RL compression for volume data

Part I: Data structures

Michal Vanek

Goals

- Compress volume data using Run-length compression (RL)
- Visualize volume by raytracing
- GPU usage (OpenGL GLSL)

Input data

- F3d data
  - Additional info in header
- Needed density
- Optional
  - Color
  - Gradient (normal)
- Synthetic type of data (voxelized objects)
  - Suitable for compression

RL volume compression algorithm I

- Retrieve Z slice from volume (performance issue)
  - Density band
  - Compress each X row
  - Divide RL runs into values and indexes
- Compress other bands according to compressed density band
- Results
  - Index/count array
  - Density array
  - Color and gradient arrays

RL volume compression algorithm II

- Problem
  - Random access incorporates iteration and decompression through whole RL volume
- We need to skip values
- Solution
  - Auxiliary 2D array – index row array

RL volume compression algorithm III

- Auxiliary 2D array
  - Stores indexes, that represents start of X row in previous arrays
  - ZY size

RL runs

10-0				
4-0	1-0.3	1-0.7	4-0	
2-0	1-0.3	4-1	1-0.7	2-0
4-0	1-0.3	1-0.7	4-0	
10-0				

Indexes

1
2,3,4,5
6,7,8,9,10
11,12,13,14
15

Index row array

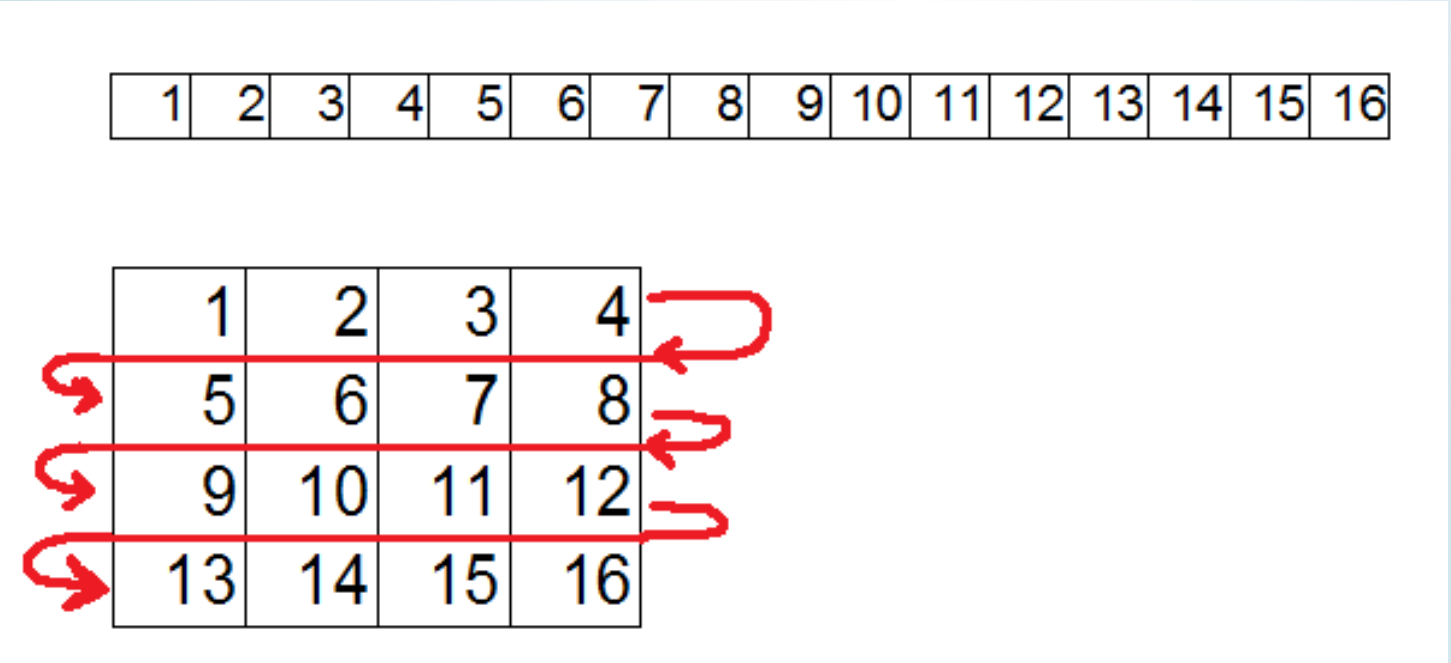
1
2
6
11
15

RL values textures encoding I

- Texture is the only enough big buffer accessible from GLSL shader
- Max. size of textures – aprox. 16000 px. for RGBA tex, total more than 65000 values
  - Typical RL volume has more than 10 times more values
- Solution
  - Code RL values into 2D textures + rgba components
  - 8096x8096 texture * 4 components - more than 16 mil. values

RL values textures encoding II

- 1D array → 2D texture



Random access algorithm in shader

- We have x,y,z location of ray
- Access to voxel:
  - Get offset from index row texture with z & y value
  - Read compressed row from values texture
  - Decompress row and iterate through it
  - Get appropriate voxel
  - Use decompressed row texture coordinates to get color and gradient from other textures
- If gradient is not present, calculate it
  - Get 6 more voxels (4 rows)

Tests

- DodecaFloat.f3d
  - Only density
- Before compression
  - 256 x 256 x 256 volume
  - 16 777 216 voxels (64 MB)
- After
  - About 650 000 voxel values + 256 x 256 short int for index row texture (aprox. 3 MB)
- More than 20 times compression ratio

Problems &future impovements

- Non power of 2 textures
  - ARB_texture_non_power_of_two
- Shader performance
  - RL decompression
  - Too many texture fetches
- Recursive RL compression
  - Compress whole slice if it contains only one value
  - Join rows and compress them together

Master thesis progress

- ✔ Builder plugin for Visualization Engine
  - ✔ Algorithm for compressing volume data
  - ✔ RL Data structure suitable for GPU
- ✗ Renderer plugin
  - ✗ RL GPU row decompression
  - ✗ GPU raytracing algorithm

Thank you for your attention.