

1) Introduction

In this paper I will discuss my work towards coming up with a way to visualize density in simulations where it is not readily apparent where the majority of the density is located. To help me work towards this goal, I used the RIT GRAPEcluster as my visualization system, and wrote different visualizers for the system, which worked flawlessly.

The difficulty in viewing density is generally due to the overabundance of particles. This makes it hard to determine, just by looking at the picture and/or movie, which portions are denser than others. In order to come up with a solution to the problem, I worked with Josiah Allen, and borrowed from work from William McLaughlin.

The rest of this paper is broken up into four sections:

- 2) Pre-Visualization Development: In which the work done prior to starting on the development is described.
- 3) Visualization Algorithms: In which the work done to develop the algorithms, and a description of the algorithms used is described.
- 4) Results/Conclusions: In which the effectiveness of the algorithms used to help visualize density is described.
- 5) Further Development: In which a course for further work in this area is looked at.

Sections 3 and 4 are further broken up into sub categories, once for each of the visualization techniques I developed while working on this project. These techniques are:

- 1) Density as Contoured Sections
- 2) Density as Contour Follow
- 3) Density as Slices

For each of my visualizations, it was decided to take the data, and project it down as planer (2 dimensional) data on each of the three planes of a cube (xy plane, yz plane, and xz plane) instead of inside the cube where the stars actually inhabited. This process is described further in Section 2.

2) Pre-Visualization Development

In order to develop improved ways to view density in the GRAPEcluster environment, Josiah and I needed a way to calculate the density at different points in the cube. To do this, we came up with the following algorithm:

- 1) Ask the user to for a number of slices.
- 2) Setup a 3x3 “bucket” matrix of the inputted number of slices.

- 3) Iterate through all of the stars
 - a. Add the star to the 3x3 matrix in the “bucket” corresponding to its (x,y,z) position

After the “bucket” matrix was setup, Josiah then wrote a function to kernelize the data. This is a process by which data (such as this density data) is “smoothed” by taking a weighted average of adjacent cubes, and then adjusting the value in each cell to reflect the fact that neighboring cells have an effect on the density of any particular cell.

The next thing I had to do was setup a function to convert the 3x3 matrix into three 2x2 matrices, one each for each of the three planes. This was necessary because my visualizations were all going to be projected onto the three planes of the cube, namely the: xy plane, yz plane, and xz plane. In order to do this, I devised the following algorithm:

- 1) For each of the (x, y, z) coordinates, choose one coordinate: c, to keep static.
 - a. Create a 2x2 matrix with the same number of slices as the 3x3 matrix.
 - b. Set all values of the 2x2 matrix to 0.
 - c. Loop over the other two coordinates: a, b.
 - i. Loop over c.
 1. In the 2x2 matrix in position a, b add the value of the 3x3 matrix in position a, b, c.
- 2) Kernelize the new 2x2 matrices.

With this kernelized data, I was able to begin work on the new visualizations based on the projected data.

3) Visualization Algorithms

In this section, all the algorithms describing the new visualization techniques will be described.

3.1) Density as Contoured Sections

The first algorithm developed was Density as Contour Sections. Contour drawing (or level line drawing) is based on the idea that lines can be drawn to connect points of common value (in this case density) to form curves representing specific values. In this algorithm, the entire square area of density is broken up into smaller square cells as described above. Looping over these cells, the maximum and minimum densities are found, and then, based on the number of curves requested, the values of the densities at which each of the curves are to be drawn are calculated.

Looping over all of the cells again, each one is cut in half along the upper-left to lower-right diagonal. In each of the triangles, the predetermined curve values are looped over, and the algorithm attempts to find two points that are on the curve value by interpolating between each of the three points on the triangle. If two points are found, a line is drawn between the two points using the color value calculated based on the density value. If three points are found, however, no line can be drawn because that is considered a degenerate case. The next triangle is then selected by the loop, and the process repeated.

Finally the resulting pictures are rotated, and moved to be put onto a cube.

