Parallel Rendering

Molnar, Cox, Ellsworth, and Fuchs. “A Sorting Classification of Parallel Rendering.” *IEEE Computer Graphics and Applications*. July, 1994.

Why Parallelism

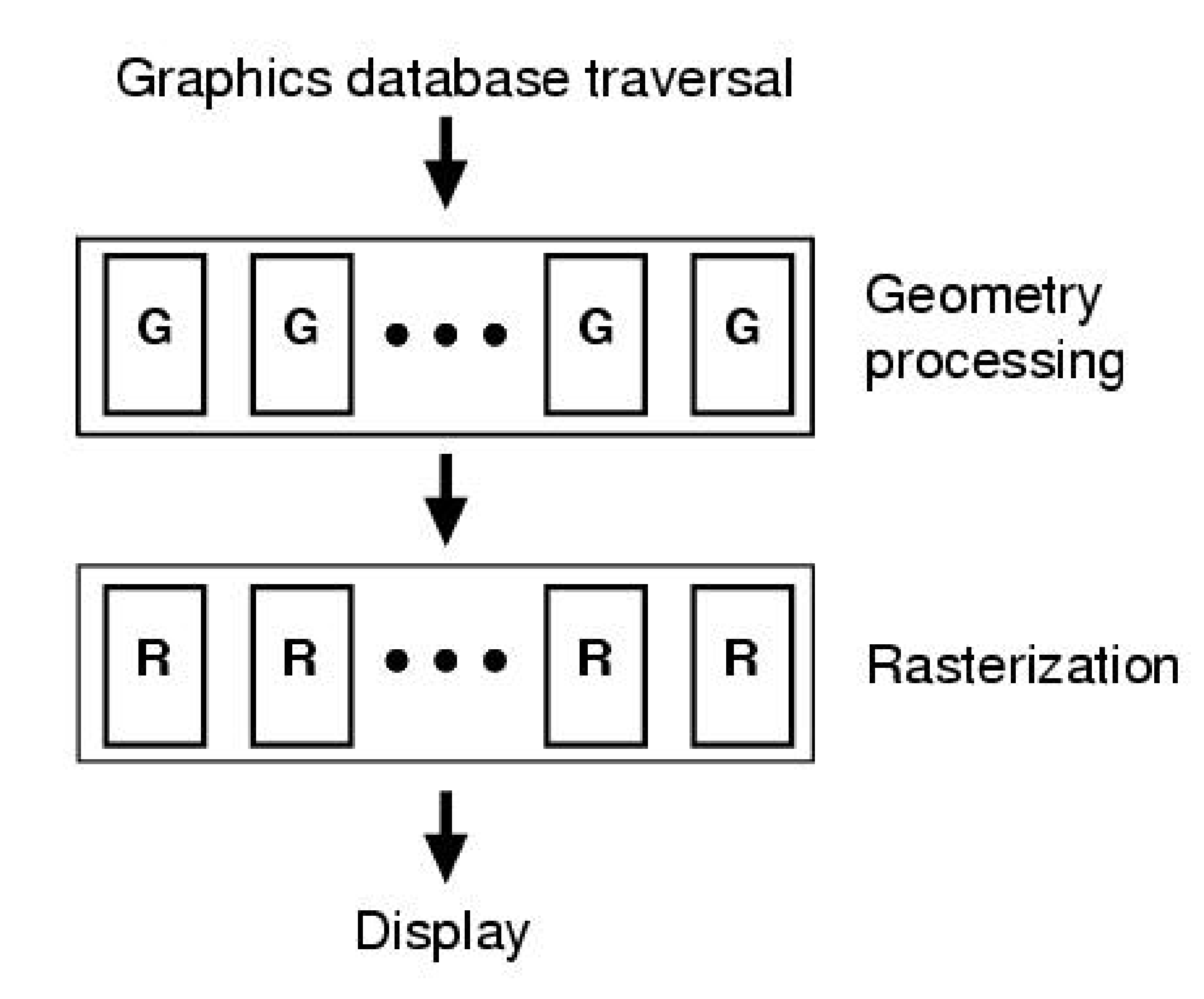
Applications need:

- High frame rates
- High resolution
- Large geometric models
- Stereo
- Antialiasing
- etc.

