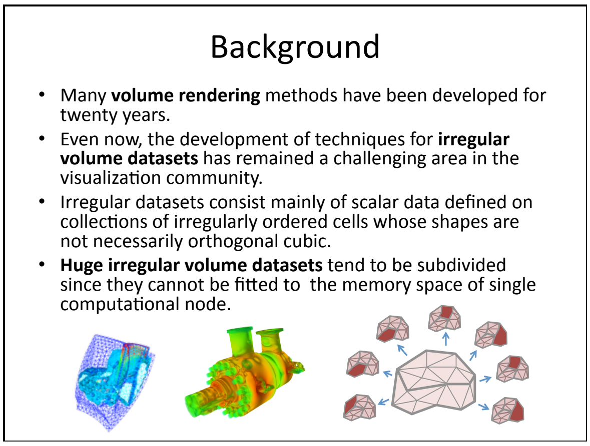
粒子ベースボリュームレンダリング
-口腔シミュレーション結果への適用-

坂本尚久（京大）　野崎一徳（阪大）
宮地英生（KGT）　江原康生　小山田耕二（京大）

第32回CG・可視化研究会（CAVE研究会）
共催：VR学会テレイマージョン技術研究会
大阪大学CMC-CAVE研究会

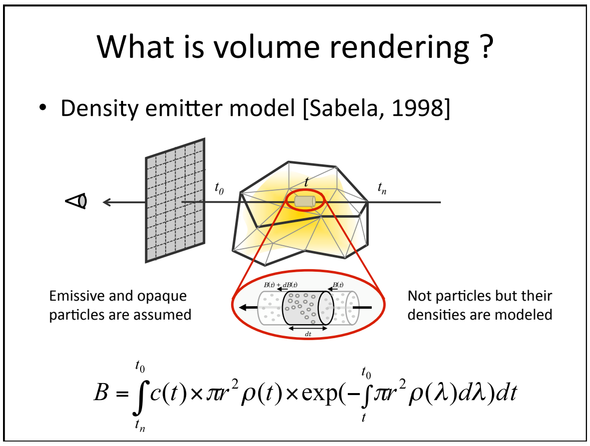
2008年7月3日（木）
海洋研究開発機構　横浜研究所

Contents



Background

- Many **volume rendering** methods have been developed for twenty years.
- Even now, the development of techniques for **irregular volume datasets** has remained a challenging area in the visualization community.
- Irregular datasets consist mainly of scalar data defined on collections of irregularly ordered cells whose shapes are not necessarily orthogonal cubic.
- **Huge irregular volume datasets** tend to be subdivided since they cannot be fitted to the memory space of single computational node.



What is volume rendering ?

- Density emitter model [Sabela, 1998]

Emissive and opaque particles are assumed

Not particles but their densities are modeled

$$B = \int_{t_n}^{t_0} c(t) \times \pi r^2 \rho(t) \times \exp(-\int_{t}^{t_0} \pi r^2 \rho(\lambda) d\lambda) dt$$

Contents

Related Works

- Performance improvement of visibility sorting
    - HAVS [Steven et. al., 2005]
        - 1.3 fps for 1.4M tets.
    - Point-based technique [Erik.W et. al., 2007]
        - 0.3 fps for 6.3M tets.
- Technique without visibility sorting
    - Only absorption [Cselfalvi et. al., 2003]
    - Only emission [Stefan et. al., 2003]

Development of a technique **without visibility sorting** which considers both **absorption and emission** is required

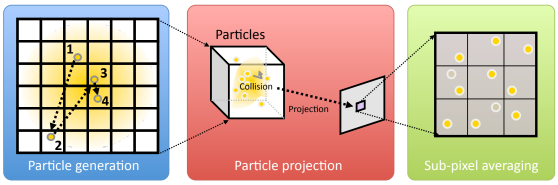
Contents

1. Background
2. Related Works
3. Particle-based Volume Rendering
4. Application to Huge Irregular Volumes
   1. Distributed Particle-based Volume Rendering
   2. CFD-based Prediction of Dental Fricative Sound
5. Conclusions

Particle-based Volume Rendering

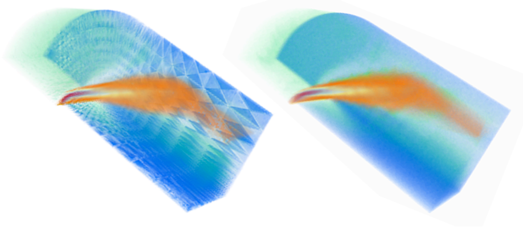
[Sakamoto et. al., 2007]

- Regression to the density emitter model
  - Generate particles using Metropolis method
  - Project particles onto image plane
  - Average sub-pixel values
- Simple and efficient
  - No visibility sorting
  - No alpha blending