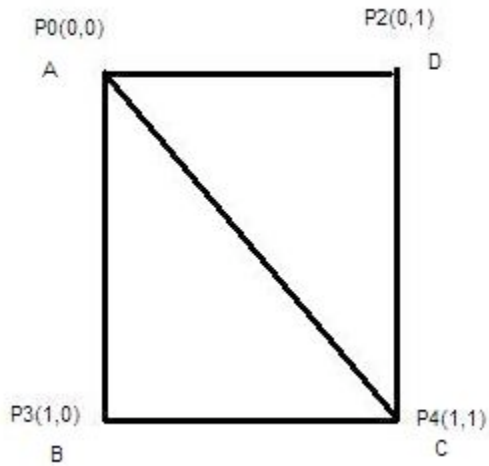


Figure 1: Figure 1: This is an example of a single cell after it has been cut from the larger whole. A level line will be drawn if the value to be plotted is between the three corners of the current triangle. The two triangles have vertices: A, B, C for the “lower” triangle, and A, C, D for the “upper” triangle.

3.2) Density as Contour Follow

This technique is similar to the one applied above. The same technique for determining the minimum and maximum densities, as well as the values of the level curves to be drawn is applied. Once the minimum and maximum values, along with the contour values are calculated, the algorithms start to differ.

In this algorithm, the cells are also looped over. If, however, one of the contour values is between the three vertices, then the two points are found the same way as above. The next step, however is to determine a starting edge and a finishing edge are determined for the curve. If the contour goes through a vertex on the corner of the triangle, then that vertex is chosen as one of the edges as opposed to arbitrarily choosing one of the edges that it lies on. The starting edge (or point) is determined either by going from South – West – North – East, or if there is an edge on the border, that is chosen as the start edge. The triangle is then added to a collection of “seen” triangles. The next triangle, as opposed to being determined by the loop, is determined by the end edge of the previous triangle. Thus, if the triangle ends on the East edge, the west edge of the triangle to the right is chosen. Similarly, if the North edge is the end edge the South edge of the triangle directly above it is chosen, if it is the South edge, the North edge of the triangle directly below it is chosen, and finally if it is the West edge the East edge of the triangle to the left of it is chosen. If, however, one of the points is a vertex of one of the triangles, then there is no easy way to determine where the next triangle is. Instead of knowing where the curve must go, one must loop through the triangles around that vertex, and determining where the contour “goes” by seeing which of those triangles contains the curve value, and the same point that ended the previous triangle is chosen to start the new triangle, with the modification that instead of the reference point being the previous triangle, it is the new triangle. The drawing of the curve finally stops when either: a) a

triangle which has already been encountered is seen again, or b) if the contour attempts to go off the edge of the mapping area. Once the end of a closed curve is found by either of the two situations, the algorithm continues at the triangle that would have come after the first triangle if the loop wasn't preempted by the contour follow function. Finally the resulting pictures are rotated, and moved to be put onto a cube.

3.3) Density as Slices

This technique is vastly different than the other two techniques in terms of what is seen once the drawing is completed. As opposed to drawing level curves, a mountain range like map is drawn with the heights each point representing the density at that point.

Once the data is broken up into the cells, and the minimum and maximum densities are calculated as before, a modification must occur. To ensure that the "height" of any "mountain peak" does not exceed a predetermined value, and thereby making the picture harder to discern, the densities must be normalized so that the max height is of a predetermined value. Once this is done, the two dimensional array is looped over, and a height value is added to a vector corresponding to each of the points. This is then made into a plane. This plane, with its peaks and valleys are then moved onto the cube on the corresponding face, and then displayed.

4) Results and Conclusions

4.1) Density as Contoured Sections

As expected, this method did not perform well at all. The algorithm skipped some cells, and the contours did not follow correctly from cell to cell. Even with a large number of slices, the drawings produced were not crisp, and cells were still skipped, but the algorithm did give some idea as to the underlying structure of the density. The following pictures are an example of what the algorithm produced on sample input.

