ELSEVIER

Neurocomputing 32–33 (2000) 801–808

NEUROCOMPUTING

www.elsevier.com/locate/neucom

Visualizing the visual system

Greg Hood*, John Burkardt, Greg Foss

Pittsburgh Supercomputing Center, Mellon Institute Building, 4400 Forbes Ave., Pittsburgh, PA 15213, USA

Accepted 13 January 2000

Abstract

As part of a forthcoming planetarium show about the human brain, we are producing realistic models of the central nervous system at a variety of scales, from whole brain images to images of individual neurons. We have focused on the visual system and especially the visual cortex, and are building sets of simulated neurons using stochastic growth rules that mimic the geometries of actual neurons. We anticipate that some of the techniques we have developed in this work may also prove useful in various areas of computational neuroscience. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Visualization; Structural MRI; Visual cortex; Neuron morphology; Education

1. Introduction

"The Brain Project" [9] at Carnegie Mellon University is centered around the production of a planetarium-based interactive show which will help to give students a glimpse of how the brain operates. Though the show is targeted at the junior high school level, we anticipate that many people of all ages will see it. Our role in this project has been to produce animated visualizations that illustrate the structure and function of several components of the visual pathway.

One of the weaknesses of many prior animations of neural behavior is that they depict grossly oversimplified neurons with little sense of scale of how those neurons fit into the central nervous system as a whole. We are attempting to bridge this gap between the whole brain level and the cellular level by smoothly zooming in and out of

* Corresponding author. Tel.: + 1-412-268-7063; fax: + 1-412-268-8200.
E-mail address: ghood@psc.edu (G. Hood).

PII: S0925-2312(00)00247-2

the cortex. At the highest level (10 cm scale) we see sulci and gyri, at the next level (1 cm scale) a single gyrus, then (at 1 mm scale) distinct cortical layers with dense populations of neurons; next (at 100 mm scale) individual neurons become prominent; and finally (at 10 mm scale) synapses are visible.

The general strategy of our approach has been to develop an overall structural model of the brain's gray matter, and then to embed replicated copies of a more detailed neural model within that gray matter surface. This is similar in many respects to the "Exploring the brain forest" environment described in [12]. Since it would be impossible to build a neural model for more than a miniscule area of visual cortex, we construct a model only encompassing a 0.1 mm $\times$ 0.1 mm area (with 2.0 mm thickness), then tile the cortical surface with this pattern. The neural model is built in such a way that axons and dendrites wrap-around so that the tiling appears seamless, and one cannot easily discern the boundary where one tile ends and another begins.

2. Building a geometric model of gray matter

We begin with a conventionally obtained structural MRI image, in which different intensity levels roughly correspond to differences in fat/water ratios in the tissues. Gray matter accordingly appears slightly darker than white matter in the images.

For the first step, we used the mrGray program [6] to apply a thresholding operation to classify voxels into white matter or gray matter or CSF. It is impossibile, however, to pick a threshold that will be entirely correct over the whole brain, so some manual cleanup is necessary. We want to have a set of white matter voxels whose surface is topologically equivalent to a sphere. Thus, we use another program (called "handler") to identify those small regions where white matter forms loops. Many of these are due to white matter of adjacent folds coming very close together, and a few voxels must be relabeled to eliminate these bridges. In other places, the proper correction is to fill in voxels to eliminate holes which should not be present. This step turned out to be quite labor-intensive. Although the initial segmentation appeared relatively clean, it turned out that we had over 200 handles that each had to be inspected and manually corrected.

Once the handles had been removed, we "grew" one layer of gray matter using the mrGray program, and saved the resulting surface. A single slice through the final result is shown in Fig. 1, with the whitened voxels classified as white matter, and darkened voxels classified as gray matter. The surface information consists of a list of voxels, and the connectivity relationships among voxels. Voxels are connected if they are adjacent in a 26-neighbor sense in 3-space, and if they arose from the same underlying white matter. This means that gray matter voxels which happen to be adjacent but which arose from white matter on opposite sides of a sulcus will not be connected.

