Multi-Resolution Method for Ray Tracing

Sung-Eui Yoon
(윤성의)

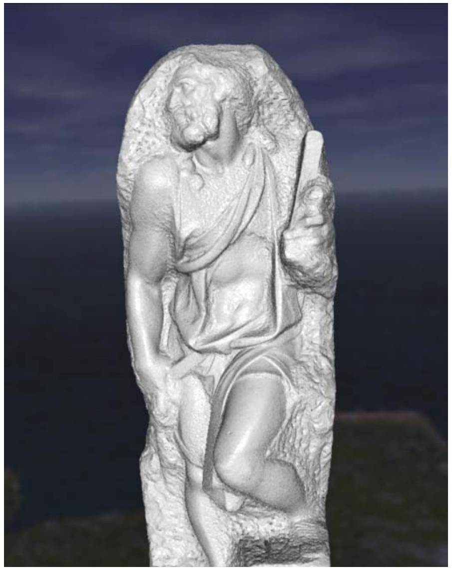
Course URL:
http://jupiter.kaist.ac.kr/~sungeui/SGA/

At the Previous Class

- **Studied LOD techniques for rasterization**

Goal

- **Perform an interactive ray tracing of massive models**
  - **Handles various kinds of polygonal meshes (e.g., scanned data and CAD)**



St. Matthew
372M triangles

Double eagle tanker
82M triangles

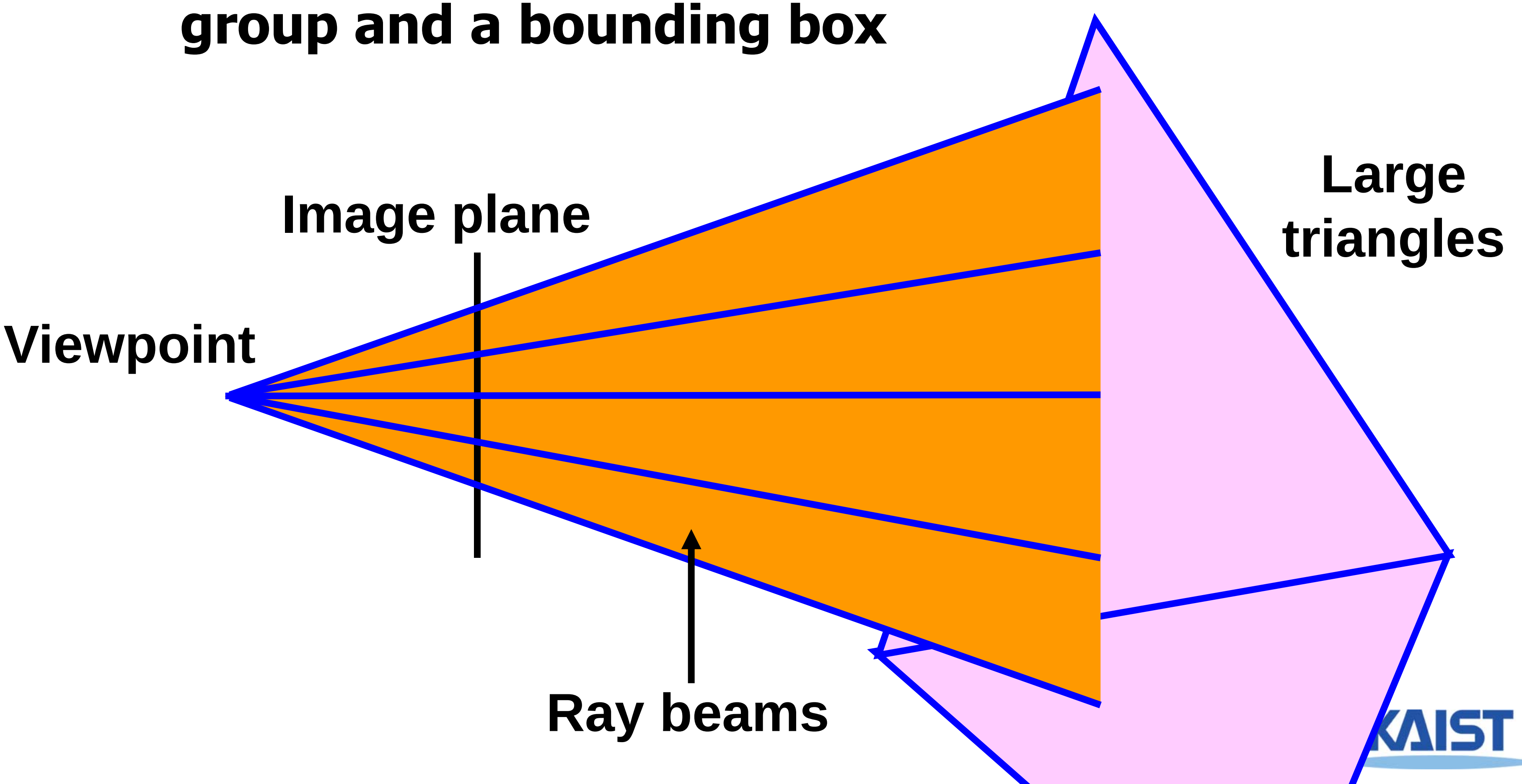
Forest model (32M)

