Joint Eurographics - IEEE TCVG Symposium on Visualization (2002)
D. Ebert, P. Brunet, I. Navazo (Editors)

# View-Dependent Multiresolution Splatting of Non-Uniform Data

Justin Jang, William Ribarsky, Christopher D. Shaw and Nickolas Faust

jang,ribarsky,cdshaw@cc.gatech.edu, nick.faust@gtri.gatech.edu
GVU Center, College of Computing, Georgia Institute of Technology, 801 Atlantic Drive, Atlanta, Georgia, 30332, USA

**Abstract**
*This paper develops an approach for the splat-based visualization of large scale, non-uniform data. A hierarchical structure is generated that permits detailed treatment at the leaf nodes of the non-uniform distribution. A set of levels of detail (LODs) are generated based on the levels of the hierarchy. These yield two metrics, one in terms of the spatial extent of the bounding box containing the splat and one in terms of the variation of the scalar field over this box. The former yields a view-dependent choice of LODs while the latter yields a view-independent LOD based on the field variation. To show the utility of this general approach it is applied to a set of application data for a whole earth environment and some test data. Performance results are given.*

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Display Algorithms. Volume Rendering, Splatting, Multiresolution, View-Dependent, Visualization

## 1. Introduction

This paper presents the following results. It develops an approach for organizing multiresolution splats of 3D data into a hierarchical structure. The hierarchical structure and the multiresolution splats are generated automatically for a given dataset. The structure is detailed in terms of a global, earth-based hierarchy. However, the methods are general and applicable to other types of data distribution. A view-dependent procedure permits on-the-fly selection of levels of detail (LODs) based on the user viewpoint, using an adjustable screen space error metric. This results in efficient display with emphasis given to important or visually prominent details. The method is extended to non-uniform data distributions, including those containing incommensurate parts and distinctly different topologies. Lastly, the approach is applied to non-uniform 3D weather patterns with holes and to various test datasets.

We use splatting because it is efficient and lends itself to a multiresolution framework. Splats can be treated independently and, if desired, without reference to the underlying data topology. Thus splats can be applied to irregular volume data sets [20, 10] and even to unstructured points with no explicit topology. (However, as discussed below, optimal splatting would take into account the local distribution of data points around the splat.)

### 1.1. Non-uniform data

Increasingly there are sources that produce non-uniform 3D data. This is especially true for digital data acquired from measuring devices or sensors. But it is also true for computer simulations such as finite element calculations (e.g., for a helicopter) that employ multiple, interlocking curivlinear meshes. The goal is ultimately to view relevant data simulaneously, such as real acquired data and virtual simulation results that may have completely unrelated data distributions.

To visualize non-uniform data in the form of unstructured grids, the approach is usually to resample on a regular grid [16, 18]. However, there are data for which the details of the original distribution are important. For the Doppler radar data discussed below, the signature of a severe storm cell or even a tornado may be contained in a few sample points. The same is true for a computational fluid dynamics simulation of combustion in an engine, where the flame front may be only a few sample points wide and quite non-flat. A regular sampling that would capture these features will typically produce lots of extra points, even in adaptive, hierarchical approaches where higher resolution samples are displayed only where needed. A more efficient alternative is to use a sampling that depends on the local distribution of the original data. The trade-offs here are similar to those in using regular triangulations [6] versus triangular irregular networks [4] to represent complex surface geometry.

Ideally, one would like an approach that automatically decides when to perform a regular resampling of the grid at lower resolutions and when to reveal the details of the non-uniform structure at high resolution. To do this, we have chosen a view-dependent hierarchical structure that is regular in its upper part but reveals the non-uniform distribution of the original data in its lower part. A multiresolution splatting structure fits naturally into this hierarchical scheme. In addition the scheme can be extended to the fusion of different types of data structures for simultaneous visualization.