With this surface connectivity information we then attempted to "flatten" the gray matter using two different flattening programs, mrUnfold [8] and CARET/FLAT-MORPH [10]. Unfortunately, we ran into obstacles with both, as they are more oriented toward flattening smaller regions than the entire cortical surface of both

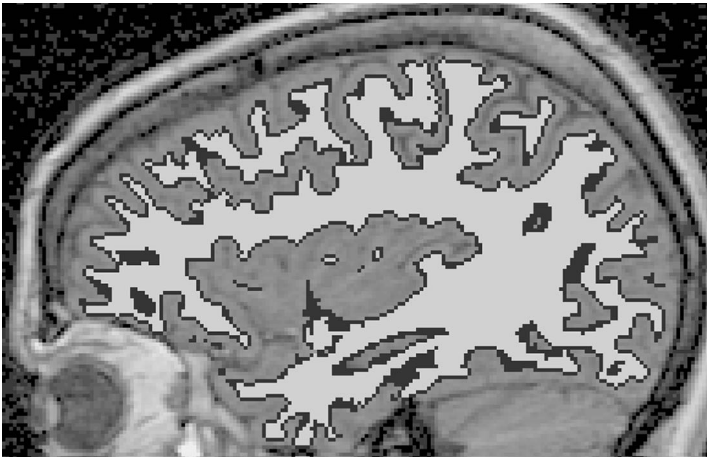


Fig. 1. Surface voxels in a single MRI slice.

hemispheres. We then developed a flattening method based on a relaxation/simulated annealing approach. In the first phase of this, points on the cortical surface are migrated until they lie approximately on the surface of a sphere, and also are distributed fairly uniformly over the surface of the sphere. While gross spatial relationships are roughly preserved during this step, there is no attempt to preserve local spatial relationships, so many overlaps occur. As the "temperature" of the annealing process is reduced, we gradually transition to a relaxation process in which local spatial relationships are restored, and these overlaps are removed. Unfortunately this is unstable, and if left to run too long, will begin to distort the overall map, with points gradually becoming congested instead of spread over the entire sphere. By terminating this second phase before much overall distortion has occurred, we end up with the lower gray matter surface (bottom of layer 6) unfolded onto a sphere.

Once we have all the points mapped on the spherical surface, we simply use a latitude/longitude projection to then map these onto a flat rectangle. Fig. 2 visualizes the result of this flattening process, with each point representing one (bottom) surface point on the gray matter. Points are colored according to how far they are from the center point of the corpus callosum, with the color spectrum repeated several times to show smaller distance changes more easily. The left hemisphere is on the left, the right hemisphere on the right, with the occipital regions at the bottom of the figure.

We use this to interpolate locations (in original 3D space) for a rectangular grid in flattened space (in our case every 0.5° of latitude and longitude). The interpolation algorithm we used is ACM Algorithm 773 developed by Robert J. Renka [5], which also incorporates smoothing. Finally, we calculate surface normals and take into account the thickness of the gray matter, to produce a visualization of the top cortical surface (top of layer 1) as seen in Fig. 3 from the top. Since we have effectively imposed a grid onto the cortical surface, we also have a means of tiling this surface with our neural model.

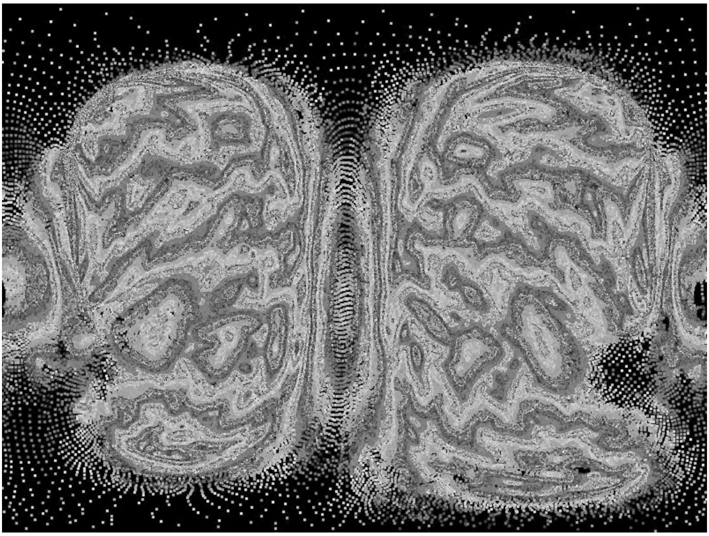


Fig. 2. Flattened left and right hemispheres.

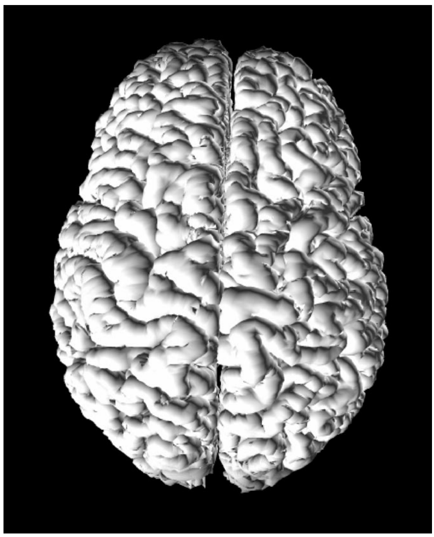


Fig. 3. Smooth brain surface reconstructed after unflattening the gridded surface.