Recent Advances for Interactive Ray Tracing

- **Hardware improvements**
  - **Exponential growth of computation power**
  - **Multi-core architectures**
- **Algorithmic improvements**
  - **Efficient ray coherence techniques [Wald et al. 01, Reshetov et al. 05]**

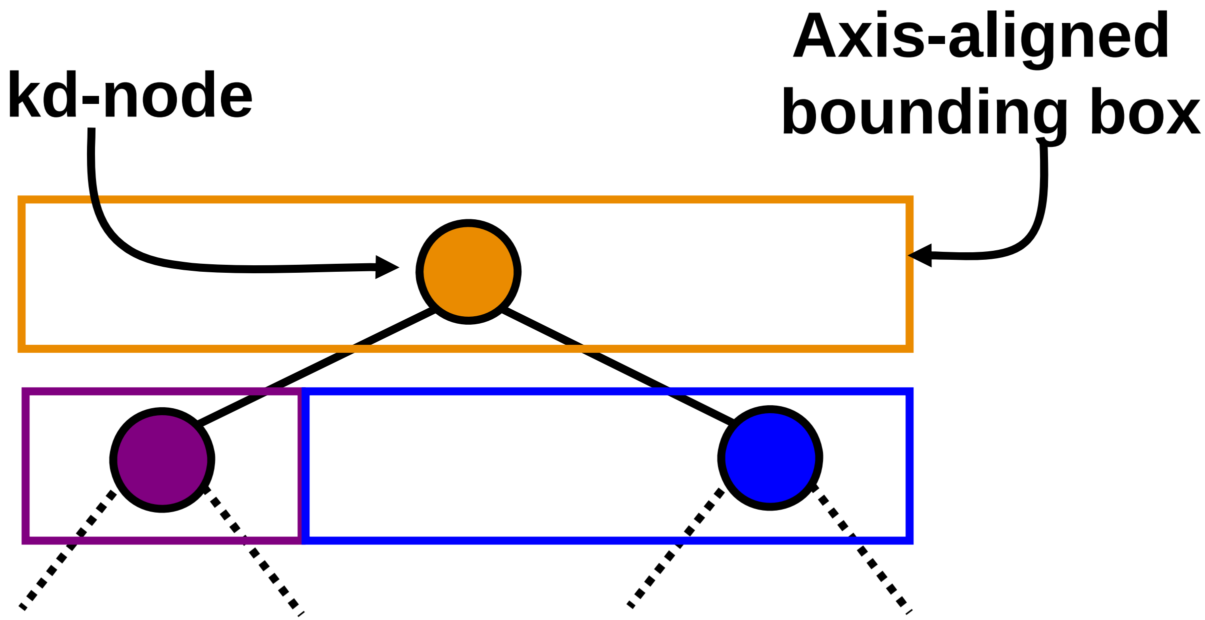
Ray Coherence Techniques

- **Models with large primitives**
  - **Group rays and test intersections between the group and a bounding box**



Hierarchical Acceleration Data Structures

- kd-trees for interactive ray tracing [Wald 04]
  - Simplicity and efficiency
  - Used for efficient object culling