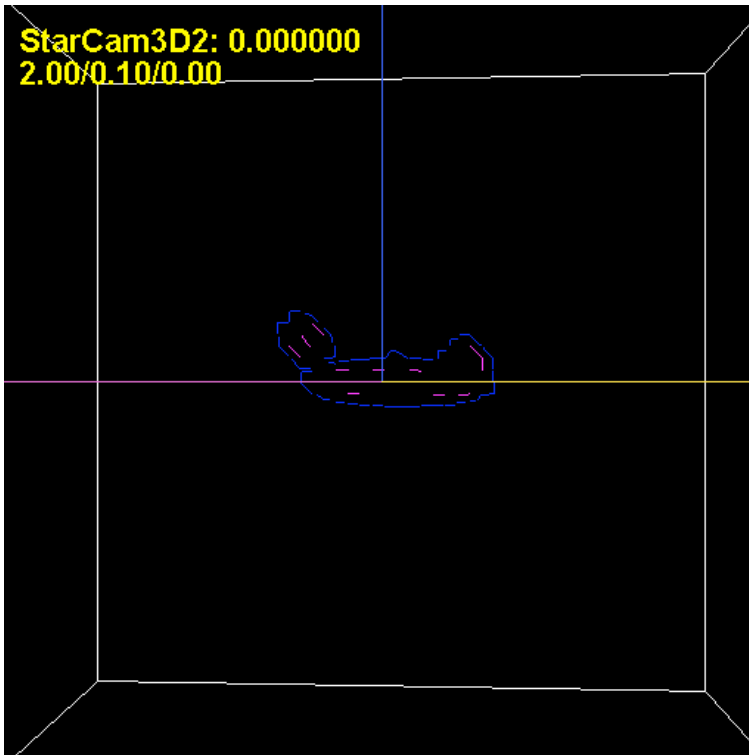


Figure 2: An image produced by Density as Contoured Sections in the yz plane.

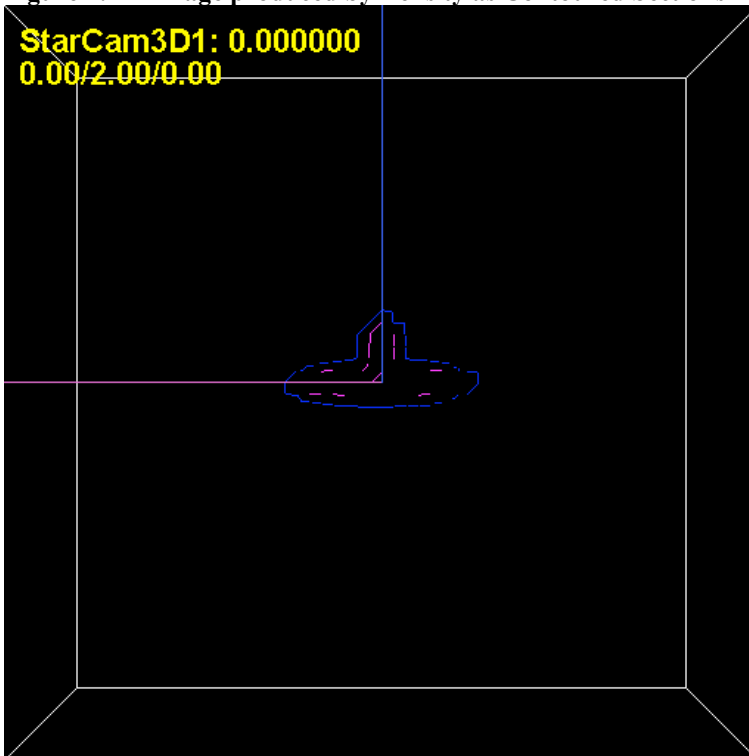


Figure 3: An image produced by Density as Contoured Sections in the xz plane.

