Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2004)
O. Deussen, C. Hansen, D.A. Keim, D. Saupe (Editors)

Visualization For Public-Resource Climate Modeling

D.A. Stainforth[†], D. Frame[†] and J.P.R.B. Walton[‡]

[†]Atmospheric, Oceanic and Planetary Physics, University of Oxford, Clarendon Laboratory, Parks Road, Oxford OX1 3PU, United Kingdom
[‡]The Numerical Algorithms Group Ltd, Wilkinson House, Jordan Hill Road, Oxford OX2 8DR, United Kingdom

Abstract

Climateprediction.net aims to harness the spare CPU cycles of a million individual users' PCs to run a massive ensemble of climate simulations using an up-to-date, full-scale 3D atmosphere-ocean climate model. Although it has many similarities with other public-resource computing projects, it is distinguished by the complexity of its computational task, its system demands and the level of participant interaction, data volume and analysis procedures. For simulations running on individual PCs, there is a requirement for visualizations that are compelling and readily grasped, since most users will be interested in the output from the model, but will have a limited level of scientific experience. This paper describes the design and implementation of these visualizations.

Categories and Subject Descriptors (according to ACM CCS): I.3.8 [Computer Graphics]: Applications I.6.3 [Simulation And Modeling]: Applications C.1.4 [Parallel Architectures]: Distributed Architectures J.2 [Physical Sciences And Engineering]: Earth and Atmospheric Sciences

1. Introduction

The increase in the speed and capacity of widely available networked PCs is providing an opportunity to use distributed computing to tackle modeling tasks that would previously have required access to a supercomputer. Projects such as SETI@home [KWA*01] have demonstrated the principles of *public-resource computing* which allows researchers to tap into the idle processing capacity of millions of PCs volunteered by businesses and the general public. Previous projects of this type have required the installation of an analysis application on the participant's machine, typically followed by the download from a central server of a small amount of data that is tested by the application. The test result, which is usually of the form “match” or “no match”, is then returned to the server, and the next piece of data is downloaded. As each test takes only a few hours to process, the participant need not have a long-term commitment to the project.

This paradigm has been taken a step further by climate*prediction*.net [SKA*02], which differs from previous projects in the volume and management of distributed data, the complexity, duration and granularity of the modeling task, and the need for tools which will maintain long-term participant interest. More specifically, the application that participants are invited to install is a full-scale climate model which is run locally to simulate the Earth's climate up to 2050. Each run is characterized by a unique set of values for certain parameters used by the model, leading to different representations of the physics in each simulation. The combination of results from all runs can be used to make a probabilistic climate forecast which accounts for uncertainty in the model formulation and the processes it represents.

The high resource requirements for each run (each 100 year simulation takes around three months on a 1.4 GHz machine), the large number of runs needed for an accurate forecast and the fact that climate change is of interest to the general public all make this project a good candidate for public-resource computing using volunteered resources. To date, successful projects of this nature have been characterized by the stimulation of the public's imagination and a feeling of being able to contribute to the solution of a problem which is global in its effect.

If the project is to have a good level of uptake, the application must be simple enough to be run by inexperienced PC users with slow Internet connections. At the same time, the output from the package should be compelling enough to stimulate continued participation. Visualization plays an important role in meeting this demand, since it allows the user to view the progress of their run, whilst also raising awareness of the issues being studied by the project.

This paper describes the design and implementation of real-time visualization for the climate*prediction*.net desktop package. We introduce the project in the following section, and discuss its visualization component in §3, including some example displays. A final section discusses the present status of the project and describes the future evolution of its visualization requirements.

2. Global Climate Prediction

2.1. Project Rationale

Recent simulations using climate system models have reproduced many large-scale features of recent (*circa* 150 years) climate change with a good degree of accuracy (see, e.g. [STJ*00]). These models can be used to make projections of the effect of various levels of greenhouse gas emissions on future climate change for the Earth. However, there are no objective uncertainty limits on current predictions, which makes them less useful to policy makers. Recognizing this, the Inter-Governmental Panel on Climate Change (IPCC) has recently [WG1] called for the improvement of methods to quantify these uncertainties, and this is the main scientific goal of climate*prediction*.net.

