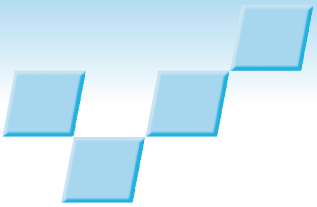


Seeing the Difference between Cosmological Simulations

Steve Haroz ■ *University of California at Davis*

Katrin Heitmann ■ *Los Alamos National Laboratory*

As cosmology simulations help us understand the universe, we must understand how the results of different simulations vary. Visualization and modern graphics hardware can now provide the ability to visually explore these differences interactively.

Visualizing the time-variant results of a simulation can help scientists see patterns that would be difficult to find using only statistics. Furthermore, visualizing the differences between multiple simulations can let scientists directly analyze the simulations' consistency. In this article, we describe our interactive application for viewing the differences between multiple time-variant cosmological simulations. Using programmable shaders and multiple visualization techniques, the application illustrates how the properties of millions of particles vary across simulations. In doing so, we aim to help scientists find patterns in variance across dimensions, simulation parameters, and time.

The driving force behind this study is a series of cosmological particle data sets with multiple dimensions of data as well as temporal changes and variability across all of the dimensions. We specifically examine past scenarios wherein a particle visualization requires a defined spatial element and multiple additional dimensions.[1,2] Expanding on our previous work,[3] we also show examples of the multiple structural components of the data sets, which can be difficult to qualitatively investigate without visualization.

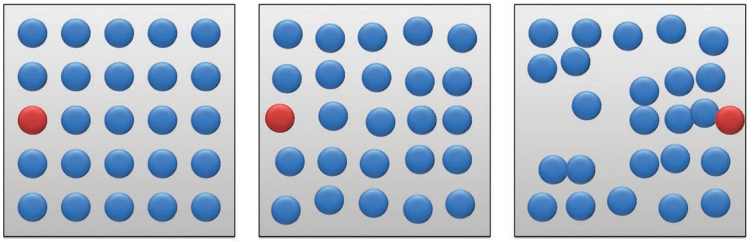
Cosmological simulation comparison

In simulating the universe's evolution, cosmologists employ various different simulation algorithms. Understanding the effect of algorithmic differences on the outcome is critical in ensuring that judgments based on the results are accurate. We examine the results of a cosmological particle simulation generated for a study of cosmological simulation robustness.[4] Our goal is to visualize inconsistencies between different simulations that begin with the same initial conditions.[1]

The simulation begins as 256^3 particles arranged evenly on a cubical grid. The particles are then moved by small amounts to establish the correct cosmological initial conditions (see Figure 1, next page), which are constrained by observations of the cosmic microwave background and the distribution of galaxies on large scales. The particles then move under the influence of gravity in an expanding universe for a number of time steps until the current epoch is reached. As is typical for cosmological simulations, periodic boundary conditions are imposed. When a particle moves past one edge of an axis, it appears on the other end. This property is intriguing from a visualization perspective because the axes wrap. As we'll discuss later, we address this important hurdle in our visualization implementation.

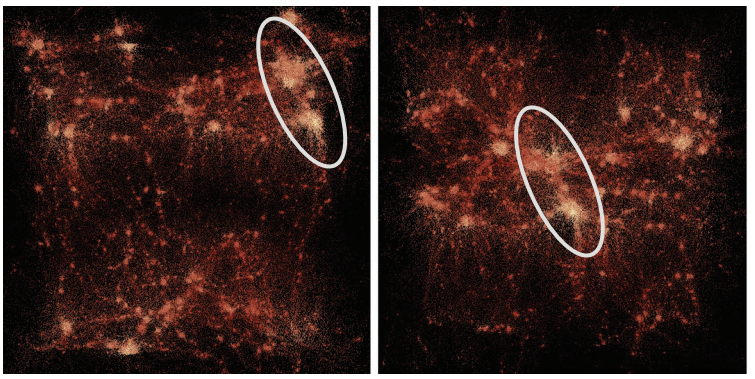
The numerous simulator algorithms used to compute interparticle forces and the approximated values that drive them present results with quantifiable differences.[4,5] Some simulators use hierarchical sampling of the system phase space distribution function while others simplify by distance.[5] Each simulation was run separately, and as a result of their implementation differences, every simulation produced slight deviations in the final particle distributions and velocities.

Past visualization approaches have made significant progress in displaying similar types of particle data. These methods have merit in their effective portrayal of the particles' position and velocity.[1] However, we now have the benefit of hindsight and newer technology. Earlier approaches to vi-



(a) (b) (c)

Figure 1. Simulation progression. (a) The particles are arranged evenly on a 3D grid. (b) They are then perturbed slightly to satisfy the cosmological initial conditions. (c) The simulation then moves the particles based on gravitational forces forming dense clusters and sparse regions. Notice that the red particle moves so far left that it wraps around the edge. In reality, the particles are much smaller and the simulation has no collisions.



(a) (b)

Figure 2. These images show the Flash data set. Each image shows a different wrapping offset of the same data from the same view, and the circled clusters are the same in each image. The clusters and sparse regions of the particles are clearly visible. The coloring is based on velocity magnitude, which correlates highly with dense clusters of particles. Red represents low velocity while light yellow represents high velocity.

sualize even a single data set tended to rely on hierarchical groupings of the data, which required preprocessing and only provided complete detail at the lowest levels.[6]

Extending the single-data set efforts, James Ahrens and his colleagues studied the visualization of multiple data sets via a side-by-side comparative analysis.[1] They visualized two individual data sets in separate views with linked transforms. A different simulation algorithm generated each data set, but the initial conditions were the same. Although this method can be useful if followed by a focused serial search through the images, it unfortunately relies on the assumption that we can proficiently detect differences in side-by-side images. Testing a principle known as *change blindness*, perception researchers have found that we're surprisingly poor at detecting changes between images if a small interruption exists between them.[7] That interruption can be as insignificant as the brief, split-second blindness that occurs during a saccadic eye movement. In other words, significant changes can go unnoticed when shifting focus from one image to another. The optimal presentation of difference would lack even the slightest spatial and temporal interruptions.