Comparison with HAVS (1/2)

- Image quality

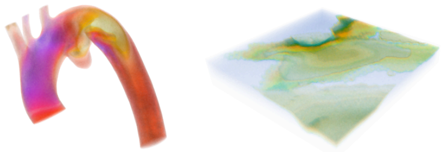


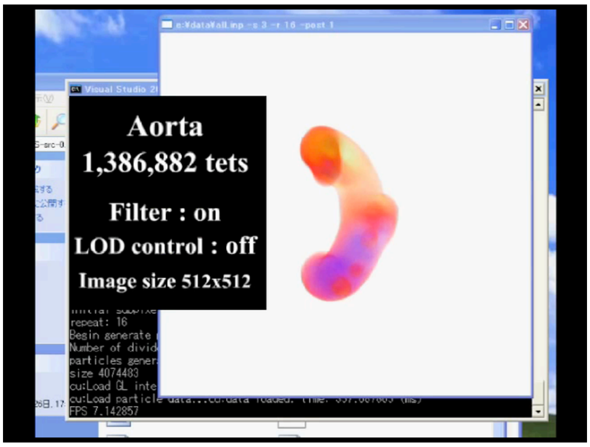
(a) HAVS (b) PBVR

Comparison with HAVS (2/2)

- Rendering performance

Dataset		GPU-based PBVR				HAVS	
	# of tets.	l_s, l_r	generate [msec]	data load [msec]	fps	pre-build [msec]	fps
Aorta	1.39 M	3,12	3272	279	9.85	8172	2.67
		3,16	3538	360	7.64		
Sf2	2.07 M	3,12	5192	517	6.92	11234	2.2
		3,16	5638	665	5.14		





Aorta
1,386,882 tets

Filter : on
LOD control : off
Image size 512x512

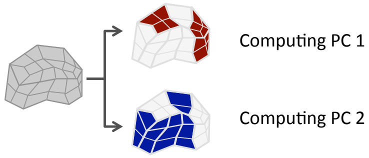
```
repeat: 16
Begin generate
Number of divid
particles gener
size 4074483
cu:Load GL inte
cu:Load particle
FPS 7.142857
```

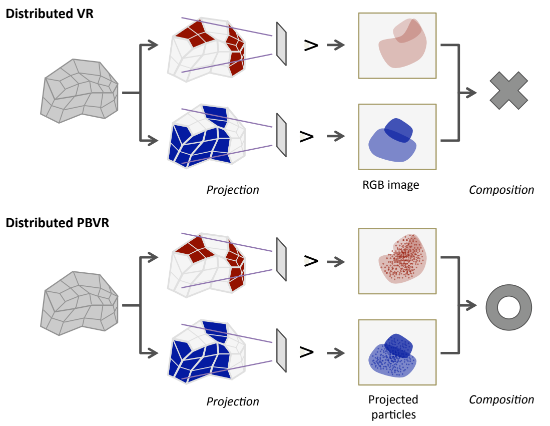
Contents

1. Background
2. Related Works
3. Particle-based Volume Rendering
4. Application to Huge Irregular Volumes
   1. Distributed Particle-based Volume Rendering
   2. CFD-based Prediction of Dental Fricative Sound
5. Conclusions

Subdivided Irregular Volumes

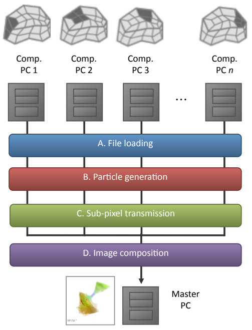
- Multiple irregular volumes are often generated by Large-scale FEM simulation
- The subdivision does not have relation with the visibility sorting





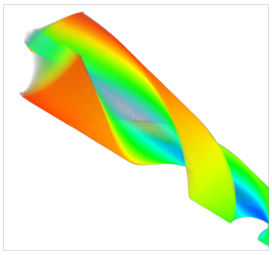
Distributed PBVR

- Computational Environment
  - PC cluster system consists of multiple computing PCs and a single master PC.
    - Master PC
      CPU: Intel Pentium 4 2.4GHz
      RAM: 2.0GB
    - Computing PC x 4
      CPU: Intel Core 2 Duo 1.86 GHz
      RAM: 1.0GB
  - The proposed system is composed of four stages:
    - A. File loading
    - B. Particle generation
    - C. Sub-pixel transmission
    - D. Image composition