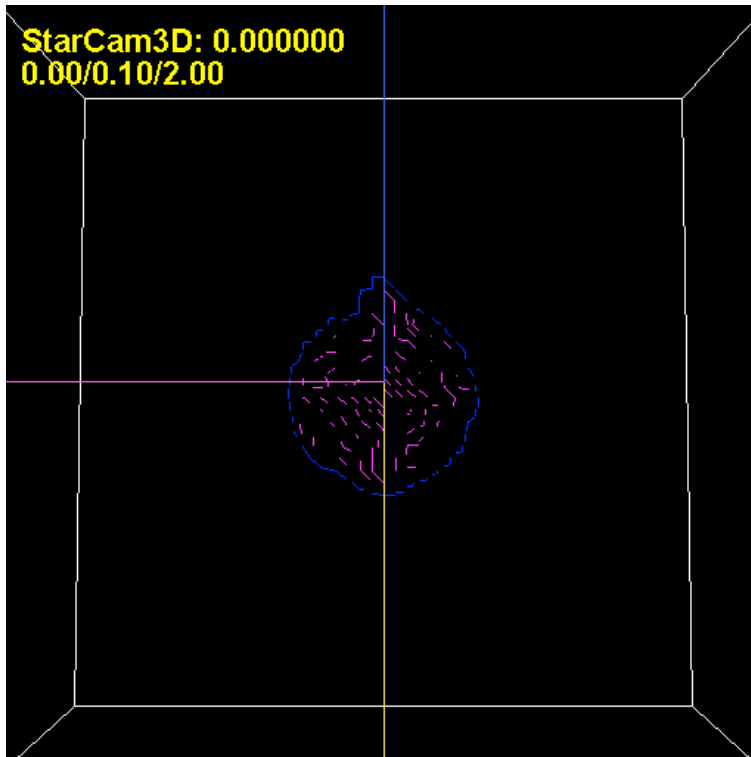


Figure 4: An image produced by Density as Contoured Sections in the xy plane.

As can be ascertained from the screenshots, the algorithm does not give the user a great idea of what is going on, only a very vague idea of where most of the density is concentrated, but not how it is concentrated in that area.

4.2) Density as Contour Follow

This method worked better than expected when the number of contours used was relatively low. The more contours there were, the less that was visible because contours would get so close together that nothing was clear. The final algorithm computed a very nice contour map that showed a very representative picture of the density in the simulation.

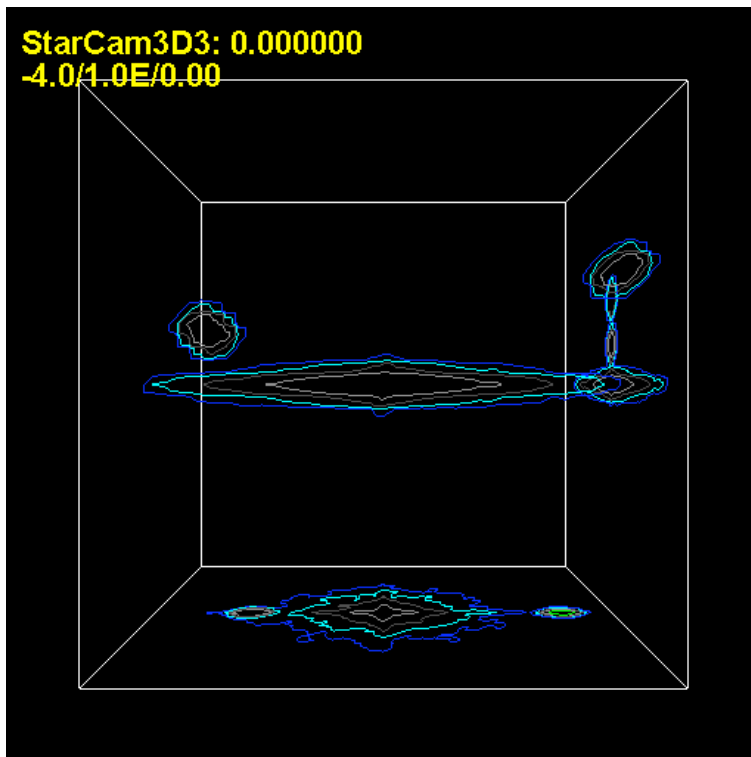


Figure 5: An image produced by Density as Contoured Sections in the yz plane.