The models contain a number of adjustable parameters whose values are poorly constrained by available data on the processes they are supposed to represent. For example, clouds play a key role in the climate system through their strong feedback on radiative forcing—they can either cool or warm the surface via their Albedo or greenhouse effect. Some of the model parameters which affect cloud formation include the critical relative humidity at each height, the threshold for liquid water content in the atmosphere over sea and land, and the rate of conversion of cloud liquid water to precipitation. The effect of perturbing parameters such as these within their range of uncertainty is non-additive, so the only way to investigate their impact is perform a simulation for each point of interest in an N-dimensional parameter space, where N is the number of parameters. Whilst intelligent sampling of this space should help reduce the number of simulations, it has been estimated [SKA*02] that between one and two million independent simulations will be required.

2.2. Model Details

We use versions of the Climate Model of the Hadley Centre for Climate Prediction and Research at the UK Meteorological Office [Cul93]. One part of the model represents the ocean, and the other the atmosphere. The latter is modelled in 3D (19 levels in the vertical, with a horizontal resolution of 2.5° in latitude by 3.75° in longitude) and interacts with the land surface and cryosphere. This is the same model as used by the Met Office in numerical weather prediction, but we are using a coarser resolution in order to simulate the long timescales associated with climate research. Our resolution is almost precisely the same as that of all the models used in the most recent IPCC climate change assessment [WG1], and it is believed that many of these models will be used at this resolution in the next assessment in 2007.

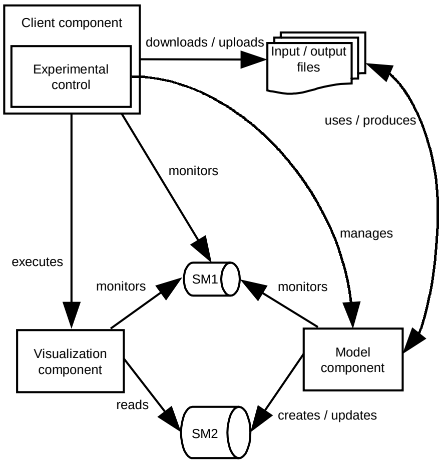

Figure 1: *Architecture of the participant package. All components are installed on the participant's machine. Details of parameter perturbations are downloaded from climateprediction.net servers and the simulation results are uploaded to these servers at the end of the run. SM1 and SM2 are two shared memory arenas which are used for communication between the client, model and visualization components (SM1) and for transfer of data from the model to the visualization (SM2).*

2.3. Implementation

We chose Windows as the initial target platform for climate*prediction*.net because of the experience of [KWA*01] who found the contribution of Windows participants to be an order of magnitude greater than that of users running on other platforms. Accordingly, the model of §2.2—consisting of around 0.5M FORTRAN lines—has been ported [SKM*02] to Windows. (Other environments are also under consideration—see §4, below.)

The first part of the main experiment consists of simulations of the 1950-2000 period. During each run, results are compared to historical data for variables such as surface temperature, and a quality-of-fit value is calculated. Those parameter sets leading to models with the closest fit to past observations will then, in the second part of the experiment, be used in simulations of the 2000-2050 period. The results will help with an objective assessment of the uncertainty for climate projections to 2050 under a range of future emission scenarios.

To participate in climate*prediction*.net, a user first downloads the package from a distribution server. Their first set of parameter values are obtained from a client interaction server and the simulation is started as an ‘idle’ priority process. No further network communication is required until the run is completed—usually after several weeks—when the package uploads a subset of the simulation results (whose size is about 10 Mb) and contacts the client interaction server to get details of the next run to be performed.

Figure 1 shows the structure of the participant package in more detail. The model and visualization are kept separate because it is envisaged that the model executable will be comparatively stable over the lifetime of the experiment, whilst the visualization and the client will probably require periodic upgrades. In addition, since the model component takes up a substantial fraction of the package, it is desirable to limit the number of times it has to be downloaded from the distribution server. More details on the overall system design can be found in [SKM*02].

