An efficient method for compressing neuron morphology data

Longfei Li*

1College of Computer Engineering and Science, Shanghai University, Shanghai, Shanghai, 200000, China
*Corresponding author’s e-mail: lilongfei@shu.edu.cn

Abstract. The human brain has a very complex network structure. In order to understand the information integration process of the neural network in the brain, the current research of computational neurology focuses on studying the laws of neuron morphology. In recent years, the reconstruction technology in the field has developed rapidly, and the production speed of neuron data has been greatly accelerated. The problem of neuron data management under the background of Big-Data has gradually emerged. This paper proposes a method for compressing neuron morphology data based on binary coding, so as to efficiently store neuron morphology data. Relevant experimental results prove that the method in this paper is widely applicable to various types and sizes of neuron data.

1. Introduction
At the end of the 19th century, Cajal created neuron cell illustrations by hand drawing, unlocking the prelude to the development of neuron morphology research. In the past one hundred years, fluorescent and molecular labeling technology have developed rapidly, based on the three-dimensional neuron data visualization and analysis platforms like Vaa3D[1], Neurulucida[2], neuron tracing algorithms such as Virtual finger[3], SmartTracing[4] and APP2[5] have generated a large amount of neuron morphology data, which provides a solid research foundation for neuroMorpho[6], FlyCircuit[7] and other data platforms and research studios. As of 2021, NeuroMorpho, as the world's largest open neuron morphology database, has collected more than 147,472 neuron morphology data and provided more than 13,894,177 downloads.

At present, the research of neuron morphology has entered the era of the whole-brain. Under this background, the problem of large neuron size (usually containing 10,000 to 20,000 nodes, which can reach a file size of more than 300KB) emerges. In order to map the entire brain neurons more comprehensively, the magnitude of neuron data required can reach the GB-level. In this context, it is undoubtedly necessary and meaningful to use high-efficiency neuron data compression methods to store neuron data.

2. Definition
In the field of computational neurology, neuron three-dimensional morphological data is usually expressed in SWC file format [8]. In the SWC file format, neurons are abstracted into interconnected tubulars or spheres and stored in the form of a tree-like data structure. Each node in the tree records the location, radius, category and 3D sampling point location in the reconstructed neuron. As shown in the Figure 1, each line in the SWC text file has the same 7 field values: index value, morphological category of each segment, three-dimensional coordinate information (x, y, z), segment termination radius
information, segment Parent fragment index value information. The parent index value of the root node is always -1, which represents the SOMA part of the neuron.

![Figure 1: The value fields in neuron SWC file.](image)

In fact, since each branch of a neuron (the segment from the bifurcation point to the bifurcation point or the bifurcation point to the terminate point) is actually a continuous biological structure, there is a curve or a set of curves that can fit each branch of the neuron. Based on this idea, this paper designs a neuron compression method with spline curve fitting algorithm, combined with a binary field encoding strategy, the neuron can be represented with an extremely high file compression ratio.

3. Method

3.1. Hermite spline
The Hermite spline\[9\] algorithm is often used to fit a curve. According to the values of two sampling points in three dimension and their first derivative information, a given value can be interpolated to obtain a continuous smooth curve function. With cubic Hermite spline curve $H(t)$, given $t \in [0, 1]$, the interpolation value of the curve in a certain dimension can be obtained according to a set of inputs.

3.2. Fit a branch
In this paper, the following algorithm is designed to fit the neuron structure, which is divided into four steps:

1. Decompose neurons into a tree structure with branch as the unit.
2. The two end points of each branch are used as sampling points, the position information and tangent information of the sampling points in the three dimensions (X, Y, Z axis) are calculated as the Hermite spline curve parameters.
3. According to the spline curve parameters obtained by Hermite algorithm, reconstruct a branch curve with a same number of sampling points.
4. Calculate the paired Euclidean distance between the reconstructed branch and the original branch as the reconstruction error. Compare this error with a error threshold. If the error is less than the threshold, the curve obtained by parameter fitting meets the requirements and the processing is complete; if the error is greater than the threshold, the point with the largest error will be used as the new endpoint to divide the original branch into two new branches, then repeat step 2.
After the fitting step is completed, the neuron can be represented as a collection of Spline Nodes. Each Spline Node contains node ID information, curve parameters, parent index, type information, and segment radius information, as shown in Figure 2.

![Figure 2. Compressed neuron structure.](image)

3.3. Encoder part
The data structure obtained by curve fitting is mainly composed of int or float numeric types. This can be expressed in the form of Tag-Value. In the implementation of this article, the two types of numeric values are implemented in the following encoding methods:

**Varints encoding.** In the implementation of Varints encoding, the binary code is split into groups with size 7, stored with the group for low bit first and the high bit last. A MSB bit is first set before each group to indicate whether it is necessary to read the next byte content. If the MSB is 1, continue to read the content, if the MSB is 0, the reading process is terminated. This encoding form is suitable for encoding unsigned numbers. When the value is small, the extra high bit 0 can be removed, saving a lot of unnecessary data bits.

**ZigZag encoding.** When using the Varints method to encode a signed negative number, the sign bit of the highest bit will cause the compression of this method to fail. At this time, the ZigZag encoding method is required. Briefly, it is first mapped a signed integer number to unsigned number, then subjected to an unsigned number coding.