Data set and visualization interface

Each particle in the data set has a position and velocity in 3D space. The particles also have an ID number, so a single particle can be identified in multiple simulation results.

The particles move under the force of gravity and form large-scale structures. The overall structure has three main elements: highly dense regions (clusters), highly sparse regions (voids), and filaments. The general features of the large-scale structures (such as the clusters' positions) and the shapes of voids are the same in all the different simulations, although the exact particle positions and densities vary. Within the dense clusters, particles with varied velocity can be in extremely close proximity, so voxelizing and rendering this data as a volume isn't desirable.

For one view, our implementation renders each particle as a small line with a length and orientation that correspond to the particle's velocity. The middle of that line is the particle's location. The location property, however, cannot be taken at face value. The wrapping of the axes means that important information, such as clusters, might be split by the edges of the bounding cube (see the "Wrapping Lines" sidebar). To address this potential problem, we created a set of three sliders. Each corresponds to one of the axes and represents an offset used to wrap the axis. The user can adjust the sliders to move the interesting information away from the edges (see Figure 2).

Multivariate representation

Next to the primary spatial visualization, we include a parallel-coordinates view of the data that also acts as a control mechanism. This view lets numerous dimensions of the data be simultaneously visualized in reference to each other.[8] The axes include the positional dimensions, velocity components, velocity magnitude, positional uncertainty, and velocity uncertainty (see the bottom of Figure 3, page 40). Users can select any of the dimensions as the coloring basis for both views.

Adding the uncertainty dimension

One way to make an object appear uncertain is to blur the representation. In this case, the sheer

quantity of particles makes their average screen representation smaller than a pixel on even the largest displays. Blurring would be unnoticeable due to the existing aliasing and lack of shape, or it would create too much occlusion. We therefore use color to represent uncertainty, which Figure 3 presents.

Binding color to uncertainty would make that powerful visualization feature unavailable to other dimensions. Past researchers have effectively associated color with velocity magnitude in this data set and found insightful information, such as a strong correlation with density.[1,5] To unconstrain color, we let users interactively select a parallel coordinates dimension. This ability lets them dynamically choose which feature is important so they can focus on the information they want.

When a new dimension is selected for coloring, both the spatial view and the parallel coordinates view reflect that change. The particles in the spatial view are colored based on their associated value, and the parallel coordinates view is also redrawn under the new coloring scheme. The colored parallel coordinates let users find correlations and patterns between dimensions that might be difficult to see by only using color in the spatial view. Using parallel coordinates as a selection tool and interface for a spatial view has been shown to be particularly effective for particle visualizations.[2,8] To make the parallel coordinates accurately render over 16 million points, we used a 32-bit-per-channel frame buffer that allows for very low alpha values. In turn, the lines blend to form a smooth antialiased display.

To add different types of variability as dimensions, we need to quantify them. The user should be able to visualize, explore, and interact with uncertainty in both position and velocity in the context of all the data variables. For each particle, the application averages the vectors of each value and finds the standard deviation. It then calculates the magnitude of the resulting standard deviation vector. If only two data sets are loaded, we approximate the standard deviation as the difference. Although calculating the velocity uncertainty is straightforward, calculating the positional uncertainty requires taking the axial wrapping into account. A particle whose positions slightly straddle the edge of an axis is appropriately calculated as having little uncertainty.

Findings

As an initial test of the visualization system, we confirmed previous findings[1] that the particles with high velocities clustered in the dense regions, apparent in Figure 3.

When the velocity uncertainty, the position uncertainty, or the velocity magnitude is selected as the colored axis, a smooth vertical gradient appears between those axes (see the bottom right of Figure 3). This pattern shows that the values are highly correlated.[8] We expect this result, as a high velocity would create a large positional change resulting from only a slight variation in velocity. We also tried normalizing the velocity uncertainty by calculating it based on velocity orientation alone (for example, without incorporating velocity magnitude). The difference was minimal, and the results showed that density, velocity magnitude, and positional uncertainty also correlate strongly with uncertainty in velocity orientation.

Wrapping Lines

The wrapping of the axes is an interesting property of the data set. For the simple point-based rendering, giving users the ability to add an offset to each axis lets them center any interesting features. The implementation is easily accomplished using just a modulo operation in the vertex shader. However, rendering lines in a wrapped space presents some design and implementation hurdles.

Figure A illustrates this problem. If a point in data set A (P_A) is spatially centered while that same point in data set B (P_B) is slightly to the right, the difference in location is very small. If both points were shifted equally to the right so that P_B wrapped around but P_A did not, the distance between them shouldn't change. Their locations on the screen might be far apart, but because they were shifted equally, nothing should change about the actual distance between them. Although calculating this distance is trivial, providing an intuitive representation of that difference isn't as straightforward.

Naively drawing a line from P_A to P_B would create a line that would sprawl across the entire space. To avoid such a counterintuitive representation, we choose to move one point outside of the wrapping boundary whenever the line is larger than half of the wrapping space. Having P_B to the right of P_A lets us draw a correctly sized line between them, resulting in a consistent representation of distance by line size. With more than 16 million points, this special case happens to hundreds or even thousands of particles in most time steps. Although this solution pushes some particles outside the boundary, it's far better than having numerous long lines occlude the display.