3. Building a geometric model of a chunk of visual cortex

In building our geometric model of individual neurons of the visual cortex, we have relied largely on the description of macaque visual cortex given in [3,4]. We are extrapolating this to the human cortex by scaling layer thicknesses and making other modifications according to data in [2]. In order to populate these layers with life-like

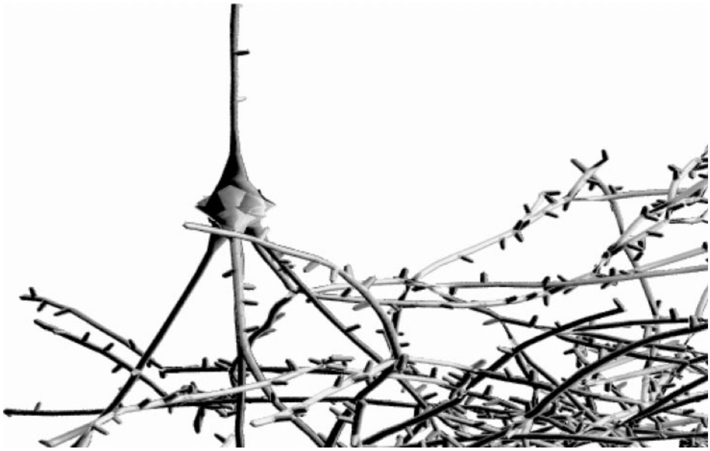


Fig. 4. Artificial pyramidal cell.

neurons, we are "growing" simulated neurons that have geometries resembling those of actual cortical neurons. We start each neuron off as a bare soma, and then grow the axon and dendritic tree using a set of stochastic growth rules (specific to that neuron type) that empirically result in a realistic branching structure. An example of such a "grown" pyramidal cell is shown in Fig. 4.

One of the challenges of constructing realistic models of cortical neurons in the computer has been their sheer size, since a typical neuron has thousands of connections to other neurons, and dendritic trees which extend through a matrix of other neurons' dendritic trees. Even with the rapid increase in computer memory capacity over the years, the size of primary memory still constrains the complexity of models which may be built. We have therefore, attempted to eliminate redundancy in our representations so as to minimize the storage consumed to encode the morphology of the neurons. We model the neurons as a large number of cylinders, and have reduced the storage requirements of each single cylinder to asymptotically approach 8 bytes as their number becomes large, while still preserving the ability to randomly access components geometrically in constant time and traverse dendritic trees in a forward and backward direction.[1]

In order to present the model to the viewer we are exploring a variety of rendering techniques. A key idea is to make smooth transitions during pans and zooms, so we must be able to efficiently display large numbers of small objects (e.g. sections of dendrites) at reduced resolution. A naive mapping of every visible neuron into a fixed number of polygons would quickly lead to impractically long rendering times. Hence, we have adopted multi-resolution strategies that employ many polygons when neurons are close-up but a smaller number when a neuron is distant. It is also necessary to fade out a large percentage of neurons at certain times, since attempting

[1] For details of this representation, please contact the authors.

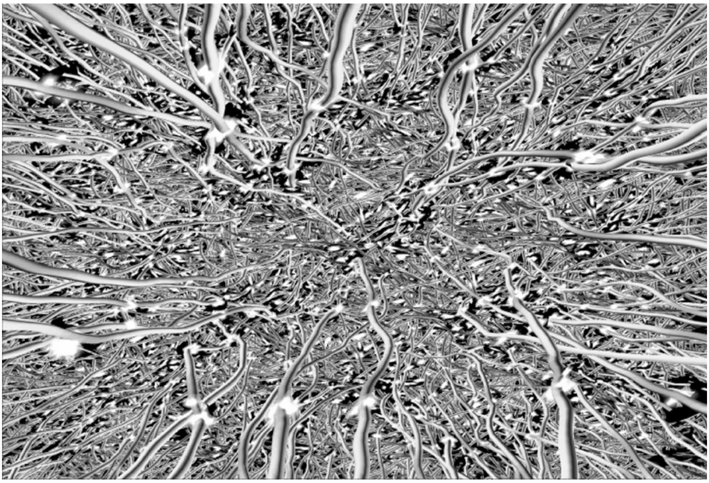


Fig. 5. View looking from layer III up toward layer I in V1.

to present the complete model at realistic densities would likely produce the impression of an impenetrable thicket of dendrites and axonal arborizations. Finally, use of a ray-tracing renderer (POV-Ray [11]) allows us to incorporate dramatic lighting and surface effects such as translucency.

