GI-Cube: An Architecture for Volumetric Global Illumination and Rendering

Frank Dachille IX and Arie Kaufman

Center for Visual Computing (CVC) and Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794-4400, USA
Outline

- Introduction
  - Related Work
  - Algorithms
  - Hardware
  - Simulation
  - Results & Discussion
  - Conclusion
Introduction

- Volume rendering hardware
- Parallel ray tracing
- Parallel radiosity
- Scalable
- Static object-based partition
- Ray reordering
- Flexible ray processing
Introduction

• Single PCI board
  – DSP interface
  – Single-chip pipelined ASIC
  – Rambus DRAM
  – eDRAM

• Global illumination
  – Lifelike rendering improves understanding
  – Reflections, shadows, indirect illumination
    • Clarify spatial relationships
Outline

- Introduction
- Related Work
- Algorithms
- Hardware
- Simulation
- Results & Discussion
- Conclusion
Related Work

- **Volume rendering hardware**
  - General purpose [Cabral et al. 94]
  - Special purpose [de Boer et al. 96]
  - [Meissner et al. 98] [Pfister Kaufman 96]
  - Multithreading [Vetterman et al. 99]
  - Commercially available [Pfister et al. 99]

- **Global illumination**
  - Perception [Levoy et al. 90]
  - Coherence, reordering [Pharr et al 97]
Related Work

- **Global illumination**
  - Volumetric shadowing
    - [Kajiya von Herzen 84] [Meinzer et al. 91]
    - [Sobierański Kaufman 94]
    - [Behrens Ratering 98]
  - Transport theory
    - [Krueger 91]
  - Participating media
    - [Pattanaik Mudur 93] [Sillion Puech 94]
    - [Perez et al. 97]
Outline

- Introduction
- Related Work
  - Algorithms
- Hardware
- Simulation
- Results & Discussion
- Conclusion
Algorithms

- A cache–conscious volume ray tracing coprocessor
- Accelerates
  - Direct volume rendering
  - Globally illuminated volume rendering
  - Generalized ray tracing, e.g.,
    - Hyper–texture
    - Photon maps
    - Tomographic reconstruction
    - BSDF evaluation
## Algorithms

- **Ray packet – 256 bits**

<table>
<thead>
<tr>
<th>bits</th>
<th>Position X</th>
<th>Position Y</th>
<th>Position Z</th>
<th>Direction X</th>
<th>Direction Y</th>
<th>Direction Z</th>
<th>Destination U</th>
<th>Destination V</th>
<th>Contribution</th>
<th>Opacity</th>
<th>Interaction</th>
<th>Generation</th>
<th>Red</th>
<th>Type</th>
<th>Green</th>
<th>Blue</th>
<th>User</th>
</tr>
</thead>
<tbody>
<tr>
<td>256</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>196</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 0 | 4 | 8 | 12 | 16 | 20 | 24 | 28 | 32 |
Algorithms

• **Base configuration**
  – $256^3$ volume, $32^3$ blocks, 4 processors

• **Hybrid image – object order**
  – Rays queued on blocks
    – Blocks processed sequentially on processor
    – Rays passed between blocks

• **Processor has multiple queues**
  – Processes most important queue first
  – Processes round–robin within queue
Algorithms

- Processor pipeline

![Diagram showing processing pipeline with components such as Prefetch and Memory Controller, Interleaved Volume Cache, Resampling, Segmentation, Shading and Compositing, Scattering and Splatting, Queue Sorter, Ray Queues, and Pixel Bus.]
Algorithms

- Parallel partitioning – load balance, communication, & coherence

Processor 0 1 2 3

Simple Slab  Repeating Slab  Block Skewed
Algorithms

• Global illumination
  – Two-pass, bidirectional
    • Light tracing
    • Ray tracing
  – Generalized rays
  – Splatting
Algorithms

- Global illumination modes
  - Low albedo
    - Few rays, little scattering
    - Good for shadowing
    - Fixed # rays sent to each edge voxel
  - High albedo
    - Many rays, much scattering
    - Good for clouds, radiosity
    - Large # rays sent randomly
Outline

• Introduction
• Related Work
• Algorithms
  ➢ Hardware
• Simulation
• Results & Discussion
• Conclusion
Hardware

- DSP
- SDRAM
- PCI Bus
- Ray Bus
- Block Processor
- RDRAM
- Block Processor
- RDRAM
- Block Processor
- RDRAM
- Block Processor
- RDRAM
Hardware

- Four components
  - DSP, Processors, Memory, and Pipelines
- DSP
  - Interface
  - Controller
  - Loads dataset
  - Generates rays
  - Assembles image
  - Relies on scratch SDRAM
Hardware