Performance implications:

- Hundreds of MFLOPS compute power
- Gigabytes per second memory bandwidth

Stages of Parallelism



Processing Tasks

Geometry Processors

- Each processor gets a subset of primitives
- Transformation
- (Lighting)
- Set-up for Rasterization

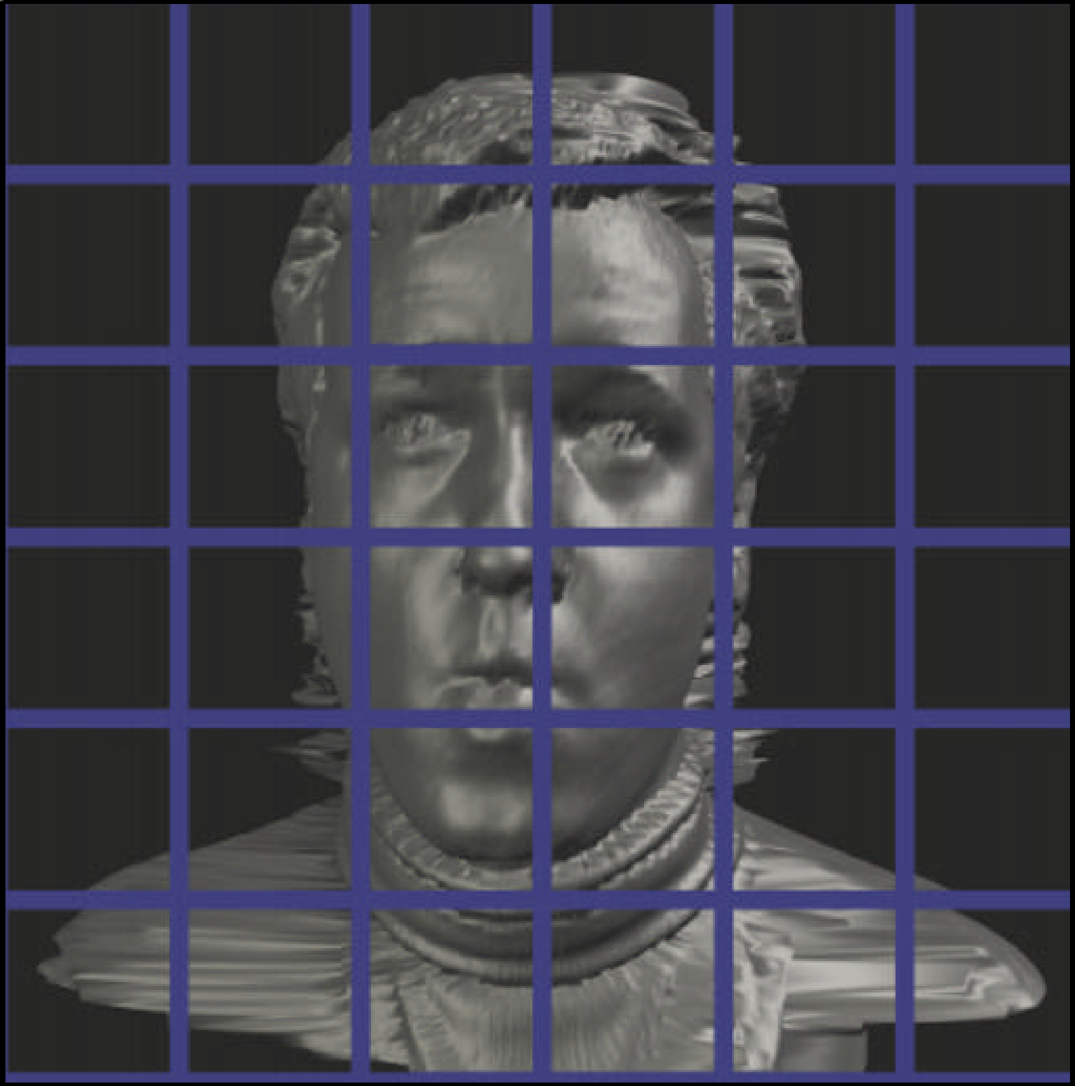
Rasterization Processors

- Each processor gets a subset of pixels
- Visibility computation
- Shading

Rendering as Sorting

- Primitives may lie anywhere on or off screen
- Determine effect of each primitive on each pixel
- Primitives are “sorted” onto screen
- Sorting affects distribution of data on geometry and rasterization processors