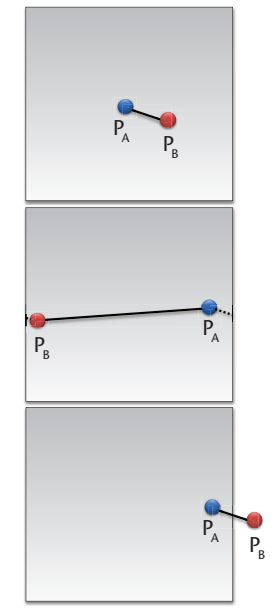


Figure A. The blue point is P_A and the red point is P_B. Normally, a line could easily be drawn between them (top), but if a wrapping problem occurs (middle), one point must be moved outside of the wrapping boundary to maintain a correct distance (bottom).

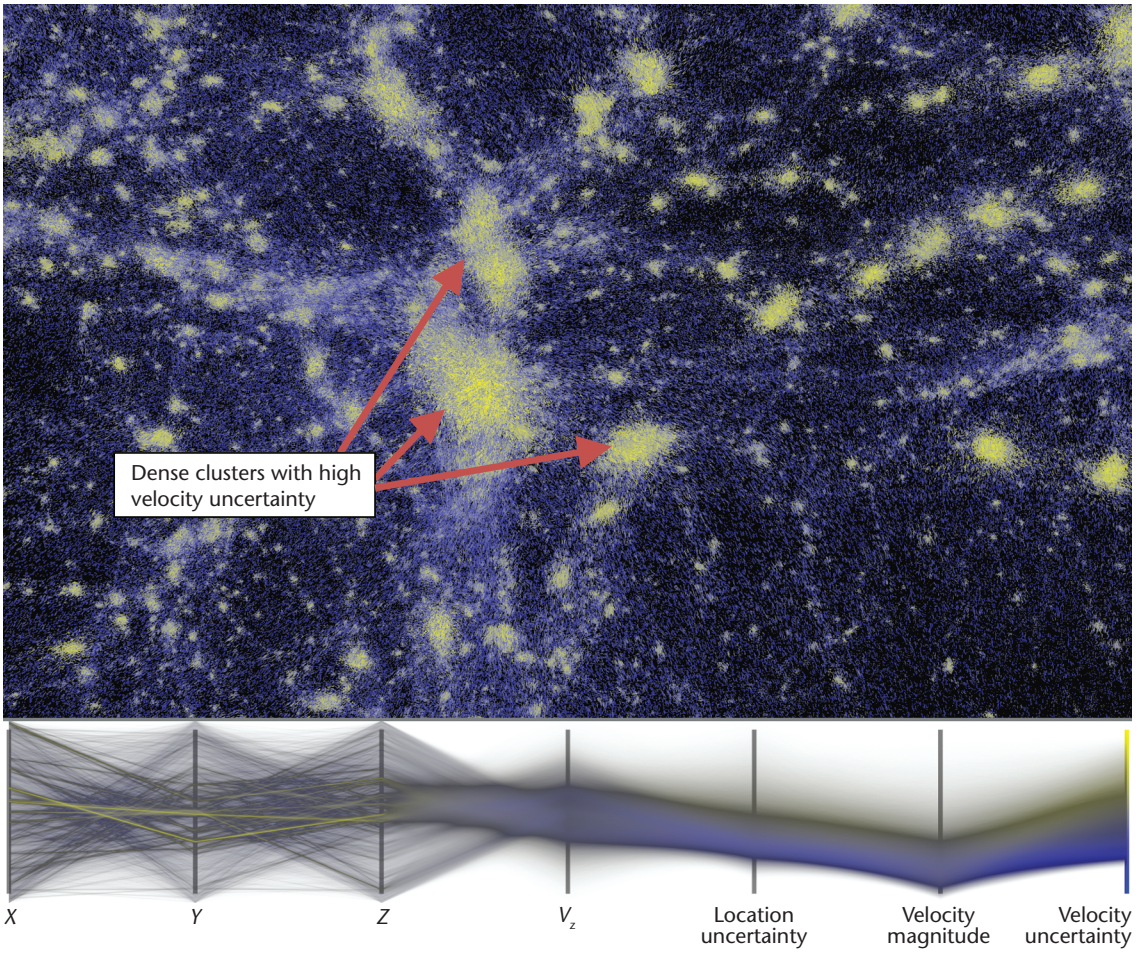


Figure 3. This zoomed view shows a comparison of the Flash and Hashed-Oct Tree simulations with the particles colored by velocity uncertainty. Yellow particles have the highest velocity uncertainty while blue particles are more certain. The parallel coordinates view on the bottom shows a smooth gradient along the rightmost axes, velocity, and location uncertainty. The bright yellow lines along the center of *x*, *y*, and *z* axes (on the left) show the positions of the dense and highly uncertain clusters in the center of the view. The simulators' different force resolutions result in high variation (bright yellow) in the dense regions.

A correlation exists between positions along any spatial axis and the velocity along that axis because the particles move toward a group of highly dense clusters. As seen in Figure 4 (next page), coloring the particles based on the velocity's angular distance from a particular vector, such as the *z*-axis, results in a somewhat sharp color border on either side of the cluster. Such insight would be difficult to find if color were locked to vector magnitude.

Variation over time

Another aspect of the simulations that scientists wish to explore is the impact of differently initialized particles. Scientists want to follow the universe's evolution from the very first moments to today, yet the simulators' accuracy places limitations on the starting time's simulation. In the simulation's early time steps, the small perturbations can be too miniscule to be accurately stored by the single precision floats that the simulations use. The scientists therefore need to start the simulation a few million years after the Big Bang (the universe is roughly 13.66 billion years old). Fortunately, the physics during that initial period is well understood. They can use the Zel'Dovich approximation to skip over the initial time steps with the added bonus of reducing simulation time.[9] The caveat of the approximation is that using it at a time too close to the current epoch can adversely impact the simulation's accuracy, as structures don't seem to form in the same way. Furthermore, the accuracy of the Zel'Dovich approximation requires that particle paths don't cross. The questions that arise are what the optimal starting time is and how the later start impacts the results at later times in the simulation.