## 2. Previous Work

Splatting [17] is a technique where a continuous volume is represented for the purpose of rendering by the compositing of independent footprint functions. The footprints are projections onto screen space of 3D kernels centered on the sample points in the volume dataset. This method can provide high quality images and is efficient with respect to other direct rendering algorithms (e.g., ray-casting) since it uses 2D convolution of samples along the viewing ray instead of 3D convolution, such as in ray-casting. Since they are precomputed, the splat kernels can also be of higher quality than are typically used in ray-casting methods. Recently Zwicker et. al. [20] have developed a general framework for splatting that removes aliasing artifacts without excessive blurring and is suitable for regular or irregular volume datasets. This work is based on a similar but less general approach developed by Swan et. al. [15] The method of Zwicker et. al. derives a significant part of its efficiency from the use of elliptical Gaussian kernels for which one can derive an analytic expression of the footprint in screen space. There has been relatively little work applying splatting to non-uniform data. Ma et.al. apply spherical splats to the visualization of 3D shock waves on unstructured grids [9]. They center the spherical splat on each grid point and use a radius that is the average distance to the nearest neighbors. This can be a gross approximation for highly non-uniform data. An accurate splat would reflect the shape of the corresponding local region around a data point (or, generally, the enclosing Voronoi cell). Since this shape is not elliptical in general, it cannot be treated with existing compositing algorithms [20]. An alternative is to place additional points in the neighborhood of the existing data point, taking into account the local data distribution, and place elliptical Gaussians at these points. The criterion is to provide uniform coverage over the space of the volume. Mao [10] has done this using a 3D generalization of a stochastic Poisson disk algorithm. This approach gives good results when applied to multigridded hexahedral or tetrahedral datasets. However, the stochastic resampling step is quite time-consuming. Also, the resulting resampled dataset can be a factor of 20 or more larger than the original non-uniform dataset. Instead of a stochastic method, we use a much faster procedural method for placing additional splats in the neighborhood of each non-uniform data point. Added points can also be aggregated for multiresolution rendering. Axis-aligned sheet buffers [17] are often used in splatting to improve efficiency and remove artifacts such as color bleeding. The latter is due to the independent splat approximation in areas where splats overlap. However, sheet buffers introduce popping artifacts, especially noticeable during interactive rotation from one set of sheet buffers to another. Recent work removes the popping artifacts by using sheet buffers parallel to the image plane instead of parallel to the data axes [12]. However, sheet buffers are less appropriate for non-uniform data, since they amount to a regular resampling where high resolution details may be lost, as discussed above. An alternative is to slice the splats along the direction perpendicular to the image and interleave the slices. Hierarchical techniques can improve volume rendering interactivity through progressive refinement using estimated errors stored at each level of the hierarchy [5, 19] Ma et al [8] reported experimental results in which they examined the performance benefits of various types of quantization in a system that stored time-varying volume data in a sequence of octrees. Sutton and Hansen [14] have shown that a flexible branch-on-need tree (TBON) can efficiently accommodate temporal changes in the data. These approaches and most others consider uniform data. Our hierarchical approach applies to non-uniform data and is extended to incorporate view-dependence, permitting the system to efficiently and continuously present the best view as the user viewpoint changes. In addition the approach provides a general, adaptive tree structure that is generated automatically and is appropriate for fusing incommensurate datasets. Although we do not consider time-dependent data in this paper, our tree structure can be extended to the TBON [14] or other time-dependent structures.

## 3. Embedded, Adaptive Volume Tree Structure

We present a method for automatically generating a volume tree structure in an environment where there may be different types of data with different organizations and several orders of spatial magnitude difference between overviews and close-ups of important details. We particularize to earth-based data in this paper, but the method is general. At the top level, the structure has an overall hierarchy that is effective in supporting all the different types of data. At mid level are volume trees that align with top-level nodes and may differ in different regions depending on the data organization in the regions. At the bottom the structure depends on the details of the non-uniform data distribution. This structure is shown for earth-based data in Fig. 1. Here the top-level structure is a forest of quadtrees that covers the entire earth [3]. The mid-level is a volume tree aligned with a quadtree node. At the bottom level the leaf nodes contain zero, one or two non-uniform data points.

The structure in Fig. 1 turns out to work well for atmospheric data. The atmosphere is a thin layer with respect to the earths ellipsoid, so it is appropriate to use a lat/lon quadtree until the lat/lon dimensions are of the same order of magnitude as the altitude dimension. After that we switch to a volume tree. The quadnode is divided into $N_x \times N_y \times N_z$ bins where $x, y, z$ are the longitude, latitude, and altitude directions, respectively. The bin sort of non-uniform data points is fast ($O(n)$ where $n$ is the number of points). This is a key step because the bins provide a structure that is quickly aggregated into a hierarchy for detail management and for view frustum culling. However, the data element positions are retained in the bins for full resolution rendering, if needed. The hierarchy provides significant savings in memory space and retrieval cost since only data element coordinates for viewable bins at the appropriate LOD are retrieved. Note that the bins are not rectilinear in Cartesian space. At present we ignore this effect in performing the volume rendering because the earths curvature is small for the bins rendered and lat/lon quadcells are nearly rectilinear except at high latitudes. In general, the bin widths in each direction are non-uniform (e.g., each of the bins in the, say, $N_z$ direction may have a different width). This gives useful flexibility in distributing bins such as, for example, when atmospheric measurements are concentrated near the ground with a fall-off in number at higher altitudes.

Each of the dimensions $N$ in the $x, y, z$ directions is a power of 2. This permits straightforward construction of a volume hierarchy that is binary in each direction. Our tests show that this restriction does not impose an undue hardship, at least for the types of atmospheric data we are likely to encounter. The number of children at a given node will be 2, 4, or 8. If all dimensions are equal, the hierarchy is an octree. Typically the average number of children is between 4 and 8. We restrict the hierarchy to the following construction. (Others are possible.) Suppose that $N_x = 2^m$ , $N_y = 2^n$ , $N_z = 2^p$ where $m > n > p$. Then there will be $p$ 8-fold levels (i.e., each parent at that level has 8 children), $n - p$ 4-fold levels, and $m - n$ 2-fold levels. If two out of three exponents are equal, there will be only 8-fold and 4-fold levels. The placement of 2, 4, and 8-fold levels within the hierarchy will depend on the distribution for the specific type of volumetric data.

The above bin size is determined by the minimum spacing of the underlying data points. As stated above, there are no more than two points per bin. The reason for this choice is that we want a smooth transition between rendering of bin-based levels of detail and rendering of the raw data. If the data are distributed in well-separated clumps, an immense number of regularly spaced bins is required. However, in our procedure only filled bins are stored and accumulated into the tree. Where there are no filled bins, there are no tree branches. As the bins are accumulated, the properties at parent nodes are derived from weighted averages of child properties. The parent also carries the following factors: (1) the total raw volumetric data elements contained in the children; (2) the total filled bins contained in the children; (3) the average scalar field value and its variance. The dimensions of each node in the tree can easily be calculated on-the-fly and are not stored. Note that the bin structure and volume hierarchy are static in space. However, we can efficiently apply this structure even to distributions of volumetric elements that move in space as long as the range of local spatial densities (and the volume of the data) does not change much over time.