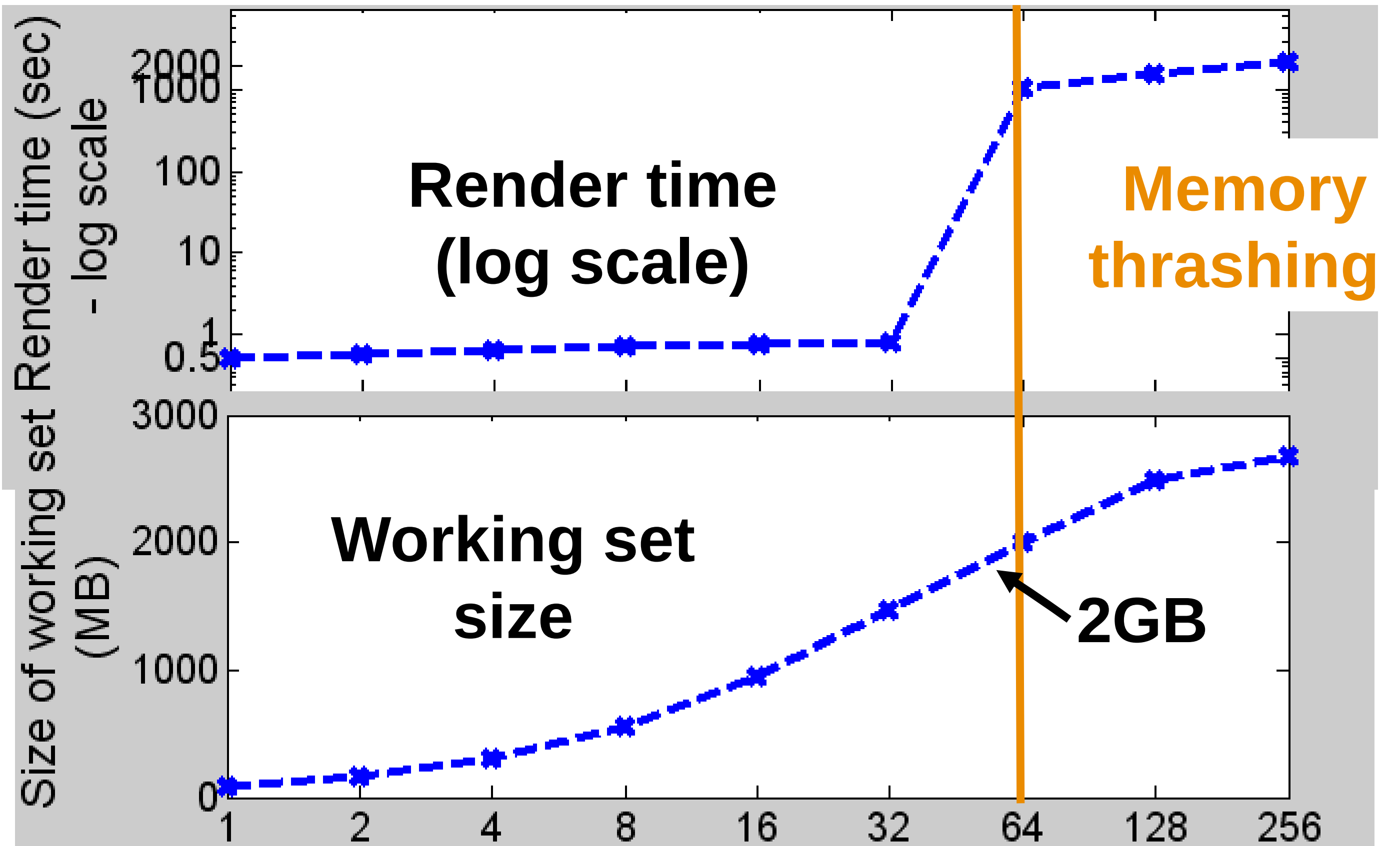


Ray Tracing of Massive Models

- **Logarithmic asymptotic behavior**
  - **Very useful for dealing with massive models**
  - **Due to the hierarchical data structures**
  - **Observed only in in-core cases**