- **Processors**
  - Take one sample per cycle
    - Barring stalls
  - Maintain and sort queues
    - Pipelined insertion sorter
    - eDRAM for ray queues
      - High on–chip bandwidth
      - Regular access pattern

- **Communication**
  - DSP, Neighbors
Hardware

- Memory
  - Parallel, distributed, scalable
  - Static load balance
  - RDRAM for bandwidth
    - 1.6 GB/s available
    - 2.8 – 4.6 GB/s required
    - 18-bits at 800 MHz
    - 4 voxels per 100 MHz pipeline cycle
  - DDR alternative
Hardware

- Pipeline

- Interleaved Volume Cache
- Resampling
- Segmentation
- Prefetch and Memory Controller
- Spacing
- Shade
- Queue Sorter
- Scatter
- Splat
- Composite
Outline

- Introduction
- Related Work
- Algorithms
- Hardware
  - Simulation
- Results & Discussion
- Conclusion
Simulation

• Two simulators
  – Algorithmic
    • Multi – threaded
    • Flexible
    • Initial testbed
  – Hardware
    • Single – threaded
    • Bit and cycle accurate
    • Gathered exact statistics
Simulation

• Direct volume rendering, reflections
  – $300^2$ image, 44 Hz

• Caustics (GI)
  – $300^2$ image, 2 Hz
### Simulation

- **C\textsubscript{60} molecule**
  - Shadows
  - Reflections
  - Caustics
  - Lighting
    - 361 ms
  - Rendering
    - 39 Hz
Cache Effectiveness

- **Bigger blocks**
- **Better**
- **Compulsory misses**
- **Pipeline latency**
- **Design point**

![Graph showing the relationship between cache hit ratio and block size. The x-axis represents the ray queue size, ranging from 8 to 1024. The y-axis represents the cache hit ratio, ranging from 0% to 100%. Different line styles and colors indicate varying block sizes.](image)
Rendering Time

- Smaller blocks
  - Better
- $8 \times 8 \times 8$
- Pipeline latency
- Design point
## Reordering

- Evaluate reordering algorithm relative to cached volume rendering

<table>
<thead>
<tr>
<th>Measure</th>
<th>Cache</th>
<th>Reordering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor throughput</td>
<td>52%</td>
<td>93%</td>
</tr>
<tr>
<td>Memory bandwidth</td>
<td>1.9 GB/s</td>
<td>0.8 GB/s</td>
</tr>
<tr>
<td>Frame rate</td>
<td>9.2 Hz</td>
<td>16.3 Hz</td>
</tr>
</tbody>
</table>
Scalability

- Near-linear
- Ray bus contention
- Pixel : voxel

![Graph showing scalability with ideal and actual speedup vs. number of processors.

- Ideal line increases at a constant rate.
- Actual line follows a similar trend but at a slightly lower rate.

Speedup vs. Number of Processors graph with_xlabel="Ideal" and_ylabel="Actual"
Pixel – to – Voxel Ratio

### Number of Misses

- **0%**
- **20%**
- **40%**
- **60%**
- **80%**
- **100%**

### Sample spacing

- **Sample spacing 1.0 voxels**
- **Sample spacing 0.25 voxels**

### Number of Misses

- 0
- 1
- 2
- 3
- 4
- 5
- 6
- 7
- 8

Sample spacing 1.0 voxels is represented by brown bars, while sample spacing 0.25 voxels is represented by green bars.
**Load Balance**

- **Simple slab**: 91%
- **Repeated slab**: 92%
- **Block skewed**: 76%

- **Block skewed**
  - 19% longer
  - Greater communication costs
Algorithmic Enhancements

- Designed into hardware
  - Free
- Data dependent

<table>
<thead>
<tr>
<th>Improvement</th>
<th>Minimum</th>
<th>Average</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space leaping</td>
<td>0%</td>
<td>21%</td>
<td>34%</td>
</tr>
<tr>
<td>Early ray termination</td>
<td>0%</td>
<td>12%</td>
<td>24%</td>
</tr>
</tbody>
</table>
Outline

- Introduction
- Related Work
- Algorithms
- Hardware
- Simulation
- Results & Discussion

Conclusion
Conclusion

- Flexible, scalable volumetric ray tracing
- Accelerates direct volume rendering
- Supports global illumination
- Extensively simulated
- Reordering doubles frame rate
- Near–linear scalability
- Feasible implementation
Acknowledgments

- ONR grant N000149710402
- Justine Dachille
- Manjushree Nulkar
- Klaus Mueller
- Ingmar Bitter
- Kevin Kreeger