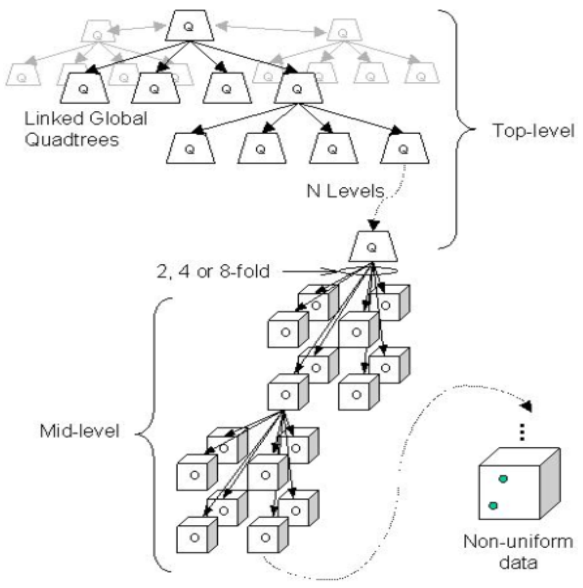

**Figure 1:** *Embedded, adaptive volume tree in global earth environment.*

## 4. View Dependence

The above hierarchical structure provides efficiencies in rendering the volumetric data. Adjusting an error metric derived from the hierarchy (e.g., in terms of the RMS deviation of scalar values contained in a node from their average value) permits one to select from the multiple resolutions contained in the hierarchy. One could use a larger error for initial interactive visualization of the volume and then, when pausing for more detailed inspection, smaller errors for progressive refinement of the volumetric rendering. Such a procedure could also be employed for compression and then progressive transmission of the volumetric data. Similar approaches have been developed by others [5, 19]. The hierarchy also permits quick culling, as the viewpoint changes, of volume elements that are not in the view frustum.

Interactive navigation of large scale data requires additional efficiencies. Here some data elements may be close to the viewpoint, some at midview, and others far away from the viewpoint. The rendering system should take into account the projection of these elements onto the screen and thus their perceivability to the user. Several neighboring elements that are small or far away may fall within a single pixel. Such elements should be rendered at lower resolution, resulting in an image that displays several LODs at once. To be effective such a set of LODs should be updated every time the viewpoint changes, which can be every frame for continuous navigation. To achieve these goals, we will extend hierarchical splatting by using a variation of the view-dependent approach that has been applied by us [6] and others [7, 4] to surface polygonal data.

The view-dependent approach needs a measure of spatial extent, which is provided in the volumetric case by the cell size of the nodes in the hierarchy. The non-uniform data points in the leaf nodes are handled separately. (See below.) Using the notation of the last section, the cells are rectilinear with sides $L_x$, $L_y$, $L_z$. Suppose $L_z$ is the largest dimension. Then, depending on orientation, the screen projected linear footprint of the cell can have a value ranging from $(L_x^2+L_y^2)^1/2$ to $(L_x^2+L_y^2+L_z^2)^1/2$. (See Fig. 2) One could store the orientations of the axes for each of these sides and, using the current viewpoint location, compute the precise footprint for each cell. For the earth-based case, not much information need be stored (at the expense of some computation to find the current cell orientation) since the cells are regular and everywhere along a radial from the earths center. Currently we have chosen to forego extra computation and use a fixed linear footprint for all orientations, conservatively set at $(L_x^2+L_y^2+L_z^2)^1/2$. A simple analysis of edges visible in various orientations shows that footprints near this size are more likely than those near, say, $(L_x^2+L_y^2)^1/2$ in size. This choice also generalizes to other hierarchical structures that may not be so uniform in orientation. The footprint is then perspective projected onto the screen (see Fig. 2), converted to pixels, and compared to a screen space error metric. Testing is done from parent to child in the hierarchy until the footprint falls below the error metric. Performance results in the Application section below show that the fixed linear footprint is not overly expensive in terms of extra splats rendered. An error metric of 1 pixel should result in an image that is barely distinguishable from one rendered with high resolution splats, depending on how close using an averaged splat is to compositing.

We now have two error metrics controlling the rendering speed and quality, a screen space metric and a scalar value error metric. The screen space metric addresses pixel-based visual quality. The scalar value metric determines the resolution of detail in the scalar field. As discussed above, the latter could be dynamically adjusted when one pauses at a certain view. It could also be keyed to ranges in the scalar field, being made smaller for those ranges deemed more important. Finally it could be coupled to a viewing device such as an interactive magic lens [1], bringing out greater detail in the lens subview.

