

# Visualization of Protein Field Data

Thomas Kerwin

## Introduction

Many of the tools and literature of modern protein research use only surface-based visualization to both represent their data to others in the field and to explore data to gain insight about the workings of the molecules and their interactions. There are many positive factors to surface-based visualizations: they have low hardware requirements and can be rendered very quickly. However, volumetric based visualizations can give greater depth of information for scalar field data in the vicinity of a protein. In this paper, I describe my application of volumetric visualization techniques and flow visualization to protein data.

## Getting data

Rather than computing scalar field quantities by myself, I decided to use a package called APBS<sup>[1]</sup> (Adaptive Poisson-Boltzmann Solver). APBS takes as input a protein structure format file in the .pdb format. This is a commonly used file format and includes all the atoms in the protein as well as connectivity information between the atoms. APBS can output many different types of fields. I chose to focus on accessibility and electrostatic potential.

I had to contend with several issues before using the field data directly for visualization. One problem was with the output file format that APBS produces, DX. Since I had no previous experience with this format, I made a reader for this format. I wrote a small program that converts the DX files into a raw binary format. This raw binary format has no absolute size information and I would have to input the physical dimensions of the data if I later wished to correspond the size of the data with some other data in a different scale space.

A second problem was related to the fact

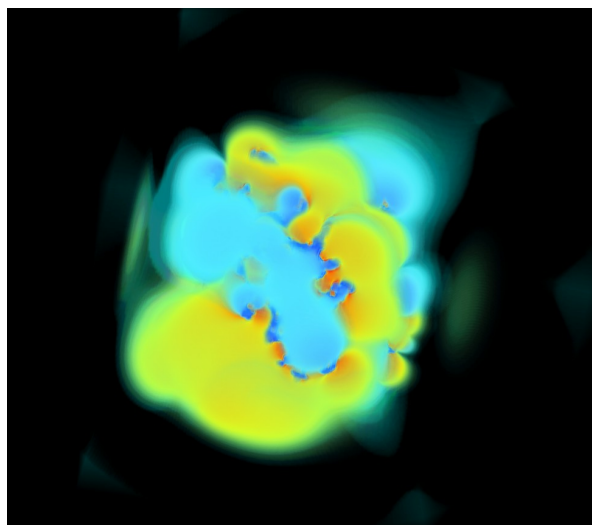
that the data was in 32-bit floating point format. This was not unexpected. All of the physical quantities represented in the scalar fields could be used for future numerical simulation and for that reason are not quantized but kept at full bit resolution for accuracy in those simulations. Unfortunately, floating point data is not easy to render directly using current graphics technology, which is primarily oriented towards four channel RGBA data with 8 bits per channel. There are some techniques that have been developed to render into floating point buffers and then use high dynamic range compression techniques, such as tone mapping, to convert that floating point buffer output into a 8-bit buffer.<sup>[2]</sup>

There are many different methods to convert a floating point dataset to fit into a range of only 256 values. For the accessibility value dataset, I use a simple linear scale. This is possible because of the data distribution. The data for accessibility ranges from 1 to 0, with a value of 1 represents a region of space fully accessible to the solvent surrounding the molecule, while a value of 0 represents a region of space completely inaccessible to the solvent. Between these values is a smooth blend at the boundary of the molecule. Therefore, using a straightforward bucketing method to lower the bit depth of the data gives acceptable results for understanding the data.

The electrostatic potential provides a greater challenge in lowering the bit depth of the data. The data ranges from around -1000 to 1000. The problem is that the data values are irregularly spaced in the dataset. The very high and low potential areas in the dataset are located inside the molecule. Therefore, attempting to do a linear mapping from the original data range to integers from 0 to 255 gives interesting information only in these high valued areas. However, from examining the data, it seemed that the interest-

ing parts of the data were actually in the periphery of the molecule. This makes sense when looking at interactions between molecules: the electrostatic potential in the area between the molecules should influence motion more directly than the high potential differences associated with interactions between atoms inside those molecules.

Therefore, I clamped the data to a much smaller range, on the order of -1 to 1. Picking this particular threshold gave much more fidelity to the parts of the dataset outside the traditional edges of the molecule. This, of course, could be easily modified by a researcher, depending on what region of the data set is deemed interesting.<sup>[3]</sup>



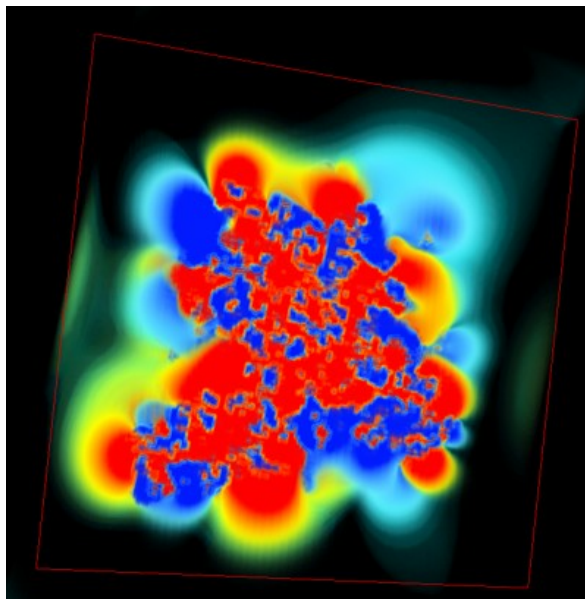
*Illustration 1: Volume rendering of electrostatic potential.*