The universe's physical expansion is scaled out of the simulation by the expansion's scale factor, a. The simulation box therefore always contains a constant "comoving" volume even though the actual physical volume always increases. Each time step represents a particular epoch in the universe's history, defined by the particular value of the scale factor or alternatively the redshift, z. The scale factor and redshift are related via the equation, $a = 1/(1 + z)$. This data set has multiple series of time-variant data. Beginning with the initial grid, each time series extends the Zel'Dovich approximation to a different time step. Each series then runs through the simulation and outputs several time steps along the way (see Figure 5). The result is a data set with the almost unstudied property of time-variant particle uncertainty. Each time step has position and velocity uncertainties that vary according to approximation length.

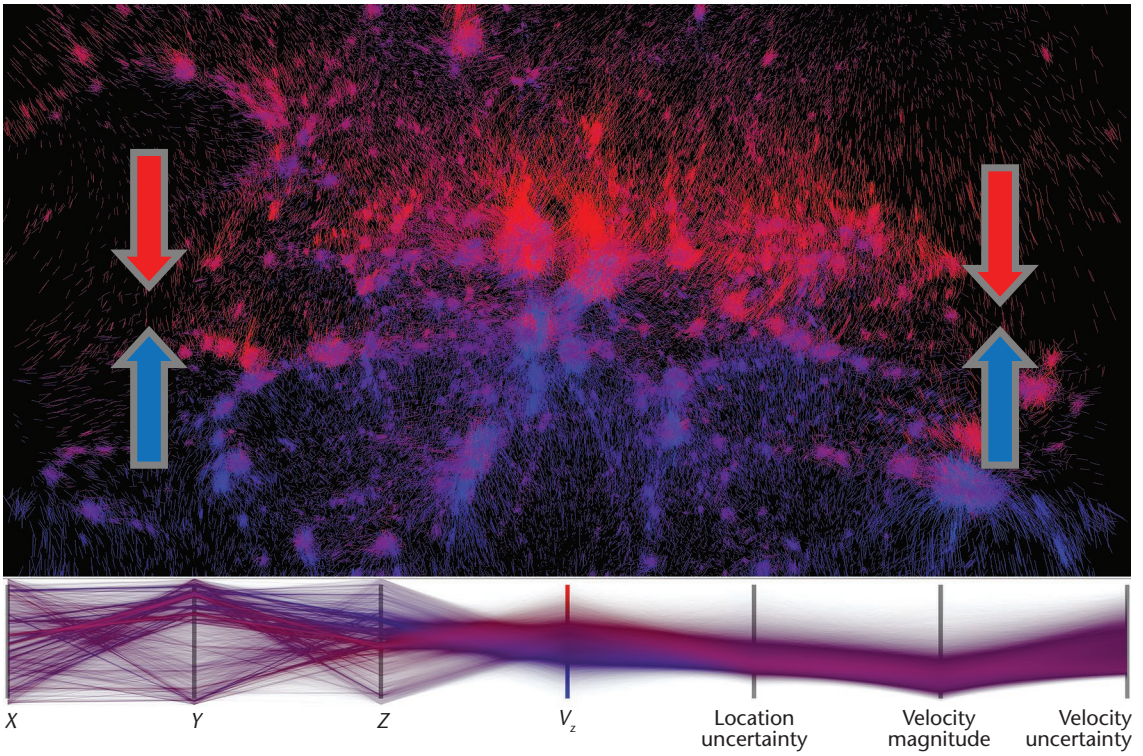


Figure 4. The particles are colored based on the projection of their velocities along the z-axis. The blue particles move up and the red particles move down. The image shows that the particles gravitate toward the dense regions in the center. The positional axes (left three) show a clear correlation due to the bands of different color, whereas the uncertainty and velocity magnitude axes (right three) have a consistent intermediate color.

Time-variant findings

Continuing to use the aforementioned spatial and parallel coordinates views, scientists can see how the data's correlations and differences vary over time. Figure 6 displays the particles at a very early time (upper panel) and in their final state (lower panel). We compare the particles' evolution from the earliest start (z = 250) and the later start (z = 150) at time step z = 50 and in the final state. Initially, the particles are distributed smoothly following a Gaussian random field. At these early times, the particle movements are still small and they haven't moved on average more than the mean interparticle spacing. Therefore, we don't expect to see big differences at these early stages. During the time evolution, gravity is acting on the particles, and they end up in the large-scale structures we mentioned before: clusters, filaments, and voids.

Figure 6 (next page) demonstrates how the uncertainties evolve with time if the simulation is started at different epochs. The velocity and location uncertainty, which represented variation among simulator algorithm results, now represents variation caused by different initialization times. As we expected, at the early time snapshot, the uncertainties are rather small and basically the same throughout the simulation volume (Figure 6a). At the final output, the uncertainties are different in the clusters, filaments, and voids (Figure 6b). We'll investigate this finding more specifically later in this section.

Scientists must examine many different particle properties in a convenient way and possibly at the same time. Our tool lets users choose to correlate the particles' color based on the property of interest. Coloring by velocity uncertainty and animating across time, users can easily spot any periods of rapid variation growth or even possible reduction (which scientists hope to find). We also give users the option of rendering a line between each of a particle's positions, so color can be used for another property such as velocity magnitude (see Figure 7, page 43).