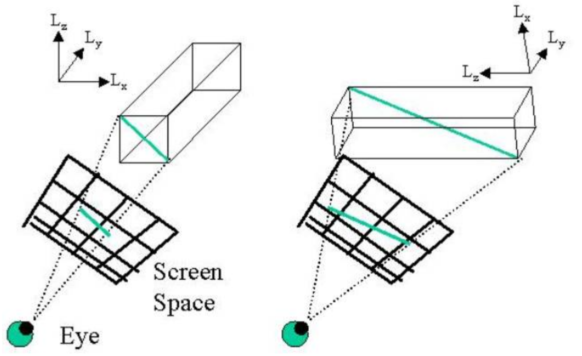

**Figure 2:** *Screen projection of a hierarchy cell at different orientations.*

It is important to note that a choice of view-dependent metric greater than or equal to one pixel will have the effect of removing aliasing in the sense of Swan et. al. [15], but with fewer splats. There will be in general an error associated with compositing an averaged splat, but this will be confined to a space of 1 pixel or less for a metrics less than or equal to 1 pixel. The nature of this approximation and how to adjust the averaged splat to minimize it will be the subject of a future paper. However, for our application below a screen space error of 1 pixel causes minimal differences from full detail rendering. Also, the error can be adjusted on-the-fly. As part of future work we will also look at mechanisms for combining the two metrics into one.

## 5. Splat Structure

Several approaches have been applied to splat footprint construction and compositing. Polygon and texture-based methods [11, 5, 13] using 2D or 3D textures can take advantage of graphics hardware to composite and render overlapping polygons. These approaches work well for regular data. However, for non-uniformly distributed data, one must either constrain to uniform bounding boxes for the splats (Meredith et. al. use cubic boxes [11] or face significant complications in correctly projecting, sampling, and rendering non-uniform splats. Using regular bounding boxes amounts to an approximation that smooths out non-uniformities. As stated at the beginning of the paper, we would like an approach that permits retention of these details, although uniform splats may also be used.

We have chosen to construct splats that are not necessarily uniform in their orientation or aspect ratios. However they all fit into rectilinear bounding boxes. The approach of Zwicker et. al. [20] permitting efficient perspective projection and correct anti-aliasing for arbitrary elliptical kernels can thus be applied. Other approaches, including texture-based methods,

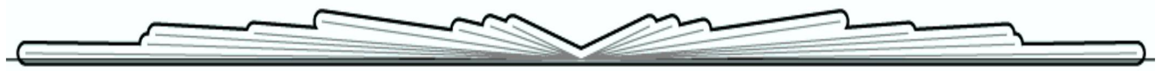

**Figure 3:** *Side cut-away view of a single Nexrad radar. The 9 gray sweep lines are surrounded by regions of higher confidence in the accuracy of the readings, At lower elevations, spacing between the sweeps is close enough to assume readings from a continuous medium.*

can be applied as well. Using splats of non-uniform orientation and aspect ratio requires more computation than uniform splats. However, the view-dependent hierarchical approach helps keep the splat construction and rendering efficient. Interactive techniques, such as magic lenses, can also be applied; the hierarchy helps with quick retrieval of data in this case.

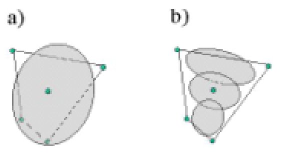

**Figure 4:** *Splatting in a 2D cell formed by nearest neighbors of central point.*

The issue for a non-uniform distribution is illustrated in Fig. 4. (This would describe the non-uniform regions at the leaf nodes of the above hierarchical structure.) A single elliptical splat, even if optimally oriented, cannot provide uniform coverage for all the neighbors of the central data point. For example in Fig. 4a if it provides correct coverage for the top neighbors, it overlaps the bottom neighbors too much. On the other hand, multiple elliptical splats (Fig. 4b) can be oriented and sized to provide better coverage over the cell. We have developed and are implementing a procedure, based on the Voronoi cell describing the region around a point, to place and orient splats to give more uniform coverage in non-uniform cells. Closest splats are combined to give LODs. At the moment we use elliptical, textured splats centered at the sample points with appropriate scale and orientation. The splat construction and compositing could be replaced with that of Zwicker et. al. [20], which provides and accurate, analytic expression of the integration of the footprint in screen space.

## 6. Application Results

In order to evaluate our approach, we have organized and rendered a variety of test and application data. The main application data are from Nexrad 3D Doppler radars. The volume is collected as a series of conical scans at increasing angle (Fig. 3). To our knowledge, the only work that explicitly acknowledges the 3D structure of Nexrad and grapples with its anisotropy is by Djurcilov and Pang [2]. They explored the creation of smooth isosurfaces in the context of missing data and irregular grids. The focus of their work is on the rendering of a collection of cones from a single Nexrad station, accurately reflecting the absence of measurement data, with the additional display of uncertainty information. Because of the curvilinear nature of the radar scan (Fig. 3), the data can be both sparse and blurred when usual visualization techniques (i.e., isosurfaces and volumetric rendering) employing regular grids are applied. Further there can be holes in the data produced by regions of null readings. Finally there can be multiple overlapping radars. (For example, there are 3 overlapping radars covering North Georgia, and 9 covering central Oklahoma.) The overlapping areas can provide regions of increased spatial complexity and non-uniformity.

Among the products collected by the Nexrad radars are reflectivities and wind velocities. The reflectivities determine the amount of precipitation in a sampled region and even the type of precipitation. The wind velocity is a line-of-sight reading; high positive/negative values for adjacent points indicate the presence of high shear and can be part of the signature of a tornado. The tornadic signature can be corroborated by the 3D structure of the reflectivity and velocity data in the region.