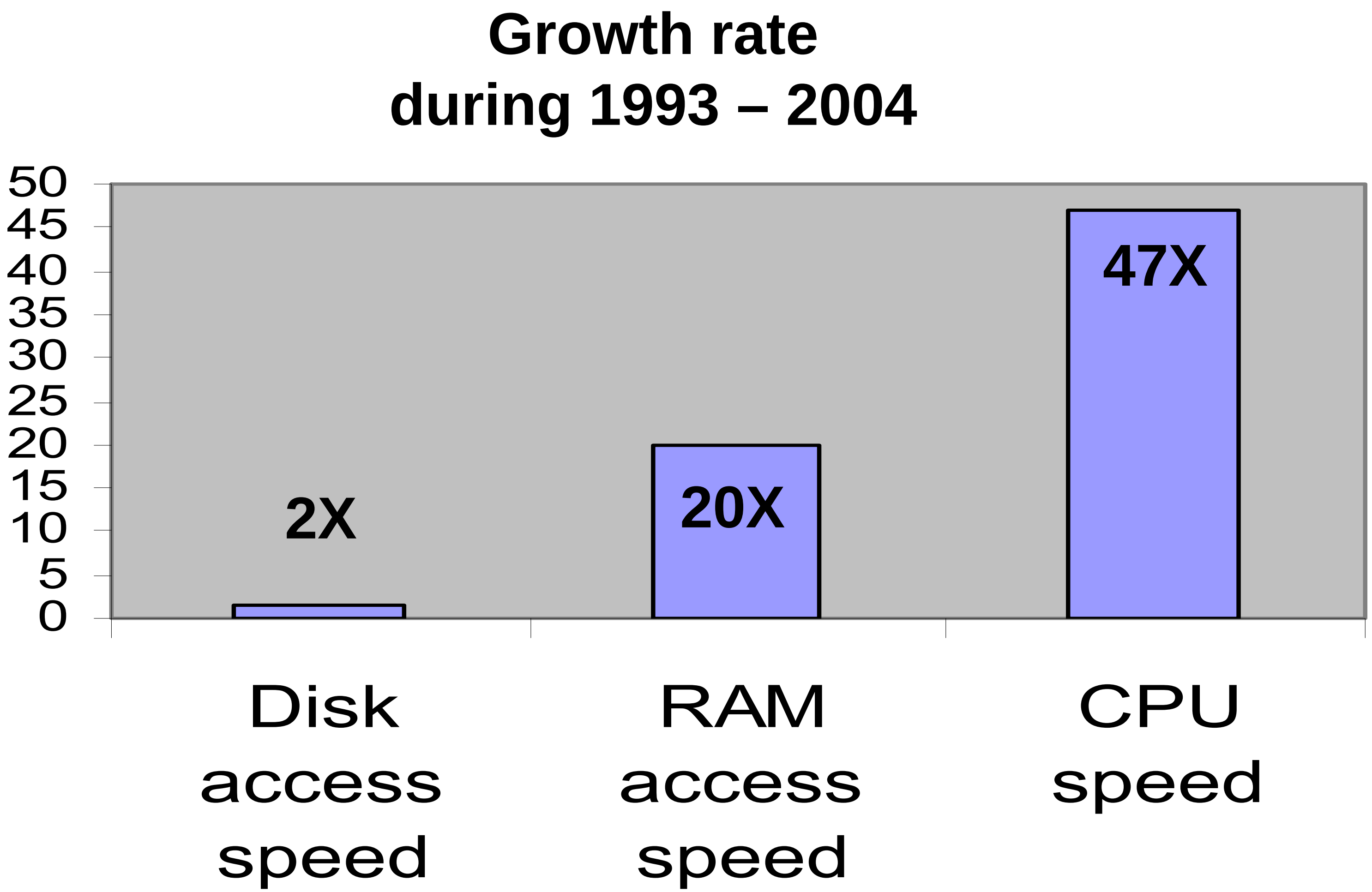
Performance of Ray Tracing with Different Model Complexity