3. Real-Time Visualization

3.1. Motivation

As noted above, an essential requirement of the design is that participants should be able to see what their simulation is doing in real time. One of the reasons for this is simply so the participant can “see something happening”, whether for reassurance, or curiosity, or scientific interest. Another reason for compelling visualization is to raise awareness of the issues under study. Climate change and the effect of greenhouse gas emissions on global warming are topics of wide concern. The scale of the problem, coupled with the degree of commitment which is required to address it, can be daunting. Watching a desktop simulation of changes in the earth’s climate could stimulate further consideration of the issues, along with a feeling of personal satisfaction at being involved—at however small a level—in a major research experiment in this field.

3.2. Previous Work

The GLOBE program [GLO] is a popular public-access project aimed at schools which allows participants to share their measurements of various types of environmental data over the internet, to plot them as maps and graphs and to collaborate on specific environmental projects. Visualization in GLOBE is done using lightweight clients within a web browser, and has not to date involved 3D display.

Middleton and his colleagues at NCAR have created visualizations of climate change [VETS] calcluated by large-scale climate models and have performed collaborative analysis using large displays and the AccessGrid. Other workers have used a visualization toolkit for climate modeling display. Treinish has produced compelling displays of ozone depletion [Tre93] using OpenDX. Sheldon and Vahlenkamp [GFDL] have used IRIS Explorer to create visualizations of global warming and hurricane formation. Finally, VisAD has a data model which maps well to the visualization of meteorological data, and has been used as the basis of NCAR’s VMET tool for the display and analysis of meteorological data [VMET].

Although much work has been performed on the display of results from live feeds or data warehouses, none appears to have been done with real-time display of simulation output in a public-resource environment, but it should be possible in this work to make use of some of the techniques previously applied to larger-scale data.

3.3. Visualization Implementation

At each time step, the simulation outputs a series of two-dimensional arrays to the visualization via the SM2 shared memory arena (see Figure 1). The model interval between time steps is 30 minutes and the resolution of each array is 96 by 73 (see §2.2). Amongst other model variables, the arrays contain values for:

- cloud cover at three heights above the earth,
- snow depth and sea ice coverage,
- surface temperature,
- precipitation.

Two of the three heights at which cloud cover is selected are the lowest and highest of the atmospheric model’s 19 vertical layers; in addition, we select the 10th level as representative of the middle of the atmosphere. The snow and ice datasets are complementary (since the former is only calculated over land) and so can be combined into a single snow/ice coverage array.

Initial visualization prototypes were built using IRIS Explorer, but its use as a basis for the visualization component of the participant package was not deemed acceptable because the size of the component—and hence, its contribution to the package download time—was too large. Subsequent work was therefore done using a lower-level graphics library, in order to reduce the size of the component’s footprint. Since Open Inventor [Wer94] was used as the basis for the graphics modules in the IRIS Explorer prototype, this was the library of choice. Using Coin’s implementation [COIN] of the Open Inventor API, the size of the visualization component was 3.1 Mb. This includes Coin libraries which will only need downloading a few times during the project; the executable alone is 0.1 Mb, which is a small proportion of 7.1 Mb, the current size of the whole participant package.

In choosing techniques for the display of simulation data, there was a requirement for images which would be easily understood by the participant, even in the absence of scientific knowledge. Considerations such as this (and some of the

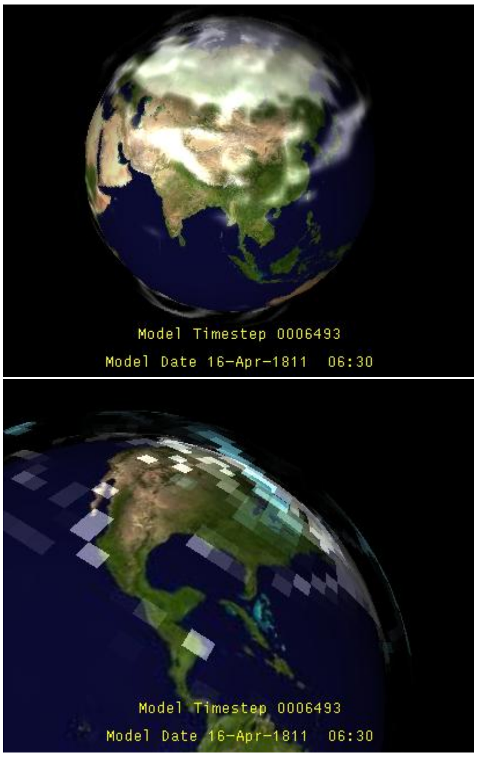

Figure 2: *Continuous cloud cover of entire globe (top) and a close-up (bottom) using cell-based display.*

work in §3.2) suggested mapping the simulation output onto a globe which could be manipulated in 3D. Two variants on this theme were developed, discussed below.

