

# Compression and Interactive Visualization of Terabyte Scale Volumetric RGBA Data with Voxel-scale Details

Mehmet Oguz Derin  
Morgenrot, Inc.  
Türkiye  
oguz@morgenrot.net

Takahiro Harada  
Morgenrot, Inc.  
, Advanced Micro Devices,  
Inc.  
USA  
takahiro.harada@amd.com

Yusuke Takeda  
Hokkaido Univ.  
Japan  
ytakeda@sci.hokudai.ac.jp

Yasuhiro Iba  
Hokkaido Univ.  
Japan  
iba@sci.hokudai.ac.jp

## ABSTRACT

We present a compressed volumetric data structure and traversal algorithm that interactively visualizes complete terabyte-scale scientific data. Previous methods rely on heavy approximation and do not provide individual sample-level representation when going beyond gigabytes. We develop an extensible pipeline that makes the data streamable on GPU using compact pointers and a compression algorithm based on wavelet transform. The resulting approach renders high-resolution captures under varying sampling characteristics in real-time.

## CCS CONCEPTS

• **Computing methodologies** → **Rendering; Ray tracing.**

## KEYWORDS

compression, sparse volumes, data structures, ray tracing, rendering, visualization

### ACM Reference Format:

Mehmet Oguz Derin, Takahiro Harada, Yusuke Takeda, and Yasuhiro Iba. 2022. Compression and Interactive Visualization of Terabyte Scale Volumetric RGBA Data with Voxel-scale Details. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Posters (SIGGRAPH '22 Posters)*, August 07-11, 2022, ACM, New York, NY, USA, 2 pages. <https://doi.org/10.1145/3532719.3543256>

## 1 INTRODUCTION

In scientific visualization, capturing and rendering high-resolution volumetric data is an active area of research for both Earth and extraterrestrial material samples. Although there are developments in compression and rendering [Graciano et al. 2021] [Aleksandrov et al. 2021] [Wald et al. 2017], no work has been there which enables the display of RGBA volumes with sizes exceeding 19,000 x 12,500 x 4,000 on 8K screens, where the preservation of voxel-level structures is crucial for findings through interactive exploration. We propose a method that scales to terabyte-scale volumetric data, where our main contributions are a data structure that captures sparsity and compresses, a construction approach that parallelizes the demanding aspects of the building of our data structure, and

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

*SIGGRAPH '22 Posters*, August 07-11, 2022, Vancouver, BC, Canada

© 2022 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-9361-4/22/08.

<https://doi.org/10.1145/3532719.3543256>

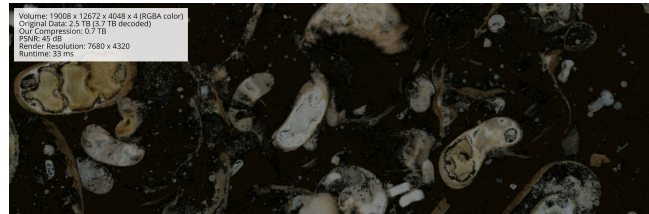


Figure 1: Render of a high-resolution volumetric capture of a rock sample with annotation of statistics and performance.

finally, a rendering approach that reuses stack and hierarchical DDA [Museth 2021] to accelerate content on GPU while streaming.

## 2 METHOD

For interactive visualization at this scale with high unpredictability of fetch patterns, we process our data into an efficient format.

### 2.1 Data Structure

Our data structure is a tree where each internal node contains, depending on the build-time configuration,  $N^3(2^3 \text{ to } 6^3)$  children, which can be null, and minimum color, average color, and maximum color attributes for our compression, besides offsets for children (see Fig. 2). Since we compress the node offset to 8-bits, the maximum value we can use for  $N$  is 6. These pointers are offsets to a 32-bit address stored within the node, introducing the primary challenge of parallelization with children exporting to a linear buffer at each level needing to be sequential, besides the secondary challenge with the need for recursive color attributes to achieve wavelet compression. At the bottom level, we contain similarly sized arrays to obtain voxel color values, enabling storage of values starting from 4-bit channels. Reconstruction of a value at a single level is either  $x_r^i = (max^{i+1} - ave^{i+1})x_c^i + ave^{i-1}$  or  $x_r^i = (ave^{i+1} - min^{i+1})x_c^i + min^{i-1}$ , where  $x_r^i, x_c^i$  are reconstructed and compressed values respectively. We select one equation depending on the sign bit we store. This is executed recursively from the root to the bottom to reconstruct voxel values.

### 2.2 Construction

**2.2.1 First Pass.** The first pass is a bottom-up process that starts with building nodes at level 0. To achieve parallelization, which is necessary due to the scale of the data, we first run a pass over the data to build nodes at level 0. Since our input data measures in terabytes, for which it is difficult to allocate all the nodes in the tree on the RAM, we use a hashmap to keep track of nodes we

