysed by considering the computation of the simulation, transport from compute server to rendering workstation, and rendering separately.

- Computation.
  As noted in section 2.2, the amount of work needed to be done in the simulation is proportional to the total number of grid cells. For each grid cell the chemical reactions, formulated as a set of stiff ODE's, must be solved. For example, if the maximum level of grid refinement is three, the number of cells per time step to solved will be proportional to:

$$4 \times 52 \times 55 + 4 \times 4 \times 52 \times 55 + 16 \times 4 \times 52 \times 55 \approx 250K$$

  Hence, if the goal is 10 time steps per second, then approximately 2.5M sets of stiff ODE's, each consisting of 17 equations, must be solved. This would be the worst case. Fortunately, since the user can control when and where refinement should take place, more reasonable figures can be realized.
  The following table shows the performance of the simulation in time steps per second on the SGI Origin2000. The tolerance value and refinement region were fixed throughout the simulation. This table clearly shows that the performance is scalable for the number of available CPUs.

CPUs	SGI Origin2000
1	0.9
2	1.7
4	3.1
8	5.4

Performance measured in steps per second.

- Data movement.
  The amount of data sent from the data manager to the renderer is proportional to the number of cells that have been computed. Each pollutant is represented as a scalar field. Thus, assuming a floating point number is represented by 4 bytes, a bandwidth of $4 \times N$ bytes is required if N is the number of calculated cells. In addition, for each time step, new wind fields and grid meshes will be sent to the renderer.
  Assuming 15 frames per second, the worst case will saturate even a HIPPI channel. However, when under user control, an ATM OC-3 would seem sufficient.
- Rendering.
  Data is rendered by OpenGL as quadrilateral strips. Each pollutant layer consists of 52 strips of 55 quadrilaterals for each layer. Many high end graphics workstations are able to handle these quantities at 30 frames per second.

In summary, performance is hampered by the simulation. We do not consider this a very big problem because the simulation is inherently scalable. With additional processing power we are confident that we can reach approximately 15 frames per seconds. At higher frame rates the transport of the data will become the performance bottleneck. The renderer can easily keep up with the performance requirements. Only if different visualization techniques are used (such as volume rendering) then the renderer may cause a bottleneck.

5 Conclusion

We have shown how a realistic and high performance smog prediction simulation is steered. Computational steering increases insight to smog forecasting. Steering combined with playback facilities provide an effective means to compare alternative scenarios.

The amount of computation needed to solve the governing transport equations is alarmingly high. Fortunately, the user can steer the underlying numerical solvers, resulting in a tradeoff between calculation accuracy versus computation speed.

The performance goals have not yet been met. Smooth animation is important for monitoring the evolution of a pollutant. The bottleneck is currently the lack of resources on the compute server. However, since the simulation code seems to be scalable, we are confident that the goal of 15 frames per second can be met.

Future work will include extending the model to 32 layers, providing additional steering possibilities, and implementing more visualization techniques such as simultaneously displaying multiple pollutants.

Acknowledgements

We would like to thank Maarten van Loon of CWI for the help on the underlying numerical methods. Michael Brown, Jan Boerhout and Roger Chu of Silicon Graphics provided valuable feedback and help on simulation tuning. This simulation has been used as a demo by Silicon Graphics at SuperComputing '96 in Pittsburgh.

References

1. H. Jetske van Rheineck Leyssius, F. de Leeuw, and B. Kesseboom. A regional scale model for the calculation of episodic concentrations and despositions of acidifying components. *Water, Air and Soil Pollution*, 51:327–344, 1990.
2. M. van Loon. *Numerical Methods in Smog Prediction.* PhD thesis, University of Amsterdam, June 1996.
3. R. van Liere and J.J. van Wijk. Computational Steering. In H. Liddell, A. Colbrook, B. Hertzberger, and P. Sloot, editors, *High-Performance Computing and Networking*, pages 696–702, Brussels, April 1996. Springer-Verlag.