## Volume rendering

After loading and transforming the electrostatic potential, I viewed the volume data and applied a transfer function to make it easier to understand. The results are shown in Illustration 1. The transfer function made areas with a higher absolute value of the electrostatic potential have a higher opacity value. Also, I colored areas with high potential blue and areas with low potential red. Deeper red or blue indicates a stronger field, either towards the negative or towards the positive, respectively.

Due to the hardware acceleration supplied by the graphics card I was using (a NVIDIA

GeForce 6800 GT), the volume rendering was real-time, rendering at 40 fps. A isosurface rendering is, of course, much faster, but in my experience, frame rates above 30 fps are more than sufficient for any kind of real-time visualization and exploration of data.



*Illustration 2: Volume rendering of electrostatic potential with a clipping plane.*

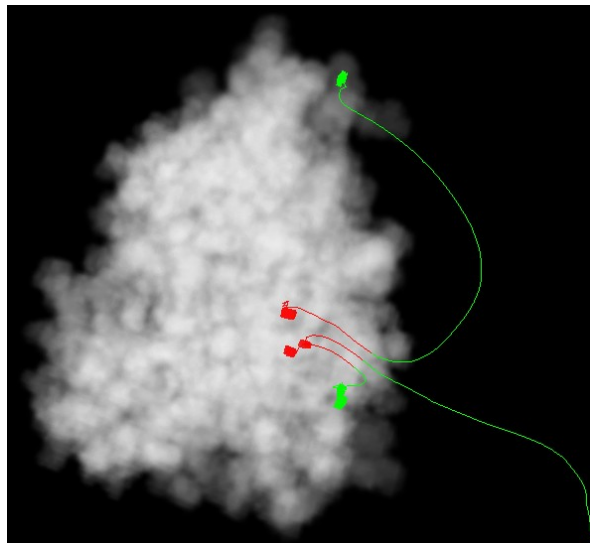
I used a clipping plane to slice into the data, cutting away part of it. As you can see in Illustration 2, the dark red and blue colors that appear internally in the molecule show areas of very high and very low potential. The clipping plane is interactive as well, and frame rates continue to stay above 30 fps.

## Pathlines

After trying the direct volume rendering of the potential, I developed an application to show the path a charged particle would take if affected by the electrostatic potential field surrounding the molecule. To do this, I found the gradient of the potential using central differences. A charged particle will follow this gradient to the maximum positive or negative charge, depending on the charge of the particle.

To give a better idea of the environment of the gradient, I sent two particles from the point selected by the user, one positive and one negative. The particle continues to be advected by the gradient in the voxel it occu-

pies until it has moved over 5000 spaces or it leaves the area in which the field is defined. The initial position of the particles is determined by the user: placement within the volume is manipulated by the keyboard. Recalculation of the new path and redisplay takes place instantaneously from the point of view of the user.



*Illustration 3: Multiple pathline display with volumetric accessibility rendering.*

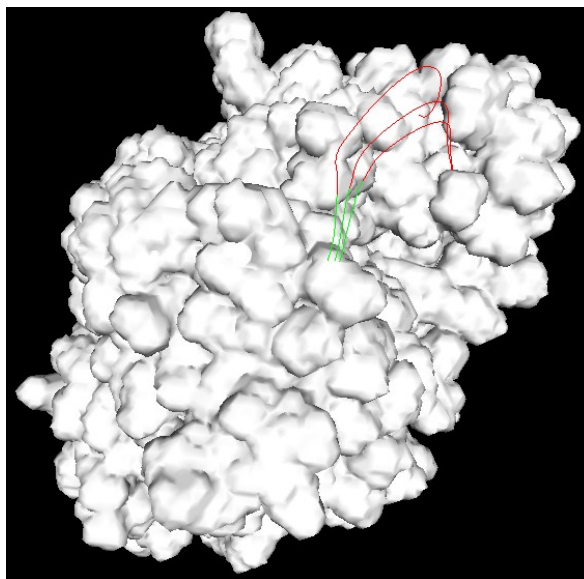
To enhance the utility of this application, I tried two approaches. The first was to add a volumetric rendering of the molecule in the same space as the pathlines. This provides the user with a view of the path of charged particles as they interact with the potential field around the molecule and a view inside as well. An additional utility of this rendering is that it shows the thickness of the molecule in the view direction. Thick areas are colored a brighter white than thin areas, giving a cloud-like appearance to the molecule.