The final step is producing the visualizations is to map the cell-level neural model into the tiles defined by the latitude–longitude grid on the flattened cortical surface. For this, we are using a transfinite interpolation technique [1,7]. In this process, we use splines to ensure the smoothness of both the top and bottom surfaces of the gray matter layers. An example scene resulting from this combination of low- and high-level models is shown in Fig. 5. Approximately 30 of the 0.1 mm $\times$ 0.1 mm tiles were used to form this scene.

4. Conclusions

Although the primary goal of our work is to generate captivating visualizations of the central nervous system for a general audience, we believe that some of our techniques will transfer over to computational neuroscience. Since we are "growing" neurons, the present work bears some resemblance to developmental modeling. Whereas we use growth rules that depend on physical variables such as distance from the soma, position within cortical layer, and direction relative to the cortical surface, an accurate developmental model would also likely include dependencies on neural activity, both of the neuron itself and that of adjacent neurons. Our data structures could also be extended to support multi-compartment simulation of a neuron's electrical activity, and form a substrate for investigating neural computation that might be dependent on the geometric properties (e.g. dendritic branching patterns) of particular neurons.

Another application of the compact geometric representation scheme is in efficiently storing large quantities of experimentally obtained neural morphologies. With the prospect that high-resolution scans of cortical tissue slices will generate petabytes of raw data, efficient schemes of encoding geometries will become important.

Finally, the brain "flattening" method shows some potential for being used in the context of psychological research involving brain mapping. We hope to more carefully evaluate this technique in comparison with existing methods of flattening, since it appears to take much less computational time, and is able to deal more easily with larger cortical regions.

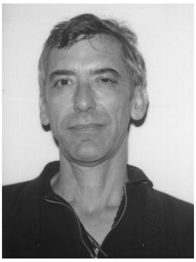
Acknowledgements

This research was supported in part by NSF Grant #9705491 from the National Science Foundation, and PSC Grant #SEE 970013P from the Pittsburgh Supercomputing Center.

References

[1] W.N. Gordon, C.A. Hall, Construction of curvilinear coordinate systems and application to mesh generation, Int. J. Numer. Methods Eng. 7 (1973) 461–477.
[2] R.O. Kuljis, The human primary visual cortex, in: A. Peters, K.S. Rockland (Eds.), Cerebral Cortex, Vol. 10, Plenum Press, New York, 1994.
[3] A. Peters, C. Sethares, Organization of pyramidal neurons in area 17 of monkey visual cortex, J. Comp. Neurol. 306 (1991) 1–23.
[4] A Peters, The organization of the primary visual cortex in the macaque, in: A. Peters, K.S. Rockland (Eds.), Cerebral Cortex, Vol. 10, Plenum Press, New York, 1994.
[5] R.J. Renka, Algorithm 773: SSRFPACK: interpolation of scattered data on the surface of a sphere with a surface under tension, ACM Trans. Math. Software 23 (3) (1997) 435–442.
[6] Teo, Sapiro, B. Wandell, IEEE Med. Trans. 16 (6) (1997) 852–863.
[7] J. Thompson, B. Soni, N. Weatherill, Handbook of Grid Generation, CRC Press, BocaRaton, FL, 1999.
[8] B. Wandell, http://white.stanford.edu/ ~ brian/mri/mrUnfold.html.
[9] http://occipita.cfa.cmu.edu/brain.
[10] http://stp.wustl.edu/.
[11] http://www.povray.org.
[12] B.P. Burton, T.S. Chow, A.T. Duchowski, W. Koh, B.H. McCormick, Exploring the brain forest, Neurocomputing 26–27 (1999) 971–980.

Greg Hood is a Neuroscience specialist at the Pittsburgh Supercomputing Center, where he has been involved in parallel software development for neuroscience applications since 1995. He contributed to the PGENESIS neural simulator, and has worked on near-realtime image processing codes for fMRI. He has an M.S. in Computer Science from Carnegie Mellon University.



John Burkardt is a Biomedical Programmer at the Pittsburgh Supercomputing Center, providing support in algorithm development. For this project, he constructed formulas and algorithms needed for a variety of geometric transformations. He has a Ph.D. in Mathematics from Virginia Tech.

Greg Foss is a computer graphics specialist providing scientific visualizations for the Pittsburgh Supercomputing Center since 1993. He works with a wide range of researchers and produces data visualizations for print, video, world wide web, and stereoscopic viewing. He holds a Master of Arts in Computer Graphics and Animation from the Ohio State University Advanced Computing Center for the Arts and Design.