Primitives in Screen-space Regions



Where to sort

Sort Middle

- Sort between geometry processing and rasterization

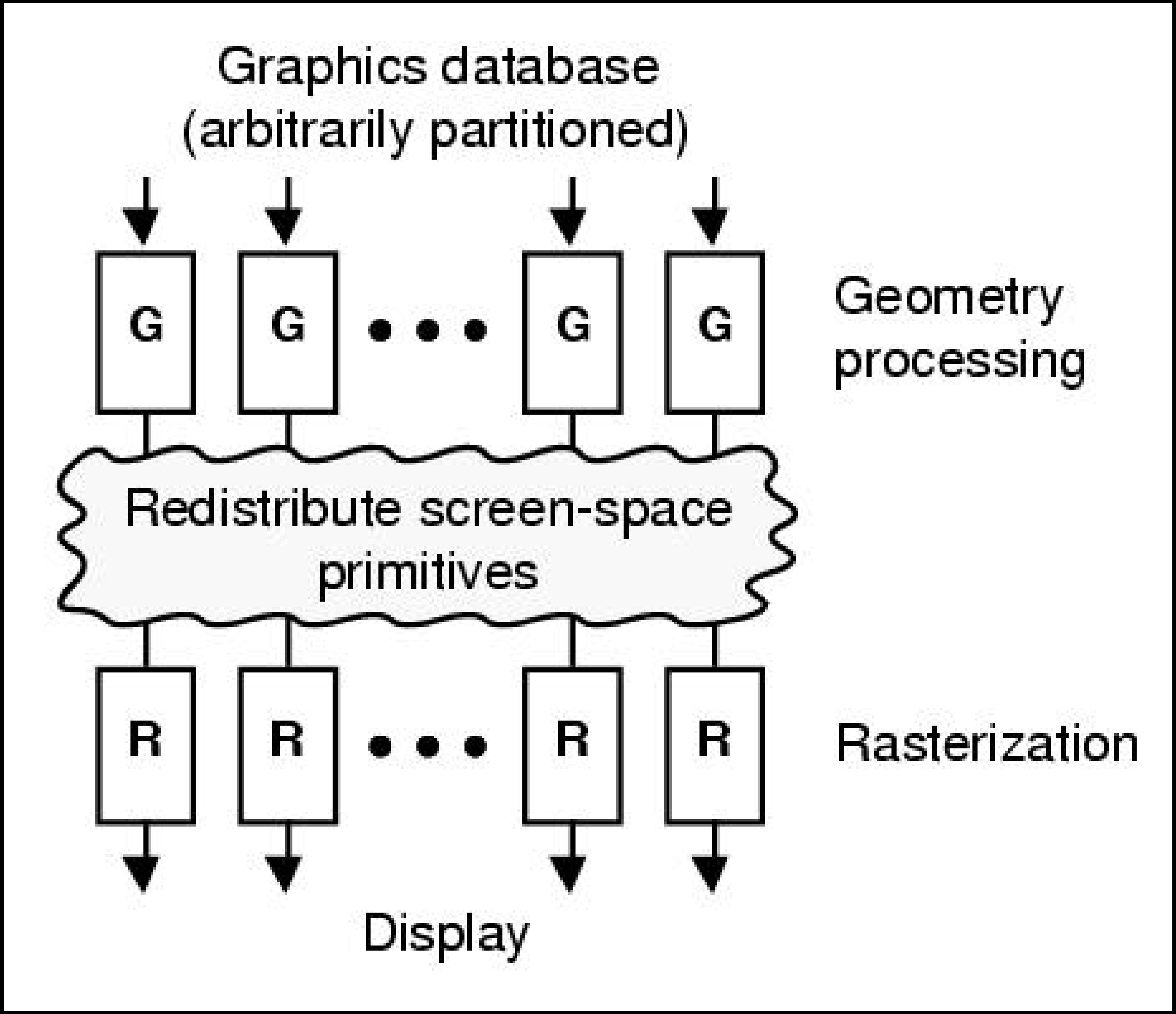
Sort First

- Sort during geometry processing

Sort Last

- Sort during rasterization

Sort Middle


Graphics database
(arbitrarily partitioned)
G
G
• • •
G
G
Geometry processing
Redistribute screen-space primitives
R
R
• • •
R
R
Rasterization
Display