Fig. 8 (left) shows a volume rendering of the reflectivity readings from a NEXRAD Doppler Radar covering North Georgia from a severe storm that passed over the Atlanta area in late March 1996. The red and yellow band running North-South through the central region is an area of very heavy rainfall. The color bars to the bottom left show the transfer function from reflectivity to RGBA values. The left bar is RGB and the right two bars show RGBA blended with black and white. Fig 8 (right) shows the velocity readings of the same storm at the same time. The major feature running from Northwest to Southeast is the line perpendicular to the wind direction for the entire storm. Because Doppler radar gives velocity only along each radial line, there will be a sign change in velocity values where the radar is perpendicular to the wind direction. In Fig. 8 right, the storm is heading Northeast. We have set up the color ramp to change abruptly from cyan (positive) to blue (negative).

Fig 9 shows a close-up of the velocity dataset with small orange and yellow circles drawn to indicate mesocyclones that were detected by WDSS, which is Doppler radar weather analysis software.

## 7. Performance Results

Table 1 shows the performance of our screen-space metric being used to select the level of detail to render. The screen-space error metric in this case is 2 pixels. As we move from an altitude of 2667 km down to an altitude of 4km, greater and greater levels of detail are selected using the algorithm presented above. The image size being rendered is 800 x 600 pixels, and the computer user to render these images was an SGI 195MHz R1000 with Infinite Reality 2 graphics. Rendering times reach a maximum of 1.7 seconds before view frustum culling starts to eliminate a significant amount of the radar from the view.

| Image | Maximim Octree Level Rendered | Splats Rendered | Time (S) |
|---|---|---|---|
| 1 | 11 | 9,681 | 0.072 |
| 2 | 12 | 12,414 | 0.117 |
| 3 | 12 | 44,488 | 0.346 |
| 4 | 13 | 121,393 | 1.080 |
| 5 | 14 | 152,331 | 1.681 |
| 6 | 15 | 151,898 | 1.705 |
| 7 | 16 | 112,697 | 1.307 |
| 8 | 17 | 65,885 | 0.854 |
| 9 | 18 | 28,965 | 0.423 |
| 10 | 20 | 10,129 | 0.174 |

Fig. 6 shows the first 5 corresponding images. We have cropped out the non-weather imagery to save space.

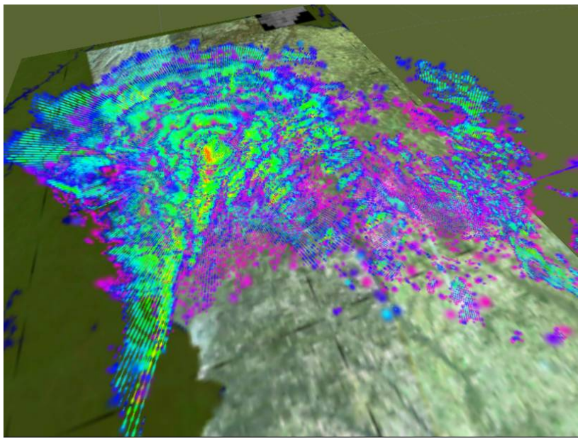

**Figure 5:** *Two overlapping radar volumes. The new radar has been added 150 miles southeast of the original radar.*

Fig. 5 shows two radars volume rendered together by our system. The data is in fact another volume scan timestep from the same series of radar volumes from March 1996. In overlap areas, overlapping reflectivity readings are treated as adjacent reflectivities from the same radar, and are therefore rendered using the same algorithm. Our intent is to be able to support any number of radars overlapping, and to be able to draw them in a seamless manner.

## 8. Conclusions and Future Work

We have demonstrated an interactive view-dependent multiresolution volume rendering system for non-uniform data. The octree structure we have built allows the user to select the appropriate level of detail that they want to see, and the system automatically renders to the required resolution. Fig. 7 shows the screen-space rendering system at work with the metrics of 4, 8, 16, and 32 pixels. Rendering times in this case were 1.26sec, 0.328sec, 0.063sec and 0.014sec respectively, indicating that we are well within the range of real-time for viewing volume rendered approximations of the data. In situations where the user will navigate quickly around the scene, we will set the screen-space size to be quite large as that few splats are drawn quickly. Note that the volume-rendered image maintains is overall character, even at the 32-pixel metric.

Our next step with this work is to build the volume rendering system into our multi-threaded terrain rendering system VGIS [3]. The octree system was designed to correspond to VGISs quadtrees, so we anticipate that combining volume-rendered radar weather and out-of-core LOD terrain rendering will proceed with only modest effort.

## 9. Acknowledgements

This work was performed in part under a grant from the NSF Large Scientific and Software Data Visualization program and under a MURI grant through the Army Research Office.

## References

1. Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony DeRose. Toolglass and Magic Lenses: The see-through interface. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 73–80, August 1993.

2. Suzana Djurcilov and Alex Pang. Visualizing gridded datasets with large number of missing values. In *IEEE Visualization '99*, pages 405–408, San Francisco, California, October 1999. IEEE.

3. N. Faust, W. Ribarsky, T.Y. Jiang, and T. Wasilewski. Real-Time Global Data Model for the Digital Earth. In *Proceedings of the International Conference on Discrete Global Grids (2000), Rep. GIT-GVU-01-10*, 2000.

4. Hugues H. Hoppe. Smooth View-Dependent Level-of-Detail Control and its Application to Terrain Rendering. In David Ebert, Hans Hagen, and Holly Rushmeier, editors, *IEEE Visualization '98*, pages 35–42, 1998.