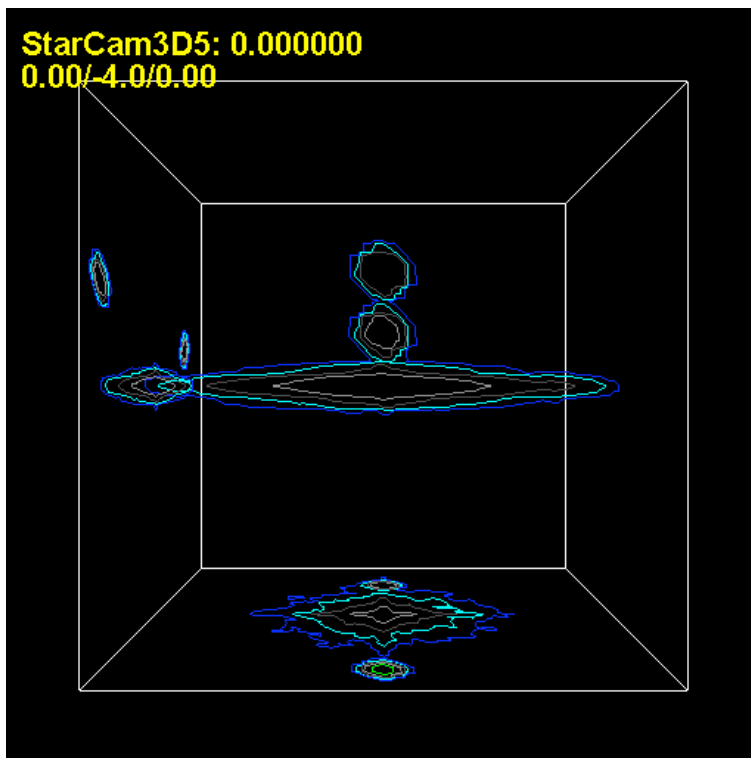


Figure 6: An image produced by Density as Contoured Sections in the xz plane.

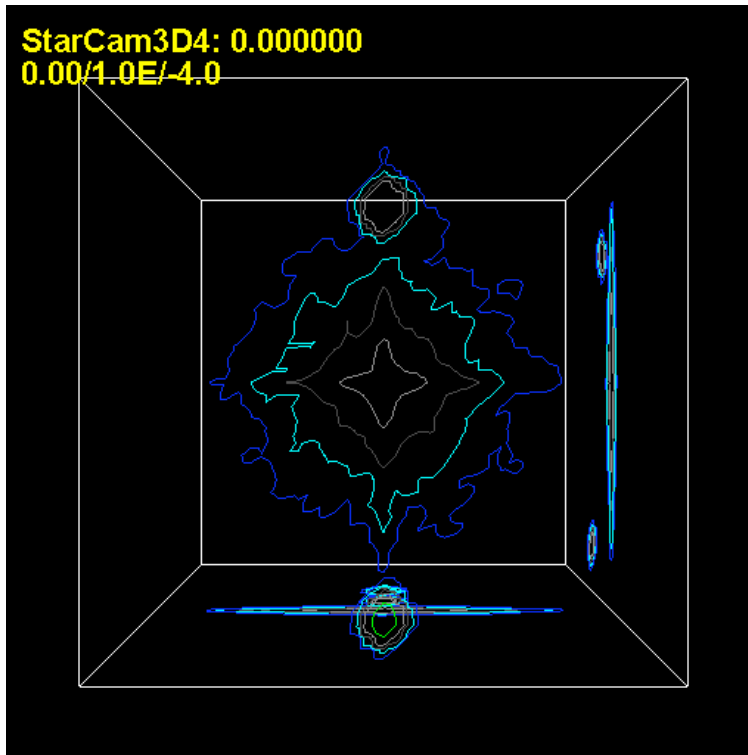


Figure 7: An image produced by Density as Contoured Sections in the xy plane.