3.3.1. Cloud Display

Here, the globe is formed by texture-mapping a sphere with a satellite image of the earth's surface. Next, the snow/ice data is mapped onto a textured sphere concentric with the globe (the textured sphere's radius was slightly greater than the globe's to ensure that the snow/ice layer was always rendered on top of the earth's surface). The texture is a two-component (intensity, alpha) image, where the value for each component is given by the snow/ice coverage at that point. In a similar way, the cloud coverage for each height is mapped onto a textured sphere centred on the globe, with a radius scaled approximately by its height but representing a greatly exaggerated atmospheric depth that clarifies the separation of different levels. The superposition of textured layers around the globe gives a sense of depth in a pseudo-3D effect (see Figure 2). We found that this type of visualization of the cloud data was quicker to generate and manipulate than fuller alternatives such as calculating isosurfaces of the complete cloud dataset, whilst still producing images having a good degree of realism.

Texturing the sphere generates a smoothly varying intensity and transparency. An alternative representation, which reflects the resolution of the simulation, is to divide the surface of the sphere up using the underlying computational mesh, and to assign an intensity/alpha value to each cell based on data value. The result is shown in Figure 2.

One of the objectives of this type of display is to produce a caricature of the appearance of the earth from space, including its illumination by the sun. Incorporating the direction of sunlight in the visualization also highlights the diurnal variation in model results. To simulate sunlight, we add a directional light which rotates around the globe once every 48 time steps. The sub-solar latitude and longitude are calculated using

$$\phi = \zeta \cos(n_d - 201) \tag{1}$$

$$\theta = 180 + 360\frac{n_h + n_m/60}{24} \tag{2}$$

where ζ is the earth tilt angle ($= 23.5^\circ$), and n_d, n_h and n_m are the day, GMT hour and minute number. The model makes the standard assumption that each month has 30 days and each year is 360 days long; then the shift of 201 in (1) gives $\phi = 0$ at the model equinoxes, namely April 21 ($n_d = 111$) and September 21 ($n_d = 291$). Similarly, the 180 offset in (2) makes midday ($n_h = 12, n_m = 0$) correspond to the prime meridian ($\theta = 360 \equiv 0$).

3.3.2. Field Display

This type of scene is intended for the display of the remaining model variables such as surface temperature and precipitation. The first visualization (shown at the top of Figure 3) is similar to the cell-based cloud display (§3.3.1)—here, each cell is colored according to field value, while context is provided by a coastline overlay. An alternative method of displaying field data is to use solid contouring (bottom of Figure 3). This can be done [HS93] in OpenGL (or Open Inventor) by first creating a texture with a one-dimensional colormap and the field values as texture coordinates. Then, if the texture quality is set to a low value, the texture will be rendered with no interpolation between data values, resulting in solid regions of color for data within some range. The texture can be applied to a simple quadrilateral mesh that is then rendered offscreen to produce an image which is in turn texture-mapped around a sphere. Scaling the size of the offscreen viewport provides a tradeoff between image quality and rendering speed.