5. David Laur and Pat Hanrahan. Hierarchical splatting: A progressive refinement algorithm for volume rendering. In Thomas W. Sederberg, editor, *Computer Graphics (SIGGRAPH '91 Proceedings)*, volume 25, pages 285–288, July 1991.

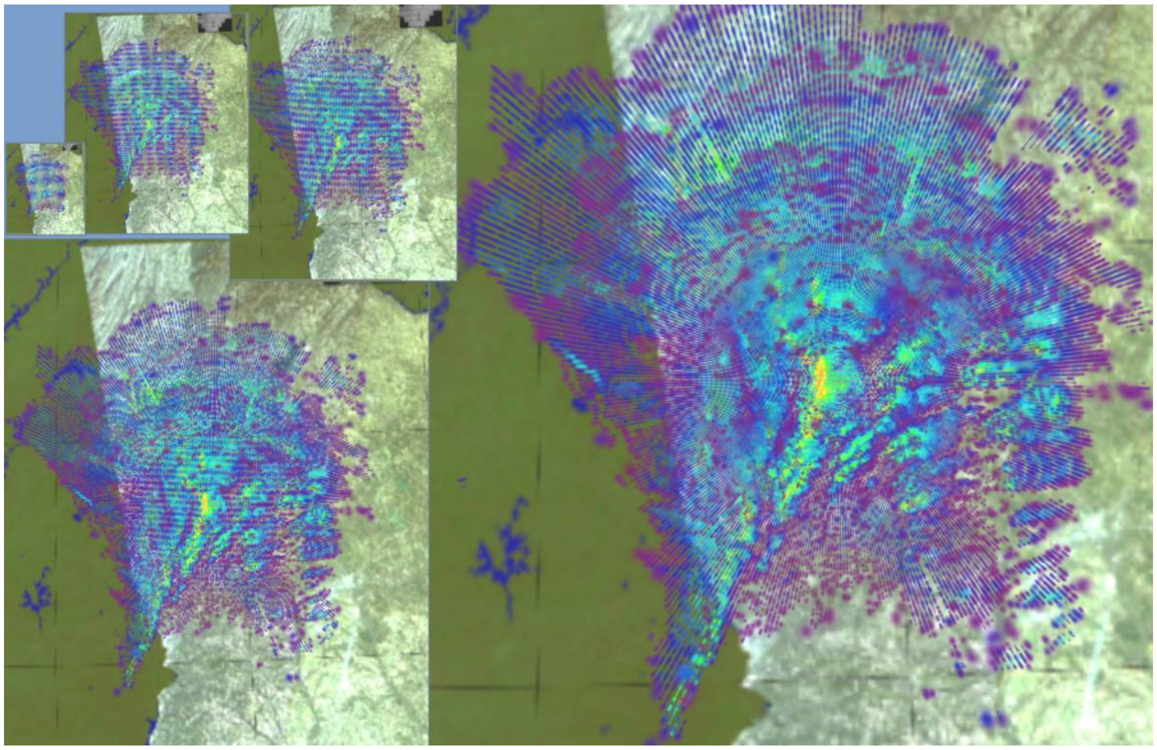

**Figure 6:** *Screen-space selected Level-of-Detail renderings of Doppler reflectivity as the viewpoint moves from 2667 km to an altitude of 229km above North Georgia. These images correspond to lines 1-5 in table 1.*

6. Peter Lindstrom, David Koller, William Ribarsky, Larry F. Hughes, Nick Faust, and Gregory Turner. Real-Time, continuous level of detail rendering of height fields. In Holly Rushmeier, editor, *SIGGRAPH 96 Conference Proceedings*, pages 109–118. ACM SIGGRAPH, August 1996.

7. David Luebke and Carl Erikson. View-dependent simplification of arbitrary polygonal environments. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, pages 199–208. ACM SIGGRAPH, August 1997.

8. Kwan-Liu Ma, Diann Smith, Ming-Yun Shih, and Han-Wei Shen. Efficient Encoding and Rendering of Time-Varying Volume Data. Technical Report ICASE Technical Report 98-22, NASA/CR-1998-208424, ICASE, Hampton, Virginia, 1998.

9. Kwan-Liu Ma, John Van Rosendale, and Willem Vermeer. 3D shock wave visualization on unstructured grids. In *1996 Volume Visualization Symposium*, pages 87–96. IEEE, October 1996. ISBN 0-89791-741-3.

10. Xiaoyang Mao. Splatting of non rectilinear volumes through stochastic resampling. *IEEE Transactions on Visualization and Computer Graphics*, 2(2), June 1996.

11. Jeremy Meredith and Kwan-Liu Ma. Multiresolution view-dependent splat based volume rendering of large irregular data. In *IEEE 2001 Symposium on Parallel and Large-Data Visualization and Graphics*, pages 93–100, 2001.

12. Klaus Mueller and Roger Crawfis. Eliminating popping artifacts in sheet buffer-based splatting. In *IEEE Visualization '98*, pages 239–246. IEEE, October 1998.

13. S. Rottger, M. Kraus, and T. Ertl. Hardware-accelerated volume and isosurface rendering based on cell-projection. In *IEEE Visualization 2000*, pages 109–116, October 2000.