- **Measured with 2GB main memory**



Model complexity (M tri) - log scale

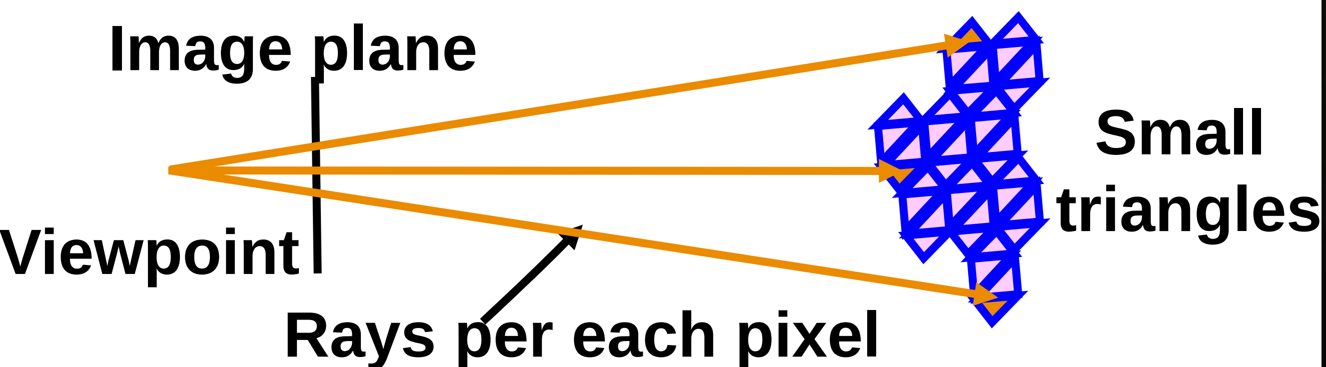
Low Growth Rate of Data Access Time



Courtesy: http://www.hcibook.com/e3/online/moores-law/

Inefficient Memory Accesses and Temporal Aliasing

- **St. Matthew (256M triangles)**
  - **Around 100M visible triangles**
- **1K by 1K image resolution**
  - **1M primary rays**
  - **Hundreds of triangle per pixel**
  - **Each triangle likely in different area of memory**



Example of Aliasing

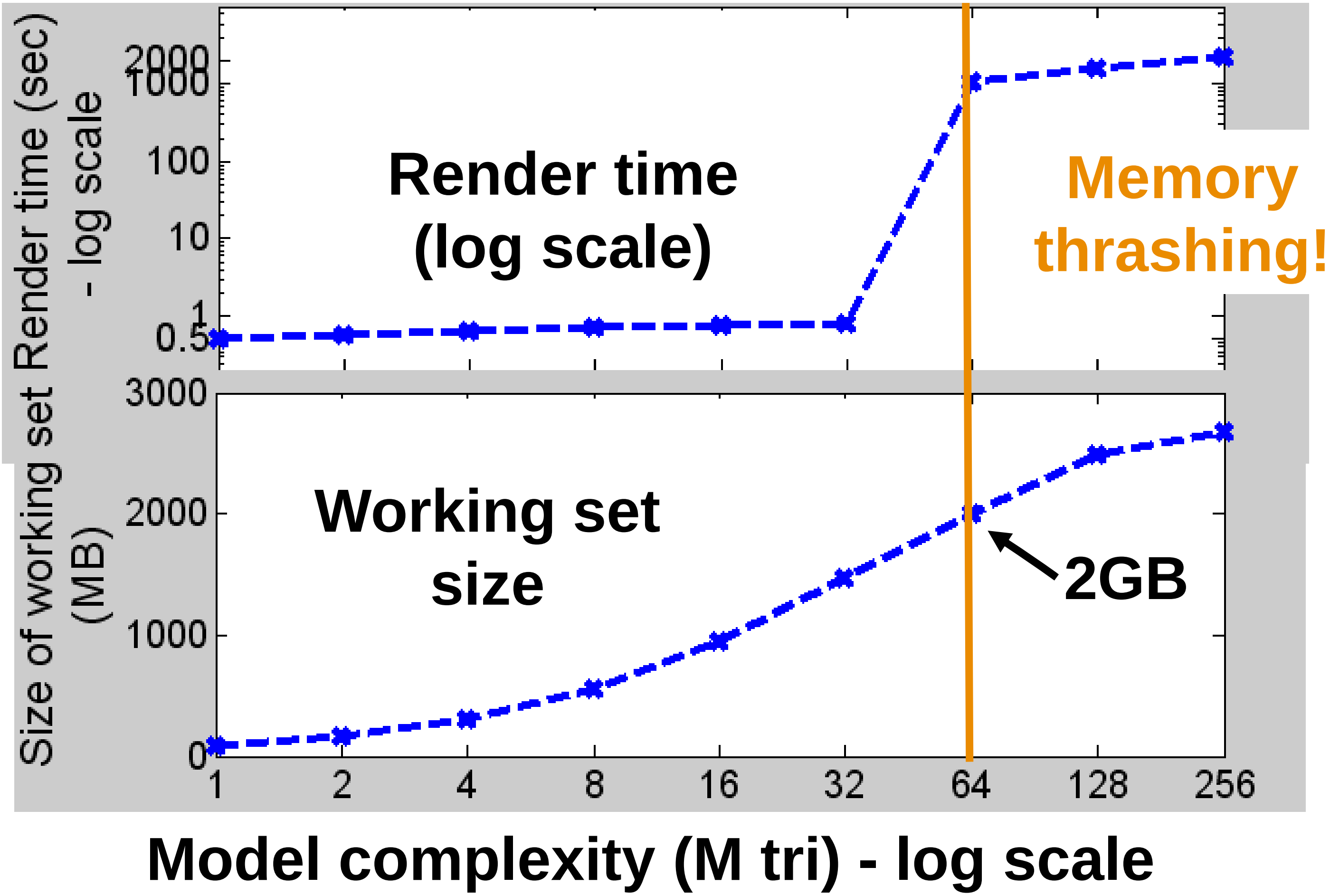
Due to the under-sampling

LOD-based Ray Tracing

- **Propose an LOD (level-of-detail)-based ray tracing of massive models**
  - **R-LOD, a novel LOD representation for Ray tracing**
  - **Efficient LOD error metric for primary and secondary rays**
  - **Integrate ray and cache coherent techniques**