Sort Middle: Data Arrangement

Geometry processors

- Arbitrary (random) distribution of primitives
- Good for load balancing

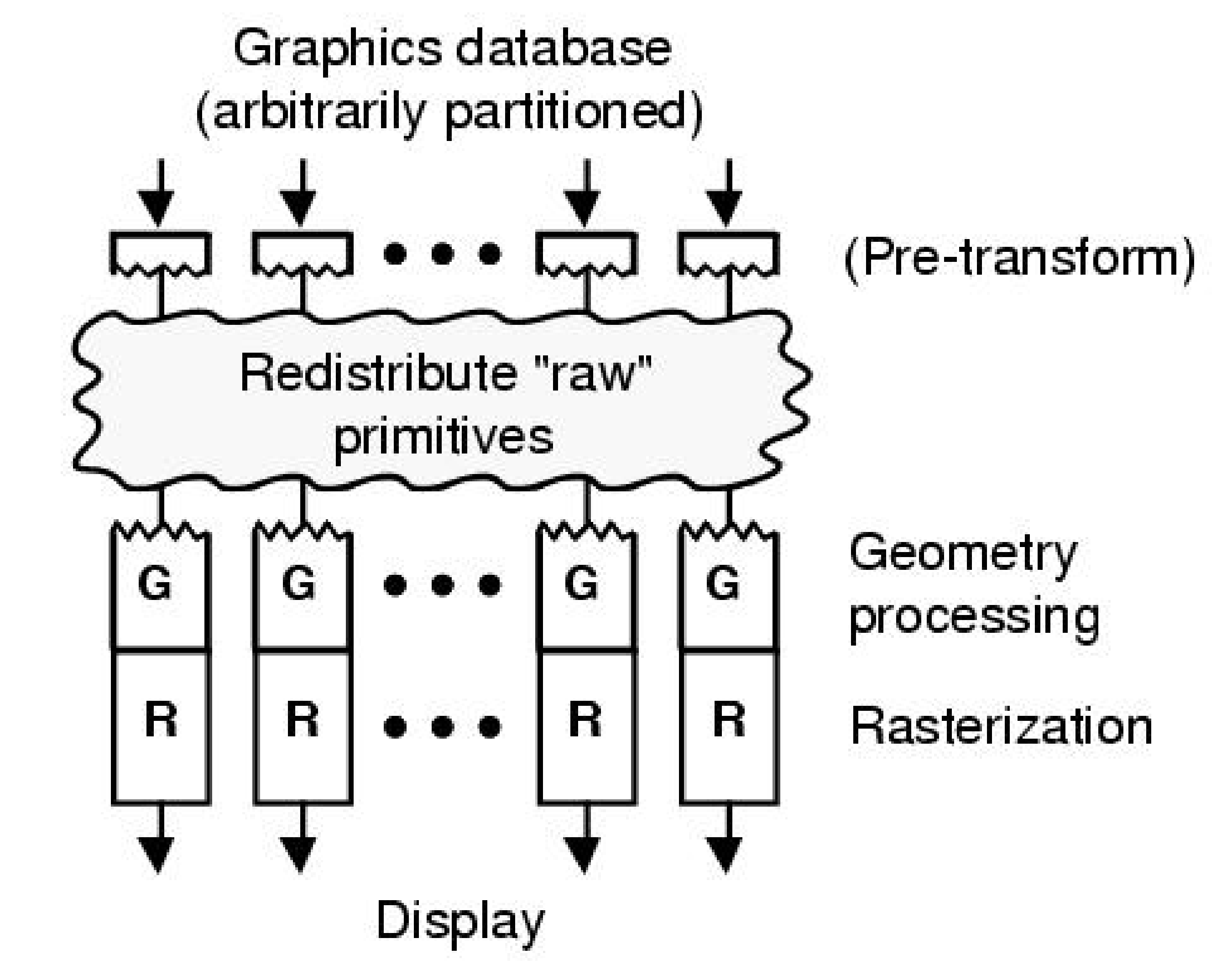
Rasterization processors

- Screen-space distribution of primitives
- Load balancing difficult

Sort Middle: Communications

- O(n^2) communications paths

Sort First


Graphics database
(arbitrarily partitioned)
(Pre-transform)
Redistribute "raw" primitives
G
G
G
G
Geometry processing
R
R
R
R
Rasterization
Display

Sort First: Data Arrangement

- Different screen regions of equal sizes may contain different numbers of primitives