Let's return to the question of the amount of uncertainty in the simulations in different large-scale structure regions. The highest densities in the simulation are the cluster regions. One important statistic of interest in cosmology is the *mass function*. The mass function measures the number of bound structures, or halos (clusters are the largest halos), in a certain mass bin. From this statistic, cosmologists can infer information about the universe's formation as well as properties of the dark matter that forms the large-scale structures. Zarija Lukić

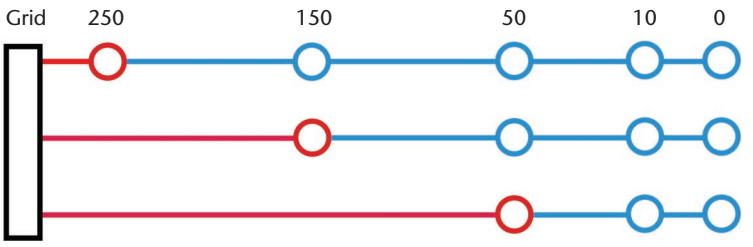


Figure 5. This diagram illustrates the correlation of data sets. The grid (black box on the left) is the unperturbed locations for all of the particles. The Zel'dovich approximation (red) progresses the particles to a particular time step. The simulation (blue) then progresses until the present (time 0). Each circle represents data set.

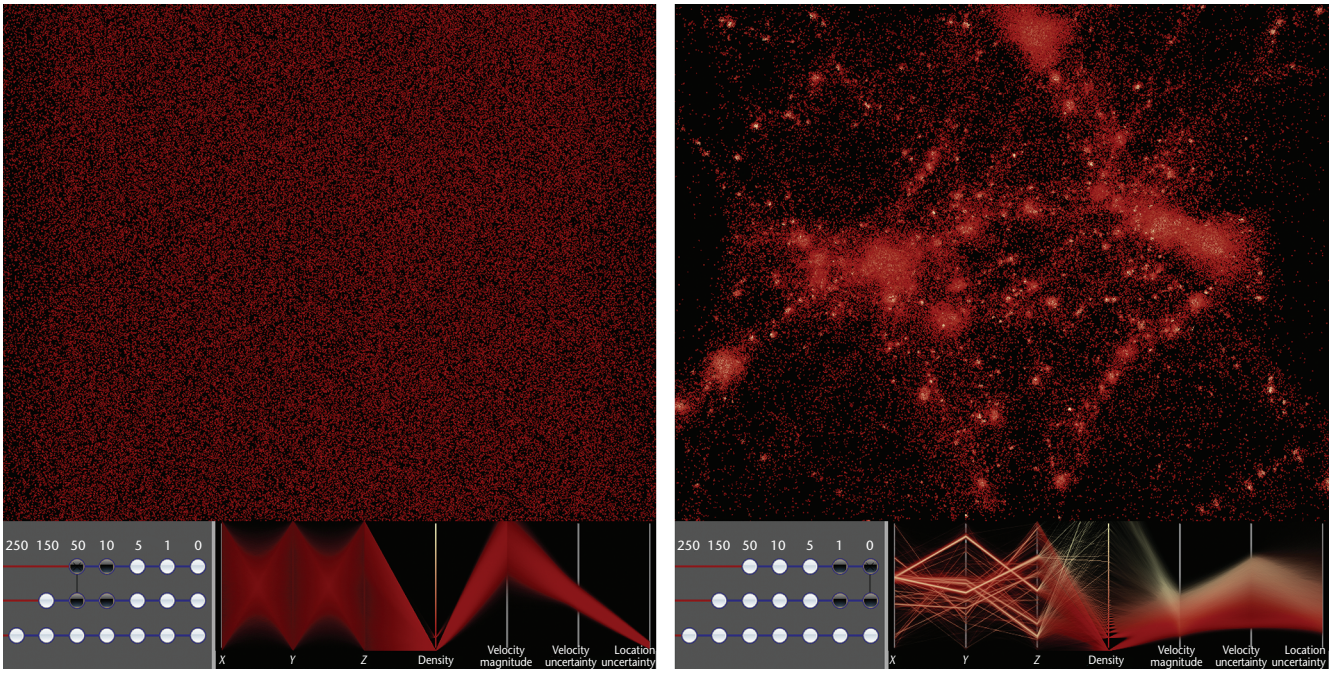


Figure 6. Comparison of two time steps. (a) Time step 50 colored by density. The particles have high velocities and are still fairly evenly distributed. The parallel coordinates view provides a lot of insight about the data. The pattern between the *x*, *y*, and *z* axes on the left, the even brightness, and the density axis in the middle show that the data are evenly distributed and have no clusters. The velocity uncertainty and location uncertainty (rightmost axes) have log scales and show the limited variation at this time step caused by the approximation. (b) Time step 0. The vast majority of the particles have coalesced into clusters. The bright lines between the spatial axes on the left represent those dense clusters, and the central density axis is noticeably higher and more varied. Both uncertainty axes on the right show significant growth compared with (a).

and colleagues found that the number of halos and therefore the mass function differs if the simulation starts at different epochs.[10] A later start results in a smaller number of halos over the full mass range.

So why are fewer halos formed when the simulation was started late? Figure 8 (page 44) provides an important clue to this question. The upper left panel shows the earlier start and points out two halos (a large one and a smaller one). These two structures are much fuzzier in the right panel (late start). Because the halo-detecting algorithm is based on finding nearest neighbors, particles in the fuzzy outer part of a halo won't be identified as belonging to that halo. Therefore, the mass of the halos (the sum of all particles belonging to the group) in the right panel will be lower and small structures will be missed. The mass function will therefore be lower overall if the simulation is started too late.