As can be seen in the screenshots, the algorithm shows the different level curves correctly, and also the curves do not cross (as they shouldn't). One gets a relatively good idea as to where the densities are located, and as to how the density is concentrated in different areas. Note that the pictures are of the same cube, taken at different angles. This view can thus be moved around to encompass even more than one view per shot, allowing more information to be shown on the screen at any one time period.

4.3) Density as Slices

This method also worked better than expected. The “mountains” make viewing of the densities at each of the points on each of the planes easy and intuitive, making this the best overall method for viewing the density. When too many slices were used, the picture became blurred, but when using the right number of slices (see below) the picture came out relatively clearly.

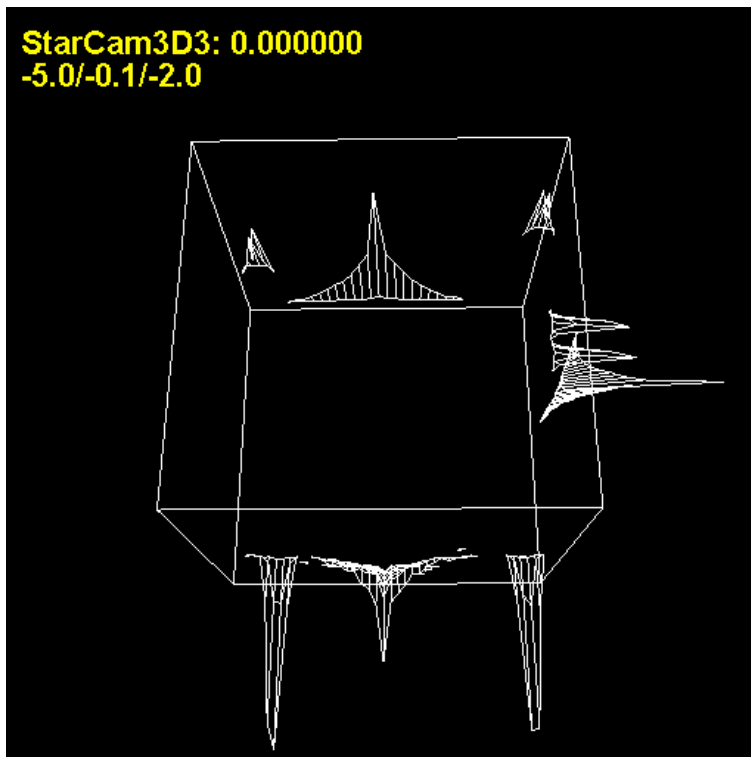


Figure 8: An image produced by Density as Contoured Sections in the yz plane.