14. Philip M. Sutton and Charles D. Hansen. Isosurface extraction in time-varying fields using a temporal branch-on-need tree (t-bon). In *IEEE Visualization '99*, pages 147–154, San Francisco, California, October 1999. IEEE.

15. J. Edward Swan, II, Klaus Mueller, Torsten Möller, Naeem Shareef, Roger A. Crawfis, and Roni Yagel. An anti-aliasing technique for splatting. In *IEEE Visualization '97*, pages 197–204, 1997.

16. M. Weiler and T. Ertl. Hardware-software-balanced resampling for the interactive visualization of unstruc-

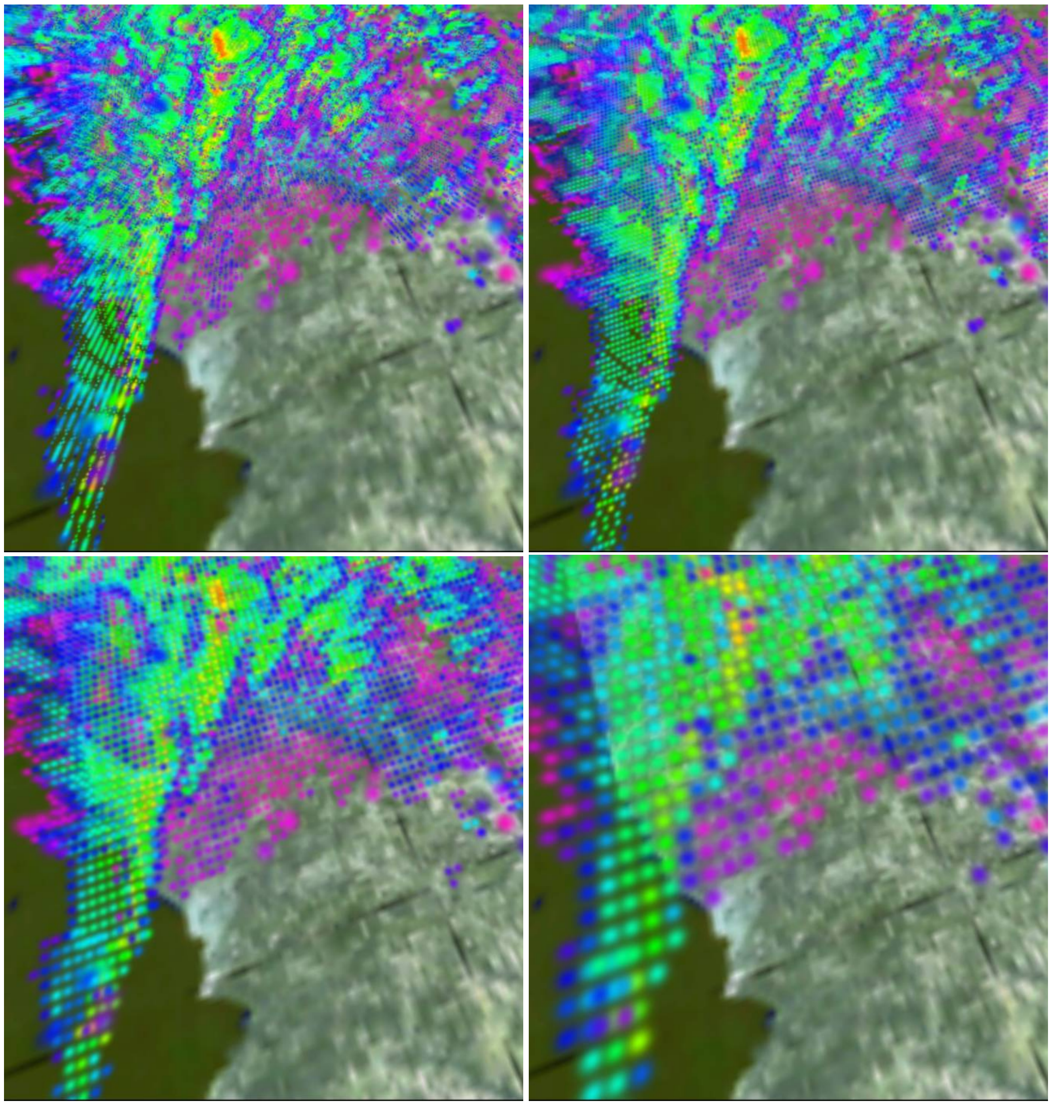

**Figure 7:** *Levels of detail. The four images from left to right are images drawn with 4, 8, 16 and 32 -pixel screen-space error metrics. The images are from the same dataset as Fig. 5, but cropped.*

tured grids. In *IEEE Visualization 2001*, pages 199–206, October 2001.

17. Lee Westover. Footprint evaluation for volume rendering. In Forest Baskett, editor, *Computer Graphics (SIGGRAPH '90 Proceedings)*, volume 24, pages 367–376, August 1990.
18. Jane Wilhelms. Pursuing Interactive Visualization of Irregular Grids. *Visual Computer*, 9(8):450–458, 1993.
19. Jane Wilhelms and Allen Van Gelder. Multi-dimensional trees for controlled volume rendering and compression. In Arie Kaufman and Wolfgang Krueger, editors, *1994 Symposium on Volume Visualization*, pages 27–34. ACM SIGGRAPH, October 1994. ISBN 0-89791-741-3.
20. M. Zwicker, H. Pfister, J. van Baar, and M. Gross. Ewa volume splatting. In *IEEE Visualization 2001*, pages 29–36, October 2001.

