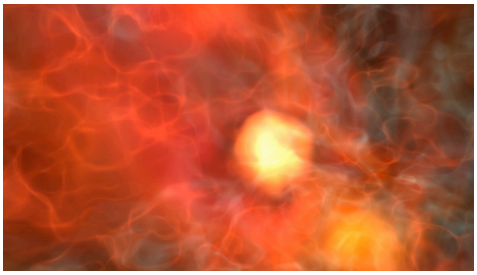
Animation of Adaptive Mesh Refinement Data

Matthew Hall David Bock Donna Cox Lorne Leonard
National Center for Supercomputing Applications
{mahall,dbock,cox,leonard}@ncsa.uiuc.edu

1: Introduction

We present a method for visualizing a time-varying Adaptive Mesh Refinement (AMR) simulation. AMR methods are increasingly being used in scientific simulations due to their ability to handle very large differences in scale (both spatial and temporal) in an efficient manner, but they are difficult to animate. We discuss our solution, as well as our AMR direct volume renderer. These techniques were used to produce the first star and supernova sequence in the digital planetarium show "Black Holes: The Other Side of Infinity" for the Denver Museum of Nature and Science, which opened February 2006.[1]



Formation of a Population III Star. Dataset is 23 levels deep, with an effective resolution of $(8*10^8)^3$ in the most refined regions.

2: Temporal Interpolation

Briefly, an AMR dataset consists of a set of nested, uniform grids, where both the spatial and temporal resolution increase as the nesting level increases. This partitioning of the domain into nested grids is not static; new grids form and disappear over the lifetime of the simulation, and the finer grids change more frequently than the coarser grids.[2] This poses problems for animation; often the coarser grids are changing more slowly than the animation rate, so without interpolation the animation will be smooth in the refined regions of space, but unacceptably jumpy elsewhere.

To interpolate the AMR data to some time ***t***, we must first find the temporally proximate data to use for our endpoints (assuming linear interpolation; we can also extend this method for higher order interpolation). This cannot be done deterministically; instead, to find the most recent state of the simulation before ***t***, we scan through all grids up to ***t***, using a kd-tree to track the most recent update for every region of space. We call the set of grids which contribute to recent state the *before set.* A similar procedure is carried out to find the *after set.*

Once these two grid sets are created, we need to interpolate between them. Unfortunately, they are unlikely to have the same nested grid structure, so that a grid in one set might intersect several grids in the other set [Kaehler 2005]. We can make a uniform structure by partitioning the grids in each set into subgrids, so that there is a one-to-one correspondence between the spatial extents of subgrids in the before set and those in the after set. We call this process finding the *common homogenization* of the two sets, and it is an extension of the grid homogenization technique described in [Kreylos *et al*. 2002]. In a simplified version of this procedure, we insert all grids from the before set and after set into a kd-tree, in order of increasing resolution. The result is that the domain is partitioned into a set of non-overlapping boxes, where each box lies entirely within a single grid in the before step, and also within a single grid in the after step. Thus, for each box, we can extract a subgrid from each of the before and after steps. While these two subgrids have the same spatial extents, they may not have the same resolution, so we super-sample the coarser of the two grids to match the resolution of the finer. Interpolation is then straightforward.

3: Rendering

Frequently, AMR output is cell-centered, while for rendering we prefer vertex-centered data. In a unigrid, one can map the data on to the dual grid to change the centering. However, as taking a dual grid shrinks the domain of a grid by a half-cell on each side, this can create unsightly gaps between neighboring AMR subgrids. This can be remedied by various methods such as resampling or grid stitching [Kreylos *et al*. 2002], however, we chose a method more suited to ray casting, dubbed the *expanded dual*. In the expanded dual we take the dual of each subgrid, and extend it by one cell in each direction. The value at each new boundary cell is either copied from a neighboring grid of the same resolution, or if none exists, trilinearly interpolated from the data in the next coarser level. This results in a modest increase in data size, but the process is simple and time-efficient.

Once we have an interpolated, vertex-centered dataset, we render it using a ray casting volume renderer. We first perform a homogenization of the dataset, using a kd-tree. For each ray cast, we traverse the intersecting subgrids in back-to-front order. For each subgrid that the ray intersects, we then perform the standard direct volume-rendering algorithm. However, since the different subgrids have different resolutions, we must adjust the distance between samples in order to capture detail at finer levels. We then get an image that is highly detailed in the refined regions, and smooth elsewhere.

References

KAEHLER R, PROHASKA S, HUTANU A, HEGE. Visualization of Time-Dependent Remote Adaptive Mesh Refinement Data. In *Proceedings of IEEE Visualization 2005*. 175-182.

KREYLOS O, WEBER G, BETHEL E, SHALF J, HAMANN B, JOY K. 2002. "Remote interactive direct volume rendering of AMR data" Lawrence Berkeley National Laboratory. Paper LBNL-49954.

[1] http://www.dmns.org/main/en/Professionals/Press/CurrentPressReleases/Press+Releases/Planetarium/blackHolesImagesPress.htm

[2] See the Supplemental Material for more details