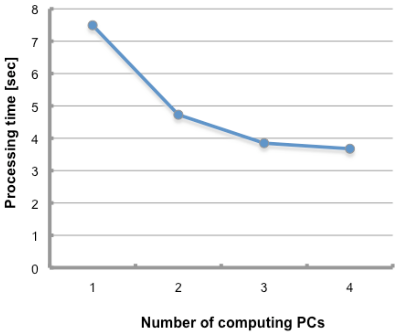
Performance Evaluation

- Drill data
  - Generated by a FEM solver FrontSTR
  - Composed of about **10M tetrahedral cells**
  - File size is **177MB**
  - Image resolution is **512x512**

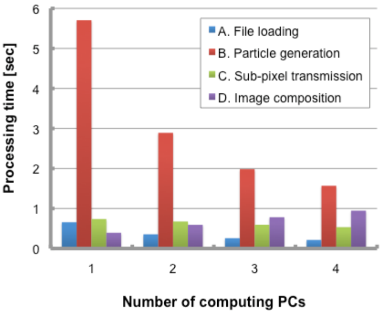


Data courtesy : U of Tokyo, Prof. Okuda

Total Processing Time



Processing Time of Each Stage



Performance Model

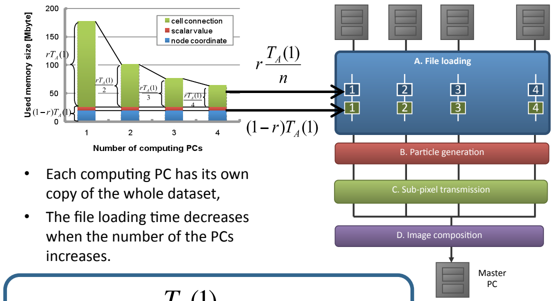
- We construct a performance model from the experimental results of the drill dataset.

$$T(n) = T_A(n) + T_B(n) + T_C(n) + T_D(n)$$

n: number of computing PCs

$T_A(n)$: loading time
$T_B(n)$: projection time
$T_C(n)$: transmission time
$T_D(n)$: composition time

File Loading Time: T_A

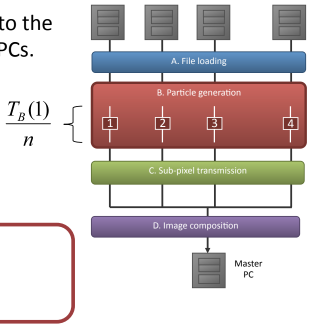


- Each computing PC has its own copy of the whole dataset,
- The file loading time decreases when the number of the PCs increases.

$$T_A(n) \cong r\frac{T_A(1)}{n} + (1-r)T_A(1)$$

Projection Time: T_B

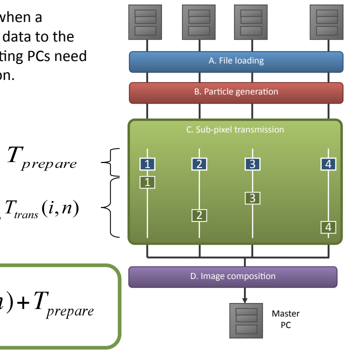
- The processing time is inversely proportional to the number of computing PCs.



$$T_B(n) \cong \frac{T_B(1)}{n}$$

Transmission Time: T_C

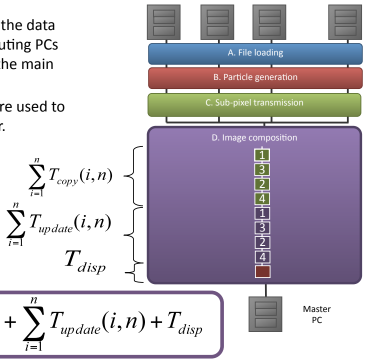
- In our implementation, when a computing PC is sending data to the master PC, other computing PCs need to wait for the completion.



$$T_C(n) = \sum_{i=1}^{n} T_{trans}(i,n) + T_{prepare}$$

Composition Time: T_D

- The master PC receives the data transmitted from computing PCs and makes the copy in the main memory.
- The copied fragments are used to update the frame buffer.



$$T_D(n) = \sum_{i=1}^{n} T_{copy}(i,n) + \sum_{i=1}^{n} T_{update}(i,n) + T_{disp}$$

Performance Construction

- The performance model is represented as:

$$\begin{aligned} T(n) &= T_A + T_B + T_C + T_D \\ &= \frac{rT_A(1) + T_B(1)}{n} + \sum_{i=1}^{n}(T_{trans}(i,n) + T_{copy}(i,n) + T_{update}(i,n)) + C \end{aligned}$$

where, $C = (1-r)T_A(1) + T_{prepare} + T_{disp}$

Identification of Model Params.