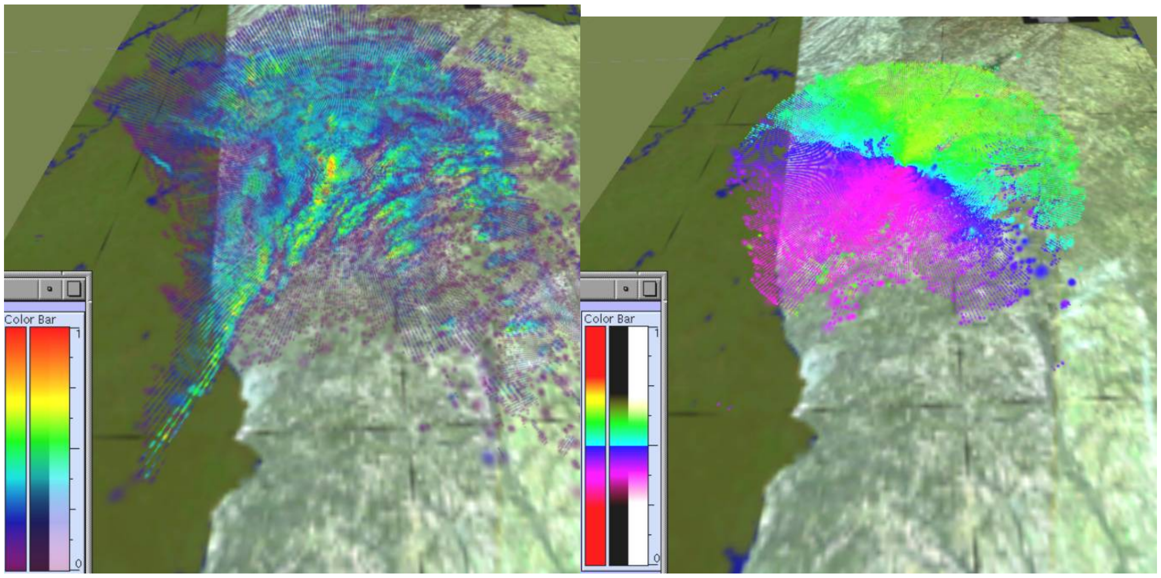

**Figure 8:** *(left) Image of Doppler radar reflectivities over North Georgia. Heavy rainfall is evident along a front from the southwest towards the center (yellow and red splats). Doppler velocities are shown in the right image for the same time step. We have drawn a more limited radius of radar. Because the winds are heading northeast, the northeast half show positive velocities (away from the radar) and the southwest half show negative velocities (toward the radar).*

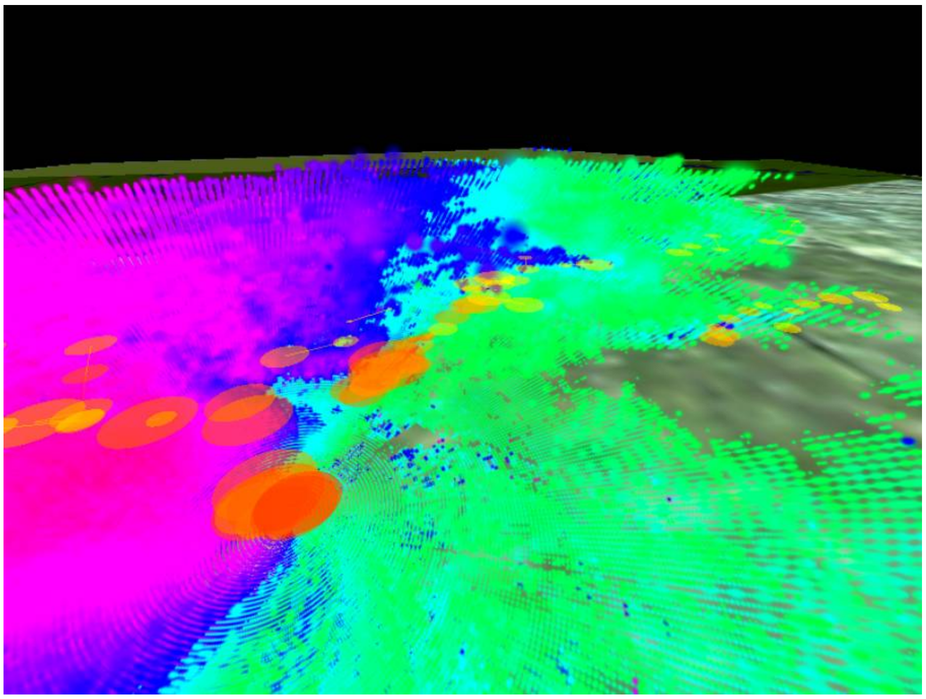

**Figure 9:** *Close-up view of velocity of Doppler radar. Mesocyclones (red and orange circles) have been detected in the velocity profile. This view of the velocity volume shows how they are distributed near an outcrop of negative velocities (dark blue top center) in among the positive velocity zone (green/cyan). The empty area to the right has been made transparent to expose the mesocyclone glyphs. Mesocyclones to the left arise from velocity shears (blue-purple) that are at low-elevation scans and hard to see from this view.*