  May need dynamic region sizes

Sort First: Communications

Must determine primitive screen coverage before full transformation

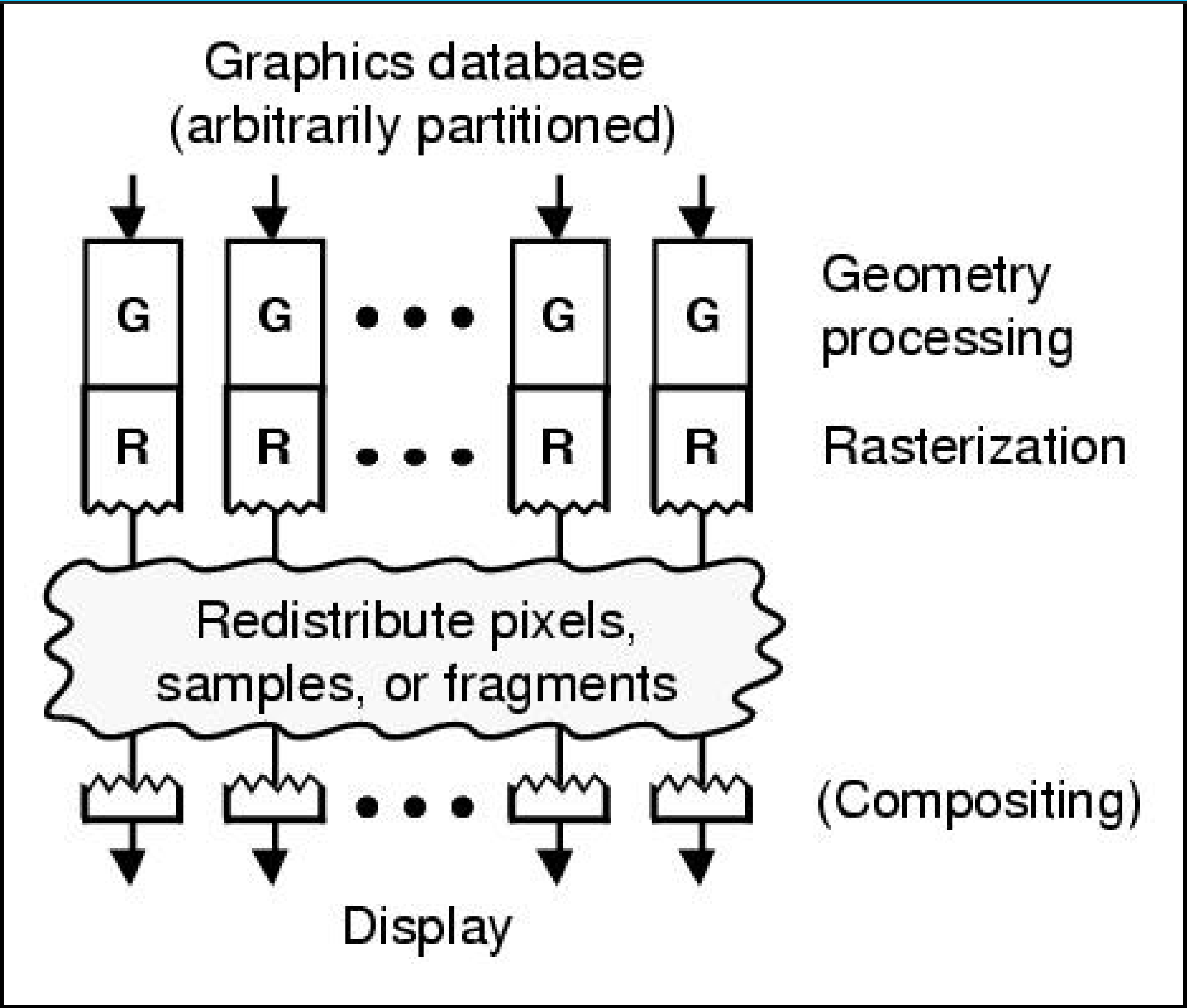
Exploit frame-to-frame coherence

-

Possibly employ primitive clustering and

- Pre-transform bounding volumes for small groups of primitives

Sort Last


Graphics database
(arbitrarily partitioned)
G
G
G
G
Geometry processing
R
R
R
R
Rasterization
Redistribute pixels, samples, or fragments
(Compositing)
Display

Sort Last: Data Arrangement

Arbitrary (random) arrangement of data on both geometry and rasterization processors

Great for load balancing

Each rasterization processor makes image of entire screen, with subset of primitives

Sort Last: Communications

Rasterization processors must communicate final pixel data

Composition of pixel data may take place along linear or tree-shaped network

Requires high bandwidth, assuming pixel data is much larger than primitive data

Advantages and Disadvantages

	Advantages	Disadvantages
SF	• Low communications when good coherence • Each processor implements entire pipeline	• Susceptible to load imbalance • Retained mode and complex data handling
SM	• General and straightforward • Natural communications placement	• High communication cost • Rasterizer load imbalance
SL	• Each processor implements entire pipeline • Easier load balancing • Linear scalability	• Large communication cost, especially for high resolution or multisampling

Video

Mueller. "Hierarchical Graphics Databases in Sort-First." ***Proceedings of 1997 Parallel Rendering Symposium*****.**