We next investigate the uncertainties in the filamentary structures. The filaments are flow regions in between large halos. The streaming velocities in filaments are relatively high (not as high as the particle velocities in clusters) and therefore one might expect relatively high uncertainties in these regions. Figure 8 shows an example of this: The central bright lines show the uncertainty in the filament. It's interesting to note that the uncertainty follows the filament's orientation and flow. The implication is that the position of a particle along a filament might vary, but whether it becomes part of a filament doesn't.

The next question to answer is how early the simulation must begin to capture all halos. To answer this requires convergence studies. If the simulation starts early enough, the results should be the same when compared with the results of an even earlier start. Figure 7 shows an example of how visualization can aid convergence studies. The simulation began at three different times, and the panels show variations of the particles at different time steps from two different starting points. At very early times, the difference between the two starts is small, indicating that the start at 150 might have been sufficiently early (Figure 7a). As the simulation progresses, the differences become larger. Starting from the second panel, nonlinearities amplify the differences in the structures, and the image accordingly shows much more variation (Figures 7b and 7c). The large-scale structures start to form at this point, and the uncertainty lines appear as a shell around where the cluster will eventually form (Figure 7d). In ongoing work, we'll investigate higher-order approximation schemes for the initial conditions; the current tool

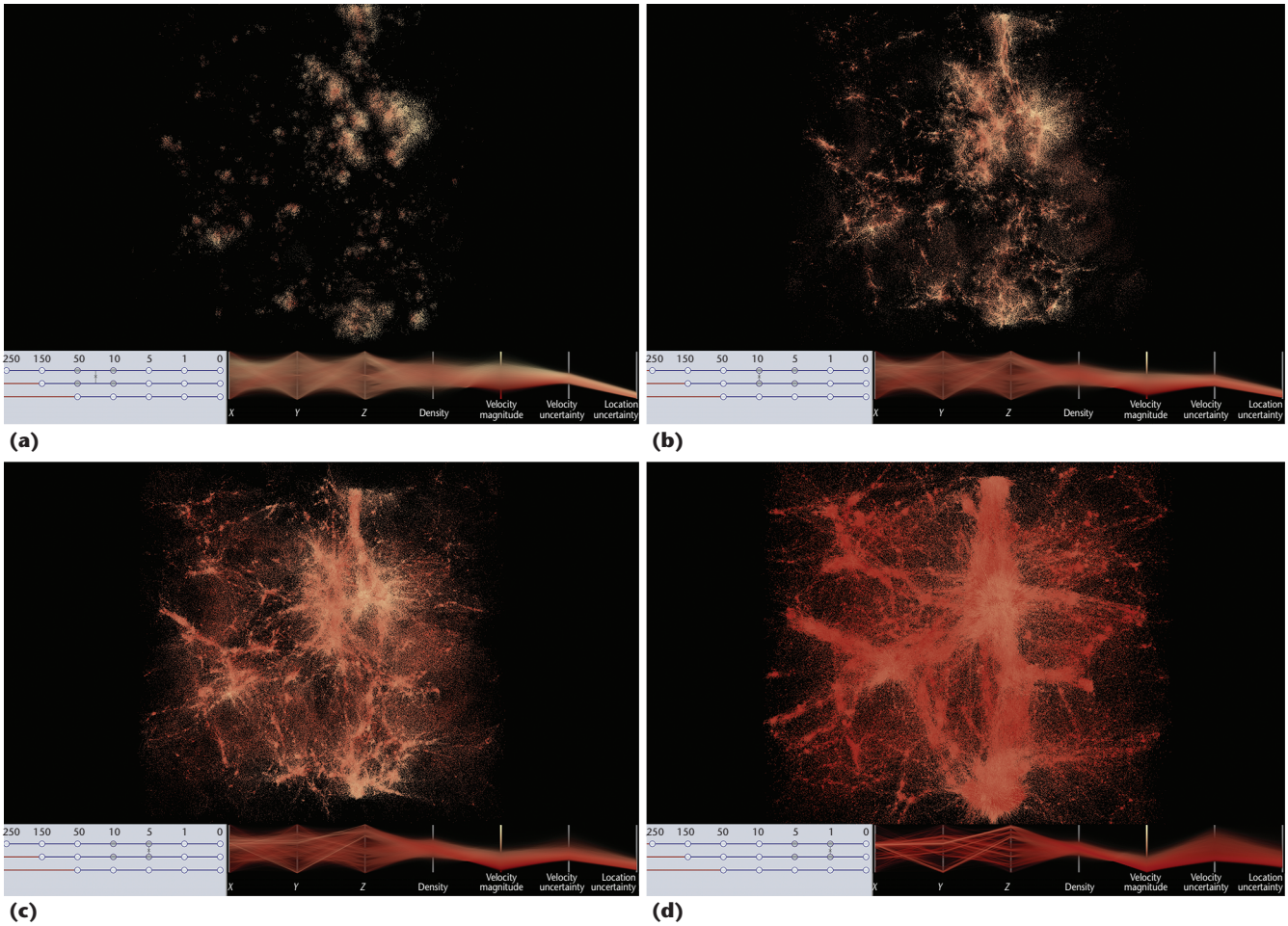


Figure 7. These images show a progression from redshift 50 to redshift 1. (a) For each particle, a line colored by velocity extends from the position in one series to the position in another. (b–d) As time advances, the variation between the series grows and becomes more structured. The time and series selection interface can be seen in the bottom left of each image with the darkened circles representing the currently active time steps.

will be helpful in exploring the differences between different schemes.