3.4. Interface And Performance

The update of the visualization is controlled by a timer sensor [Wer94] that periodically interrupts the display event

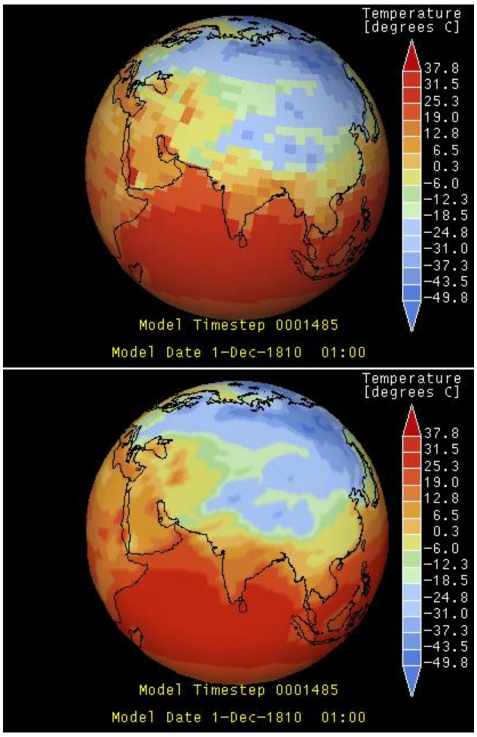

Figure 3: *Two options for visualizing field data—here, temperature—using (top) cell-based display and (bottom) solid contours.*

loop to check whether the time step has changed and rebuilds the scene using the latest data set if it has. Some consideration of the performance of the simulation is required in order to determine a sensible value for the timer interval. Besides being machine-dependent, performance details are sensitive to other machine activity since the model is running as an idle process, but a representative result may be quoted: on a 1.8 GHz Athlon XP, it takes about 195 seconds to simulate a day. It should be noted that every sixth time step takes about twelve times longer owing to an extra calculation of the radiation balance in the atmosphere; thus, the shorter steps take about one second, and this is the initial default value for the timer interval. As the visualization proceeds, it stores the moving averages of the shorter time per step, and uses this to adjust the timer interval to be commensurate with it. In practice, we have not observed any effect of the timer interrupt on user interaction with the visualization. Once a new data set has arrived, the time taken to rebuild the scene and to render it has been measured at around 0.1 sec on an SGI 540 with a Cobalt graphics card having 48 Mb of memory. The comparison of this time with the time step duration above is reinforced by more extensive beta testing of the package (§4) on a variety of hardware platforms, which shows that the visualization time is always much less than the calculation time.

The basic user interface for 3D display and manipulation is that of the generic Open Inventor Examiner viewer [Wer94], allowing rotation, zooming and panning through click-and-drag mouse actions. We have customized the viewer in two ways. Firstly, screen-based objects such as captions and legends (see Figure 3) have been introduced into the display by overriding the Examiner viewer's redraw method. Secondly, a popup menu in our component gives access to application-specific options like switching between display options for different fields and pausing the scene update. Our menu also contains some helpful generic options selected from the original Examiner viewer menu such as headlight control, stereo display and the storage and recall of a home viewing position (other options that might prove confusing to users are not exposed).

4. Current Status. Future Work

Following beta testing, the participant package (including the visualization component) was released at the formal launch of the project in September 2003. In order to encourage the participation of the general public in the experiment (the aim is to have the package running on around a million PCs worldwide), the organisers looked for—and got—extensive coverage of the launch event from both the print and broadcast media. The visualizations provided images describing the experiment which were very popular with the media (see, e.g. [BBC]).

Since the project launch, participants have suggested enhancements via the project website. Some which are currently under consideration include making the navigation through the 3D scene easier through the use of pre-loaded cameras to specify views of interest and adding a mode which only allows rotation about the globe's N-S axis. In addition, the display of other model fields such as surface pressure—possibly in composite displays with the cloud coverage—has been suggested.

Project development is contiuning with a changed emphasis after the release of the package. More work is being done on "sense-making" tools to help participants to understand the climate model results. Planned developments include:

- Customizable "user profile" pages at the website on which participants can view statistics and other information on their simulation runs.
- A semantic web browsing capability to help participants browse the site and the wider web by providing guidance and linking to their results.
- A secondary, downloadable visualization application which provides access to the diagnostic material held on the participant's disk.

This final point deserves amplification. Whilst we have concentrated in this paper on the implementation of the real-time visualization component in the participant package, this project has many further visualization requirements, particularly now that the post-processing analysis is under way. In this second phase of the project, the techniques from the real-time component will be re-used along with other applications. For example, the display of results from several sources will require the use of distributed visualization systems, probably with collaborative extensions for multi-user display [BDG*03, ESG]. And the use of the Grid as a platform for analyzing large distributed data sets will be able to build on other work linking visualization systems with Grid middleware [Gal02].

