



# THE WHITE ROSE GRID

## e-Science Centre

## Photorealistic Rendering of Synthetic Procedural Planets

### Introduction

The computer rendering of synthetic images with photorealistic quality is only possible with time consuming algorithms that correctly account for all the interactions between the light and the objects within a scene. Ray tracing is a photorealistic rendering algorithm that lends itself naturally to a grid implementation. A ray tracer computes every pixel in an image independently, making it easy to distribute the rendering work load. A distributed ray tracing solution on the White Rose Grid is here described. Our distributed ray tracer is applied to the generation of computer animations where a virtual camera flies over a procedurally generated terrain that defines the entire surface of a planet.



### Modelling Planets

Procedural models with fractal scaling properties can be used to generate the surface of a synthetic planet of any size. A planet with the same size as the Earth, for example, is easily handled. The terrain is specified by a function that returns the elevation  $f(x)$  for every point  $x$  on the surface of a smooth sphere. If the terrain generating function is fractal, it can generate continuously varying detail over all the surface of the planet and for widely different levels of detail, from an orbital altitude to an altitude of only a few centimetres above the ground. Different types of terrain can be obtained by mixing together several fractal functions, with each function being represented by a procedure in a computer program. Because the terrain is procedurally generated, no triangular meshes or other geometric primitives are required to represent the surface and the memory requirements are, therefore, minimal.

### Rendering Planets

The visualisation of procedurally defined terrains is performed with a ray tracing algorithm. For each pixel in the image, a ray is shot from the camera's viewpoint into the scene. The ray is checked for intersection with the terrain and a shading model is evaluated at the point of intersection to obtain the final colour for the pixel. If the viewpoint of the camera is made to follow a path through time, it is possible to generate a computer animation of a flyby over the terrain by rendering a sequence of image frames.

Because the terrain generating function has many small detail features, finding the correct intersection point between any ray and the terrain is a CPU intensive procedure. The following list states the algorithms, used in the photorealistic rendering of procedural planets, that have a significant contribution to the total rendering time of an image frame in a computer animation:

- A ray-terrain intersection algorithm that is guaranteed not to miss any intersection point.
- Anti-aliasing and motion blur algorithms that compensate for the discrete image sampling rate, according to the Nyquist sampling theorem.
- An atmospheric lighting algorithm based on a single-scattering Rayleigh model that accounts for the scattering of light due to air molecules.

### Ray Tracing on the Grid

Designing a grid distribution model for computer animations of ray traced images is a simple task because of the ability to render each image pixel independently. Furthermore, because we are dealing with procedural models of terrain, there is no need to transmit large geometric datasets.



All the procedural models describing the surface of the planet are already compiled into the renderer's executable code.

The distribution model comprises a meta-scheduler, written as a Python script, that runs on a desktop PC with access to the White Rose Grid through the SSH protocol. The meta-scheduler distributes all the image frames across the WRG nodes according to a dynamic load balancing strategy – as soon as a node finishes rendering a frame, it receives the next frame waiting to be rendered.

At every node, a frame is split into smaller rectangular regions called "tiles". Each tile is rendered independently on a single node processor. The set of all the tile rendering jobs for a frame is scheduled to the Sun Grid Engine as a job array. The SGE is then responsible for distributing the tile renderings across the available processors, based on the load conditions on the node.

The meta-scheduler controls the nodes through the remote invocation of shell scripts that must be present at each node.

These scripts are responsible for such tasks as the scheduling of the SGE job array and the retrieval of the individual tile rendering output files. The meta-scheduler implements a polling mechanism, where each node is polled every five minutes. As part of the

polling procedure, tile renderings from all the nodes are returned to the desktop PC and pasted together by the meta-scheduler to form the full frame renderings.

### Towards a Design Pattern

The design of the current grid rendering tool can be considered as a pattern for similar computing problems. In general, the current design solves computational problems that can be split into smaller and independent tasks. The pattern favours a two-tiered distribution model, where tasks can themselves be split into smaller and still independent sub-tasks. Tasks are distributed across grid nodes while sub-tasks are distributed among the processors of a node. This two-tiered model, however, is not a requirement. If tasks cannot be further split they can still be arbitrarily grouped. The task groups are then distributed to the nodes.

One example from Computer Graphics of another rendering algorithm where this design model can be applied is the Reyes image rendering architecture. A Reyes image renderer works by splitting the geometry (which can be made of polygon meshes, NURBS patches or subdivision surfaces) into progressively smaller fragments. When a fragment becomes much smaller than the size of a pixel it is called a *micropolygon*. It is then passed to the shading pipeline for rendering. Reyes is used by major animation studios such as Pixar for the rendering of their computer animation feature films. With Reyes, an image frame from an animation can again be partitioned into smaller tiles, with each tile being assigned to a processor. A node processor only handles the subdivision of geometric elements whose image projection is known to fall within the boundaries of the processor's assigned tile.

To go from a purely procedural modelling scenario, such as the one used to render synthetic planets, towards a more general geometric modelling scenario, such as the one used by the Reyes architecture, two steps need to be added to the current distribution model:

- Transmission of geometry data to the grid nodes prior to rendering.
- Execution of any required geometry related pre-computation steps prior to distributing the tile rendering jobs to the processors of a node.

### Some Background Information

This project was developed as part of a PhD in procedural planet modelling. The research was performed in the Computer Graphics research group at the Department of Computer Science, The University of Sheffield. Other research interests of the group include facial animation and modelling, visual speech synthesis, processing of motion capture data and Virtual Reality technology for training.

### Further Information

Contact:  
Manuel Gamito (mag@dcs.shef.ac.uk)

Relevant sites:

<http://www.dcs.shef.ac.uk/~mag/grid.html>

<http://www.shef.ac.uk/dcs/research/groups/graphics>

**Ray tracing is a photorealistic rendering algorithm that lends itself naturally to a grid implementation.**

**The design of the current grid rendering tool can be considered as a pattern for similar computing problems.**