The accuracy of simulation data is crucial to cosmologists. Ongoing and upcoming surveys will map out the structures in the universe to very high precision. To interpret these observations, simulations must be as accurate as the observations. Many aspects of research are based on these simulation results, and understanding the complex spatial and temporal patterns of the accuracy levels is difficult to accomplish using only quantitative analysis. By interactively visualizing the data, scientists can have a better understanding of where and to what extent simulation variation and approximation might impact results. Scientists commonly measure simple statistics from the simulations, such as mass functions or two-point correlation functions. Differences in these statistics due to different simulation algorithms, starting redshifts, particle loadings, force resolution, and so on have to be understood at the percentage-level accuracy. Visualizing the differences in the simulation outputs under different parameter settings can be helpful in gaining a better understanding of their causes. At the accuracy level required, the obtained insight can be pivotal for progress in understanding the error properties of complex algorithms.

A parallel coordinate control interface is helpful for visualizing correlations in particle data. Although it has been established as a powerful technique for many visualization systems, we use a parallel coordinate view as an interface tool to extend the limitations of a spatial view. This interface is helping scientists explore and gain insight into uncertain data too complex for a spatial view to display alone. As scientists continue to generate ever larger sets of uncertain, multidimensional, spatial data, they need a means of selecting the type of information to examine. By using modern programmable graphics hardware, we can give scientists a level of detail and interactivity and a range

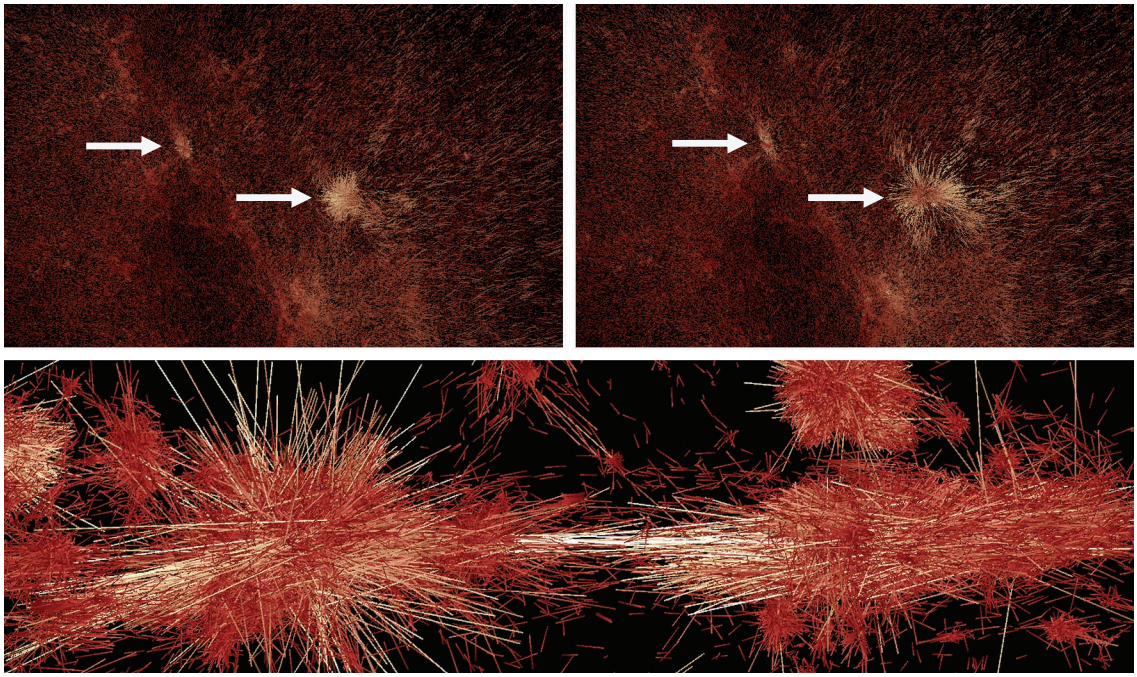


Figure 8. The top images are zoomed into a region of particles at redshift 10. Each particle is rendered as a small line oriented and sized by the velocity vector; the color represents velocity uncertainty. The left image shows the particle positions and velocities if the Zel'dovich approximation were taken to redshift 250. The particles in the right image were approximated to redshift 50. The structural formation and coalescence of the particles is much more significant with the reduced approximation (left) than with the longer approximation (right), and varied density of the bright clusters best demonstrates this observation. The bottom image shows the uncertainty along one of the filaments. By drawing a line between each particle's positions in different data sets, the direction of the uncertainty becomes apparent. Here, the uncertainty can be seen radiating outward from clusters but moving along the filament between them. The lines are colored by the magnitude of the difference, which makes the filament stand out due to its bright color.

of options impossible only two years ago (see the "Vertex Shader Implementation" sidebar). The new performance benefits also allow animation, once only relegated to prerendered clips, to be an integral, insight-enhancing, and perceptually beneficial feature. This work provides an example application that can view complex, multidimensional, and time-variant data as well as critical uncertainty information. Furthermore, it's an effective example of a juxtaposition by providing multiple views of evolving features.[11] Presenting the uncertainties as axes that are no different from standard variables, we improve the simplicity and flexibility of the users' tasks. A benefit of this system is that interacting with uncertainty in the same manner as other variables can provide insight into correlations of the uncertainty without eliminating the ability to visualize those other variables.

Acknowledgment

This research was supported in part by the US National Science Foundation through grants CCF-0634913, IIS-0552334, CNS-0551727, and OCI-0325934, and the US Department of Energy through the SciDAC program with Agreement No. DE-FC02-06ER25777. Thanks to the Cosmic Data ArXiv for making the data sets publicly available.