ZigZag coding can be implemented with a hash function hash(n) to achieve O(1) level of space and time complexity, such as the formula:

\[
\text{hash}(n) = (n ≪ 1)^{\wedge}(n ≫ 31)
\]  

ZigZag decoding can be implemented by function \(\text{hash}^{-1}(n)\), such as the formula:

\[
\text{hash}^{-1}(n) = (n ≫ 1)^{-((n & 1))}
\]  

The corresponding relationship of integer's complement is shown in Table 1, (expressed in hexadecimal).
| Value | Hex     | Zigzag   |
|-------|---------|----------|
| 0     | 00 00 00 00 | 00 00 00 00 |
| -1    | FF FF FF FF  | 00 00 00 01 |
| 1     | 00 00 00 01 | 00 00 00 02 |
| -2    | FF FF FF FE  | 00 00 00 03 |
| 2     | 00 00 00 02 | 00 00 00 04 |
| ...   | ...   | ...   |
| -64   | FF FF FF C0  | 00 00 00 7F |
| 64    | 00 00 00 40 | 00 00 00 80 |
| ...   | ...   | ...   |

**Bytes encoding.** In addition, because Varints encoding needs to "waste" a sign bit as the MSB bit, Varints encoding will waste bits when the value is greater than 2^28 (2^56 when encoding a 64-bit value). In this case, we directly save the corresponding secondary system value, that is, no more compression.

4. Experiments

Based on the compression method designed in this article, we tested the compression efficiency of the algorithm when it is used to compress neuron SWC files. Three types of neurons downloaded from NeuroMorpho: 289 GABAergic, 261 granule, and 157 pyramidal neurons are compressed. Figure 3 shows the compression ratio distribution of the three types of neurons. It can be seen that for the three types of neurons, the method effectively compresses them. Since Granule neurons often have long and simple branches, the compression effect is even more than 0.02.

![Figure 3. Distributions of neuron compression ratio.](image)

We also performed compression algorithm experiments on 219 whole-brain level neurons, and directly used the encoding method to encode the original SWC file as a baseline experiment. Figure 4 shows the compression ratio distribution of the two methods. It can be seen that the baseline method only compresses about half of the file volume. The number of sampling points and the number of branches in the whole brain neurons makes the compression effect reach below 0.06.
5. Conclusion
This article proposes a compression method for neuron morphology data. Experiments show that our method has achieved excellent compression effects for non-whole brain and whole brain level neurons, and this advantage is even more obvious for whole brain neurons. In the context of the rapid growth of neuron data, the compression method in this chapter has good application prospects.

References
[1] H. Peng, Z. Ruan, F. Long, J. H. Simpson, and E. W. Myers, “V3D enables real-time 3D visualization and quantitative analysis of large-scale biological image data sets,” Nature Biotechnology, vol. 28, no. 4, pp. 348–353, Apr. 2010, doi: 10.1038/nbt.1612.
[2] J. R. Glaser and E. M. Glaser, “Neuron imaging with neurolucida — A PC-based system for image combining microscopy,” Computerized Medical Imaging and Graphics, vol. 14, no. 5, pp. 307–317, Sep. 1990, doi: 10.1016/0895-6111(90)90105-K.
[3] H. Peng et al., “Virtual finger boosts three-dimensional imaging and microsurgery as well as terabyte volume image visualization and analysis,” Nature Communications, vol. 5, no. 1, Art. no. 1, Jul. 2014, doi: 10.1038/ncomms5342.
[4] H. Chen, H. Xiao, T. Liu, and H. Peng, “SmartTracing: self-learning-based Neuron reconstruction,” Brain Inf., vol. 2, no. 3, pp. 135–144, Sep. 2015, doi: 10.1007/s40708-015-0018-y.
[5] H. Xiao and H. Peng, “APP2: automatic tracing of 3D neuron morphology based on hierarchical pruning of a gray-weighted image distance-tree,” Bioinformatics, vol. 29, no. 11, pp. 1448–1454, Jun. 2013, doi: 10.1093/bioinformatics/btt170.
[6] G. A. Ascoli, D. E. Donohue, and M. Halavi, “NeuroMorpho.Org: A Central Resource for Neuronal Morphologies,” J. Neurosci., vol. 27, no. 35, pp. 9247–9251, Aug. 2007, doi: 10.1523/JNEUROSCI.2055-07.2007.
[7] A.-S. Chiang et al., “Three-Dimensional Reconstruction of Brain-wide Wiring Networks in Drosophila at Single-Cell Resolution,” Current Biology, vol. 21, no. 1, pp. 1–11, Jan. 2011, doi: 10.1016/j.cub.2010.11.056.
[8] R. C. Cannon, D. A. Turner, G. K. Pyapali, and H. V. Wheal, “An on-line archive of reconstructed hippocampal neurons,” Journal of Neuroscience Methods, vol. 84, no. 1, pp. 49–54, Oct. 1998, doi: 10.1016/S0165-0270(98)00091-0.
[9] J. R. and C. de Boor, “A Practical Guide to Splines.,” Mathematics of Computation, vol. 34, no. 149, p. 325, Jan. 1980, doi: 10.2307/2006241.