- From the experimental result conducted by using a single computing PC
  - $r = 0.85$
  - $T_A(1) = 0.657$ [sec], $T_B(1) = 5.705$ [sec]
- Well-balanced processing time at each computing PC:

$$\begin{aligned}\sum_{i=1}^{n} T_{trans}(i,n) &\cong nT_{trans}(1,n)\\ \sum_{i=1}^{n} T_{copy}(i,n) &\cong nT_{copy}(1,n)\\ \sum_{i=1}^{n} T_{update}(i,n) &\cong nT_{update}(1,n)\end{aligned}$$

Simple Regression Fomulas

- Transmission/Copying/Updating time

Num. of Comp. PCs	1	2	3	4	Ave.
T_{trans} [sec]	0.303	0.267	0.205	0.169	0.236

Num. of Comp. PCs	1	2	3	4	Ave.
T_{copy} [sec]	0.173	0.156	0.156	0.149	0.159

Num. of Comp. PCs	1	2	3	4	Ave.
T_{update} [sec]	0.060	0.061	0.054	0.048	0.055

$$\Rightarrow\quad\begin{aligned}T_{trans}(1,n) &= 0.15082941 + \frac{0.163317292}{n}\\ T_{copy}(1,n) &= 0.143270077 + \frac{0.029487692}{n}\\ T_{update}(1,n) &= 0.049045154 + \frac{0.012297785}{n}\end{aligned}$$

Performance Estimation

$$\begin{aligned}T(n) &= \frac{0.85\cdot 0.657 + 5.705}{n} + n\left(\sum_{i=1}^{n}(T_{trans}(i,n) + T_{copy}(i,n) + T_{update}(i,n))\right)\\ &= \frac{6.26}{n} + 0.3431n + C\end{aligned}$$

$$\frac{\partial T}{\partial n} = -\frac{6.26}{n^2} + 0.3431 < 0$$

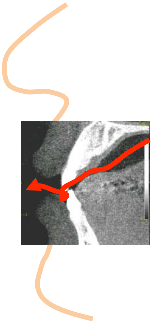
- The max. of integer of n becomes 4.
- The total performance time decreases when the number of computing PCs exceeds 5.

Contents

1. Background
2. Related Works
3. Particle-based Volume Rendering
4. Application to Huge Irregular Volumes
   1. Distributed Particle-based Volume Rendering
   2. CFD-based Prediction of Dental Fricative Sound
5. Conclusions

CFD-based Prediction of Dental Fricative Sound

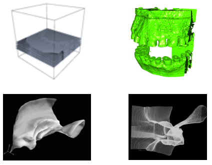
- The sound source of the dental fricative seems to be downstream obstacle in the oral airflow field.
- In the development of the fundamental treatment methods of speech disorders, it is crucial to explore the sound source of the dental fricative using CFD.
- The mesh size tends to become huge since broadband noise should be calculated in high frequency.

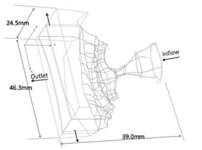


Data courtesy: Cybermedia Center, Osaka University

Mesh Generation

- STL from 512 slices of CT image with 512x512 resolution using Marching Cube
- NURBS from STL
- 2.64M hexahedral cells from NURBS
- 72M hexahedral cells by subdivision

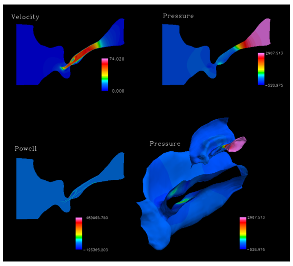




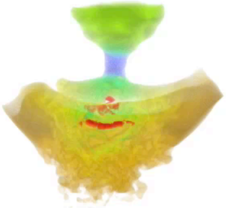
Data courtesy: Cybermedia Center, Osaka University

Visualization

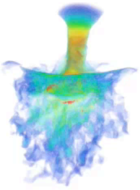
- Previous visualization is surface-based for 2.64M hexahedral cell irregular volume.
- To store 72M hexahedral cell volume dataset, a 3.5GB memory space is required



Pressure



Velocity



Contents

1. Background
2. Related Works
3. Particle-based Volume Rendering
4. Application to Huge Irregular Volumes
    1. Distributed Particle-based Volume Rendering
    2. CFD-based Prediction of Dental Fricative Sound
5. Conclusions

Conclusions

- A distributed implementation of PBVR for handling a large scale irregular volume dataset is proposed.
    - The effectiveness of our system is confirmed by applying the distributed PBVR to huge irregular volumes; the drill dataset (10M tet. cells) and the mouth dataset (71M hex. cells).
- The performance model is constructed from the drill dataset.
    - The enough parallelization effect has been achieved at stages of the file loading and the particle projection.
    - A distributed processing technique for the sub-pixel transmission and the image composition stage needs to be developed in order to accelerate the rendering performance.