References

1. J. Ahrens et al., "Quantitative and Comparative Visualization Applied to Cosmological Simulations," *J. Physics: Conf. Series*, vol. 46, no. 1, 2006, pp. 526–534.
2. C. Jones et al., "An Integrated Exploration Approach to Visualizing Multivariate Particle Data," *Computing in Science and Engineering*, vol. 10, no. 4, 2008, pp. 20–29.
3. S. Haroz, K-L. Ma, and K. Heitmann, "Multiple Uncertainties in Time-Variant Cosmological Particle Data," *Proc. IEEE Pacific Visualization Symp.*, 2008, IEEE Press, pp. 207–214.
4. K. Heitmann et al., "Robustness of Cosmological Simulations I: Large Scale Structure," *The Astrophysical J.*, vol. 160, 2005, p. 28.
5. K. Heitmann et al., "The Cosmic Code Comparison Project," 2007; www.citebase.org/abstract?id=oai:arXiv.org:0706.1270.
6. M. Hopf, M. Luttenberger, and T. Ertl, "Hierarchical Splatting of Scattered 4D Data," *IEEE Computer Graphics and Applications*, vol. 24, no. 4, 2004, pp. 64–72.

Vertex Shader Implementation

Scientists need to interactively examine how the data change over time and view the variation caused by the approximation length. We wanted to give them a tool that could recreate the movements and changes in the data and allow them to vary the visualized approximation length in real time. We accomplished this goal by parallelizing the bulk of the application: reading, interpolating, and processing the particle positions.

To achieve an interactive and real-time display, we needed to design a means of reducing the amount of data sent to the graphics card in each frame. We also needed 16-bit precision floats to reduce the size of the data and control over vertex position to compensate for wrapping. Our solution was to take advantage of the new texture access and branching capabilities of vertex shaders. Figure B shows an overview diagram. Until recently, such complexity was only available to fragment shaders,[1] which would have required the use of buffers and multiple rendering passes.

We begin by converting the locations of all particles in a data set to a texture with 16 bits per color channel. The *x*, *y*, and *z* components of the positions are respectively stored in the red, green, and blue components of the texture. The alpha channel is sometimes also used to provide additional particle data used for coloring. Although a single dimensional array would be the most appropriate storage format, limitations of texture size forced us to use 3D textures instead. After converting all four data sets to textures, we assign them to a shader-accessible 3D sampler.

To access all elements of the texture set, we create a vertex for each particle and set the vertex's *x* value to its corresponding index. These index vertices are then compiled into an OpenGL list. We only need to perform this step during initialization, and all rendered frames need only call this list with the vertex shader active.

When a vertex is sent to the shader, a 3D lookup is first performed to obtain the particle's index within the 3D textures. With this 3D index, the shader is then able to retrieve the particle's information from each of the four data sets. These four variants of the same particle are then smoothly interpolated based on the current time and series selected by the user.

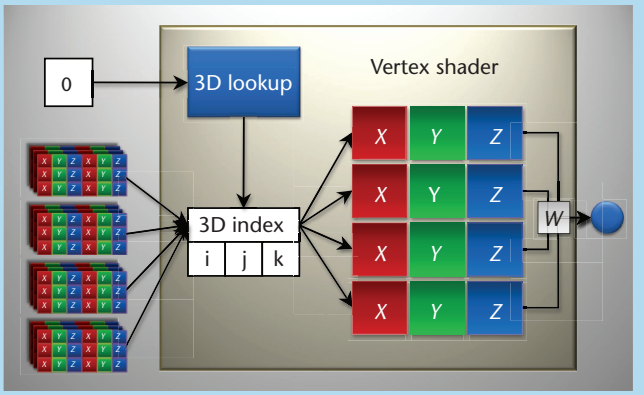


Figure B. An overview of our vertex shader design. An index and data set textures are passed in, and a correctly interpolated and positioned particle is returned.

An important stage in determining the position of the particle is determining the impact of wrapping. Represented by the small boxed *W* in Figure B, this process actually accounts for a significant amount of the shader code, as each interpolated dimension must be considered in determining the optimal position of particle.

Performing the lookup and 2D interpolation would be extremely taxing if it were performed on the central processing unit (CPU). However, because each particle can be processed independently of the rest of the data, the problem is embarrassingly parallel. Offloading this work to the CPU provides us with the real-time display and lets the scientists interactively explore the data by smoothly interpolating across both time and approximation while rendering millions of particles in real time.

References

1. D. Blythe, "The Direct3D 10 System," *ACM Trans. Graphics* (Proc. Siggraph), vol. 25, no. 3, 2006, pp. 724–734.

7. D.J. Simons and R.A. Rensink, "Change Blindness: Past, Present, and Future," *Trends in Cognitive Sciences*, vol. 9, no. 1, 2005, pp. 16–20.
8. A. Inselberg, "The Plane with Parallel Coordinates," *The Visual Computer*, vol. 1, no. 4, 1985, pp. 69–91.
9. Y.B. Zel'Dovich, "Gravitational Instability: An Approximate Theory for Large Density Perturbations," *Astronomy and Astrophysics*, vol. 5, no. 84, 1970, p. 168.
10. Z. Lukic et al., "The Halo Mass Function: High Redshift Evolution and Universality," *The Astrophysical J.*, vol. 671, 2007, p. 1160.
11. N.J. Zabusky et al., "Visiometrics, Juxtaposition and Modeling," *Physics Today (J. American Institute of Physics)*, 1993, vol. 46, no. 3, pp. 24–31.

Steve Haroz *is a PhD student in computer science at the University of California at Davis. His primary research interest is the intersection between visualization and human perception. Haroz received his BS in computer science from Clemson University. Contact him at sharoz@ucdavis.edu.*

Katrin Heitmann *is a staff member at Los Alamos National Laboratory in the Space Science and Application group. Her primary research interests are physical and computational cosmology, specifically in precision simulations of the large-scale structures in the universe. Heitmann received her PhD in physics from the University of Dortmund in Germany. Contact her at heitmann@lanl.gov.*