The second enhancement I added was the rendering of additional pathlines. In Illustration 3, you can see the two additional bidirectional pathlines in the vicinity of the middle one. This is still very fast, although slightly slower than rendering only one bidirectional pathline. The effect of this enhancement is that a single image can show critical points on the gradient field. Multiple flow lines can show places, like in Illustration 3, where a small movement in space can give much different end results after advect-

ing the particle.

## Utility of isosurface output

In addition to the volumetric rendering of the accessibility field with the pathlines, I implemented a isosurface rendering of the field. The isosurface representation is similar to many of the surface models used in many common protein visualization packages. One problem with this sort of rendering is that the depth of the molecule is not readily apparent. The molecule must be seen from many angles in order to judge depth. Another problem is that isovalue for the surface must be chosen by the user and tweaked to remove artifacts.



*Illustration 4: Isosurface based visualization with multiple pathlines.*

As you can see in Illustration 4, the isosurface representation does not give the depth of information provided by the volumetric rendering. An isosurface implementation could provide transparency for the surface, but this would still not show the same richness of depth information that the volumetric rendering does. The isosurface does give a more concretely defined boundary between the molecule and the solvent around it. However, this boundary is somewhat arbitrary, since a different isovalue defines a different surface. A important positive feature of the isosurface rendering was that it renders much faster than the volumetric visual-

ization. This might be important when rendering very large proteins or a large number of proteins.

## Examples

For all my data processing and visualizations, I used a .pdb file of mouse acetylcholinesterase (PDB id 1MAH) as my initial input. Acetylcholinesterase is an enzyme found in neural pathways that breaks down the neurotransmitter acetylcholine into the component parts acetic acid and choline. All the illustrations seen here are taken from this protein.

## Future ideas

Even more potentially useful visualizations can be derived from combining these examples. I attempted to show the pathlines in the same visualization as the colored electrostatic potential volume rendering, but the results were too cluttered. Instead of pathlines for the visualization of the particles influenced by the field, streaklines could be used to show the movement of particles released all around the dataset but for a limited number of advection timesteps.

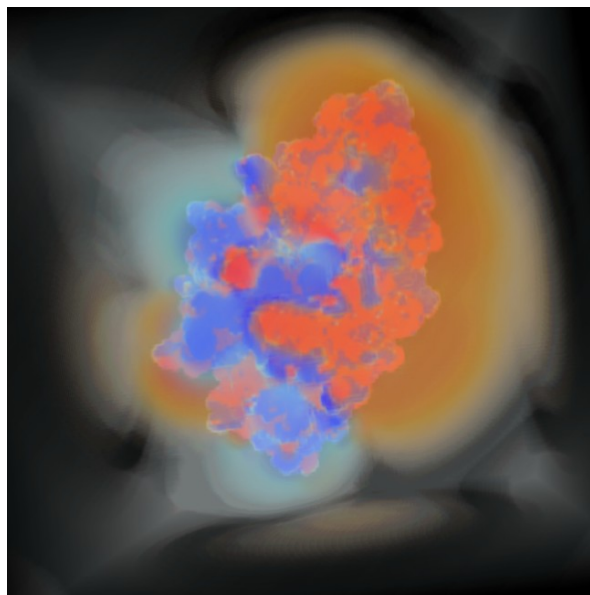


Illustration 5: Merged image from electrostatic potential and accessibility

Another combined visualization that could be very effective in the understanding of the structure of fields around the molecule is the

merging of volumetric datasets into one colored dataset. I combined the colors of the electrostatic potential data, after being processed with my red-blue color table, with the opacity values of the accessibility data. This gives a similar image as a isosurface representation, since the opacity of the merged dataset outside of the molecule's boundaries is 0, and therefore invisible. Then I merged this rendering with the rendering of the original electrostatic potential and created the image seen in Illustration 5. This shows both the somewhat chaotic behavior of the field at the surface of the molecule, and the smoother environment around the molecule in one image.

## Conclusion

Volumetric representations of proteins are as easy to use as isosurface representations, and they are nearly as fast to render. They do require a bit more preprocessing of the scalar data and the hardware requirements are higher. However, preprocessing can be automated and graphics hardware is fairly cheap: a modern generation graphics card can be purchased for less than \$150.

Pathline representation of vector fields like the electrostatic potential gradient is fairly common and easy to implement. It has the potential to assist in reasoning about movement of particles in the vicinity of proteins. Collaboration with biophysics researchers is necessary to determine with greater detail the types of questions that can be more easily solved with these types of visualizations and then more work can be done to target questions.

## References

- [1] Xiaoru Yuan, Minh X. Nguyen, Baoquan Chen, David H. Porter. High dynamic range volume visualization. In *Proceedings of IEEE Visualization 2005* pages 327-334, 2005.
- [2] Baker NA, Sept D, Joseph S, Holst MJ, McCammon JA. Electrostatics of nanosystems: application to microtubules and the ribosome. *Proc. Natl. Acad. Sci. USA* **98**, 10037-10041 2001.
- [3] I used the program VolSuite by Jason Bryan to facilitate conversions and transfer functions. <http://www.osc.edu/VolSuite>