Hierarchical Volume Rendering

Big Data Problem ...

One way to handle the big data problem is to use hierarchical data structures (hierarchical volumes)

Less amount of data are required

Issues

- Creation of hierarchical data structures
- Utilization of hierarchical data structures

Hierarchical Volume

A hierarchical volume can be constructed using the octree

Octrees

- An octree can be full or non-full
  - Full octree - each parent node has exact 8 children
  - Condition: all the dimensions are the same power of 2
  - A full octree example: 16 x 16 x 16
    - Level 1: (8) 8 x 8 x 8; level 2: (8) 4 x 4 x 4
    - Level 3: (8) 2 x 2 x 2; level 4: (8) 1 x 1 x 1

Octrees (2)

- For a volume with resolution s in each dimension, the total number of tree nodes is:
  \[ 8^0 + 8^1 + \ldots + 8^{\log_2 s - 1} = \frac{(s^3 - 1)}{7} \]

- Node to data ratio: \( \frac{(s^3 - 1)}{7s^3} = 0.1428 \) this is the optimal ratio

- If a volume can not be represented by a full octree, the node to data ratio will increase -> overhead increases
Octrees (3)
- Other benefit of full octree - Can be stored in a linear array without using pointers
  - Each parent has exactly 8 children - their positions in an array are totally predictable
  - For a parent stored at T[k], then its eight children are stored at T[8k+1], T[8k+2], ..., T[8k+8].
  - All the nodes at the same level are stored in a contiguous memory space

Octrees (4)
- Power of 2 volume is not a norm
- Full-octree thus is not always possible
  - A non-full octree example: 16 x 8 x 4
    - Level 1: (8) 8 x 4 x 2; level 2: (4) 4 x 2 x 1
    - Level 3: (4) 2 x 1 x 1; level 4: (2) 1 x 1 x 1
  - Total number of nodes: 1 + 8 + 8^2 + 8^3 = 161
  - The node to data ratio = 161 / 512 = 0.314 > optimal (0.1428)
  - Also, we need to store the pointers from parents to children

Octrees (5)
- In general, there are two ways to construct an octree for a volume
  - Top-down:
    - 1 -> 8 -> 8x8 -> 8x8x8x4 -> ...
    - n (n = total number of data)
  - Bottom-up:
    - n -> n / 8 -> n / 8x8 -> n/8x8x8x4
    - ... -> 1
  - Which one has a better node/data ratio?

Branch-on-Need (BON) Octrees
- A top-down implementation of bottom-up octrees
- Overlay the volume with a conceptual all power of 2 volume
- Subdivide the volume based on the conceptual volume

BONO (BON-Octrees)

BONO Construction
**BONO v.s top-down**

- Put more 8-way branches toward to the leaf nodes
- In general, BONO requires a smaller number of tree nodes

**Octrees and visualization**

- Isosurface Extraction
- Each BONO node stores the min/max values of the corresponding subvolume
- The empty node can be skipped rapidly

**Octrees for Volume Rendering**

- Each tree node stores the mean value of the encompassed voxels
- Alternatively, each tree node stores a subvolume of reduced resolution
- Each tree node also stores an error measure: e.g. standard deviation of the voxels (root mean square to the approximated value)

**Octrees for Volume Rendering**

- At run time, the error measures stored in the octree nodes are compared with a threshold
- The traversal is stopped at the tree nodes that pass the threshold

**Example:**

![Example Image](image-url)