In parallel with these developments, the project will be extended to allow the embedding of regional models within the experimental architecture along with the inclusion of other climate models. The client will also be made available on other platforms such as Linux and Macintosh, probably by making use of the BOINC platform [BOI] which is aimed at public-resource distributed computing projects of this type.

5. Acknowledgements

Early work on the cloud display was carried out by Andy Heaps. We thank Richard Gillis, Carl Christensen and Richard Gault for help with system integration, Myles Allen for helpful discussions, and Rebecca Walton and an anonymous reviewer of an earlier version for useful comments on the text.

References

[BBC] `http://news.bbc.co.uk/1/hi/sci/tech/3100024.stm`. 5

[BDG*03] BRODLIE K. W., DUCE D. A., GALLOP J. R., WALTON J. P. R. B., WOOD J. D.: Distributed And Collaborative Visualization. In *Eurographics 2003 State Of The Art Reports* (2003), Montani C., Pueyo X., (Eds.), The Eurographics Association, pp. 139–166. 6

[BOI] `http://boinc.berkeley.edu/index.html`. 6

[COIN] `http://www.coin3d.org/`. 3

[Cul93] CULLEN M. J. P.: The Unified Forecast / Climate Model. *Meteorological Magazine 122*, 1449 (1993), 81–94. 2

[ESG] `http://www.earthsystemgrid.org/`. 6

[Gal02] GALLOP J.: gViz: Visualisation Middleware For E-Science: A New Project In The UK E-Science Programme. *ERCIM News 21* (October 2002). 6

[GFDL] `http://www.gfdl.gov/~jps/GFDL_VG_Gallery.html`. 3

[GLO] `http://www.globe.gov/`. 3

[HS93] HAEBERLI P., SEGAL M.: Texture Mapping As A Fundamental Drawing Primitive. In *Fourth Eurographics Workshop on Rendering* (1993), Cohen M., Puech C.,, Sillion F., (Eds.), pp. 259–266. 4

[KWA*01] KORPELA E., WERTHIMER D., ANDERSON D., COBB J., LEBOFSKY M.: SETI@home—Massively Distributed Computing For SETI. *Computing in Science and Engineering 3*, 1 (Jan–Feb 2001), 78–83. 1, 2

[SKA*02] STAINFORTH D. A., KETTLEBOROUGH J., ALLEN M. R., COLLINS M., HEAPS A., MURPHY J.: Distributed Computing For Public Interest Climate Modelling Research. *Computing in Science and Engineering 4*, 3 (2002), 82–89. 1, 2

[SKM*02] STAINFORTH D. A., KETTLEBOROUGH J., MARTIN A., SIMPSON A., GILLIS R., AKKAS A., GAULT R., COLLINS M., GAVAGHAN D., ALLEN M. R.: Climate*prediction*.net: Design Principles For Public-Resource Modeling Research. In *Proceedings of the 14th IASTED International Conference: Parallel And Distributed Computing And Systems* (2002), pp. 32–38. 2, 3

[STJ*00] STOTT P., TETT S., JONES G., ALLEN M., MITCHELL J., JENKINS G.: External Control Of Twentieth Century Temperature Variations By Natural And Anthropogenic Forcings. *Science 290* (2000), 2133–2137. 2

[Tre93] TREINISH L.: Visualization Techniques For Correlative Data Analysis In The Earth And Space Sciences. In *Animation And Scientific Visualization: Tools And Applications* (1993), Earnshaw R. A., Watson D., (Eds.), Academic Press, pp. 193–204. 3

[VETS] `http://www.vets.ucar.edu/vg/CSM/index.shtml`. 3

[VMET] `http://www.4dwx.org/vmet`. 3

[Wer94] WERNECKE J.: *The Inventor Mentor: Programming Object-Oriented Graphics with Open Inventor, Release 2*. Addison-Wesley, 1994. 3, 4, 5

[WG1] WORKING GROUP 1: *Climate Change 2001: The Scientific Basis. Third Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, 2001. 2, 2