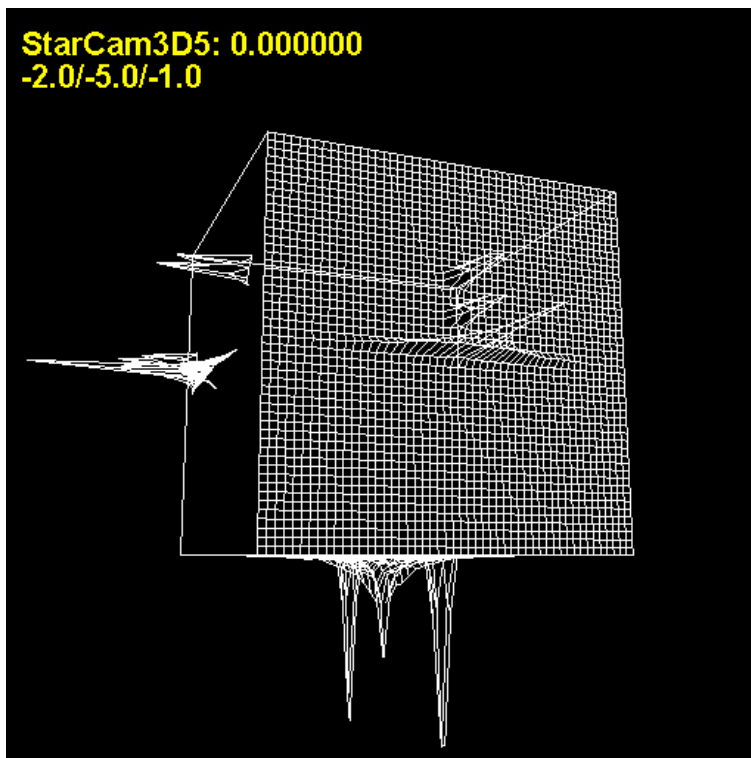


Figure 9: An image produced by Density as Contoured Sections in the xz plane.

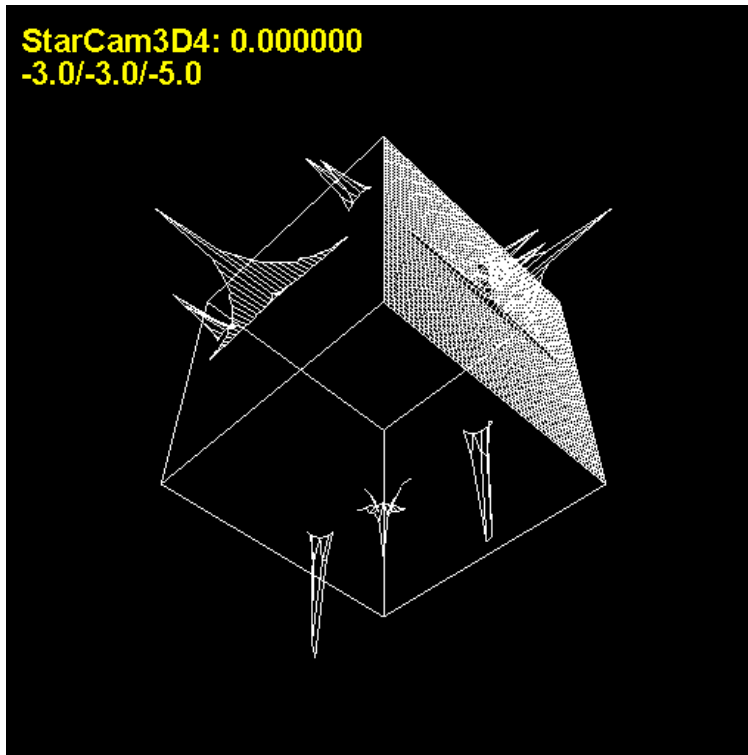


Figure 10: An image produced by Density as Contoured Sections in the xy plane.

As can be seen in the pictures, the mountains give a very good idea as to where the density is located in each of the planes. The only bug is that for some views, the lines representing negligible density are not shown. This is a reproducible bug which I cannot eliminate from the pictures themselves. This bug, however allows the user to come to a better conclusion as to which parts of the cube are of highest density, and which of the lowest density because there is no clutter of where the lowest density actually is.

5) Further Development

For further development I suggest refining method 3 (Density as Slices) to allow for coloring of different heights. This would allow for multiple cues to help the viewer determine the heights at various times during a moving presentation.

The other two methods (Density as Contoured Slices and Density as Followed Contour) I believe are not worth further development, unless it is minor tweaks to allow different coloring schemes to be applied. This is due to the fact that through all the work I put into those two methods, I feel little was accomplished by developing them compared to the amount of time I spent trying to code them, and that the time could have been better spent developing different methods of visualizing data.

In terms of other ways to visualize density, I believe that many ways can be built and expanded upon, and the ways presented here are not the best ways to approach the problem, but merely stepping stones that other visualization techniques can draw from and use to create an even better way of visualizing this abstract concept.