Performance of Ray Tracing with Different Model Complexity

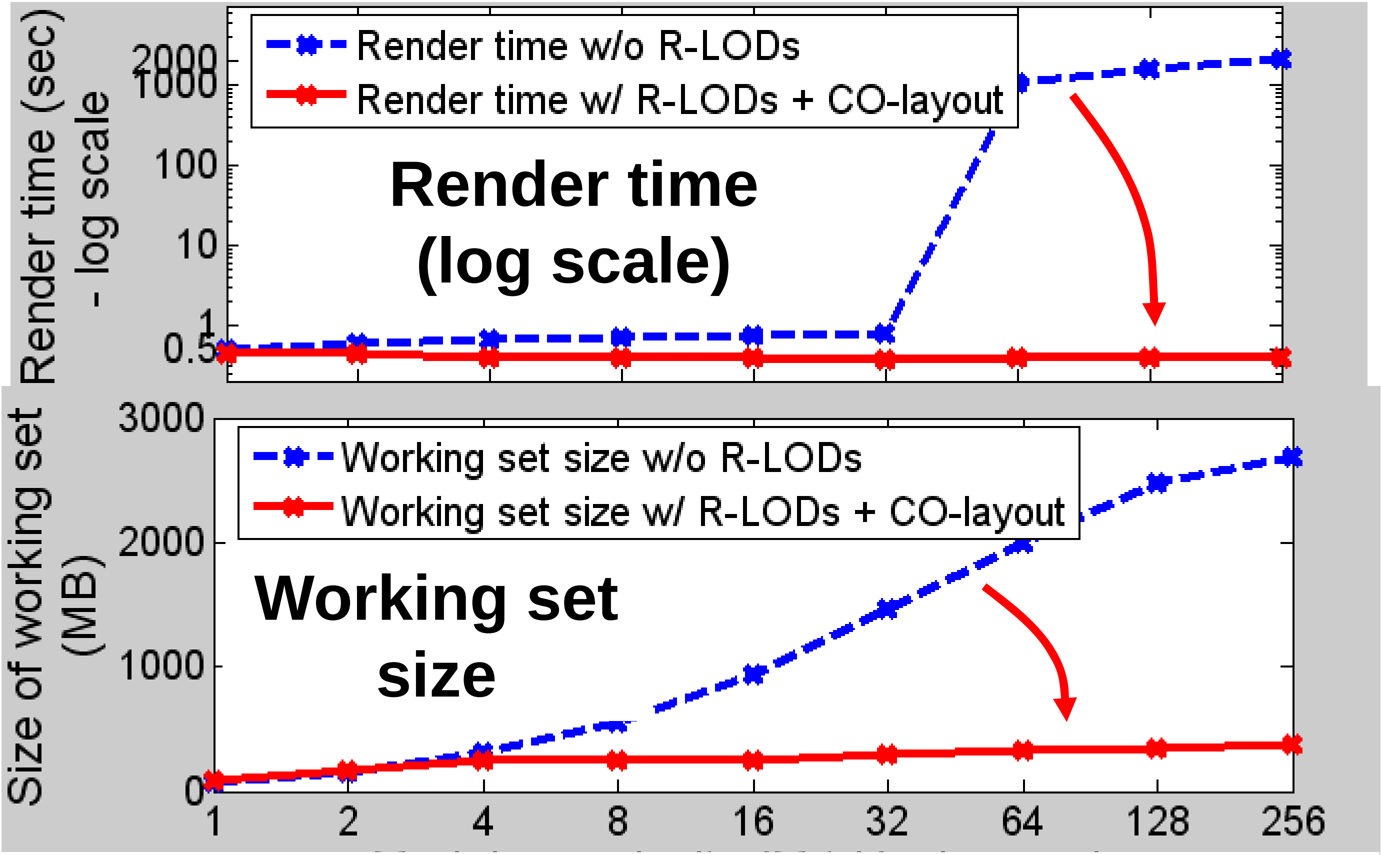
- **Measured with 2GB main memory**



Performance of LOD-based Ray Tracing

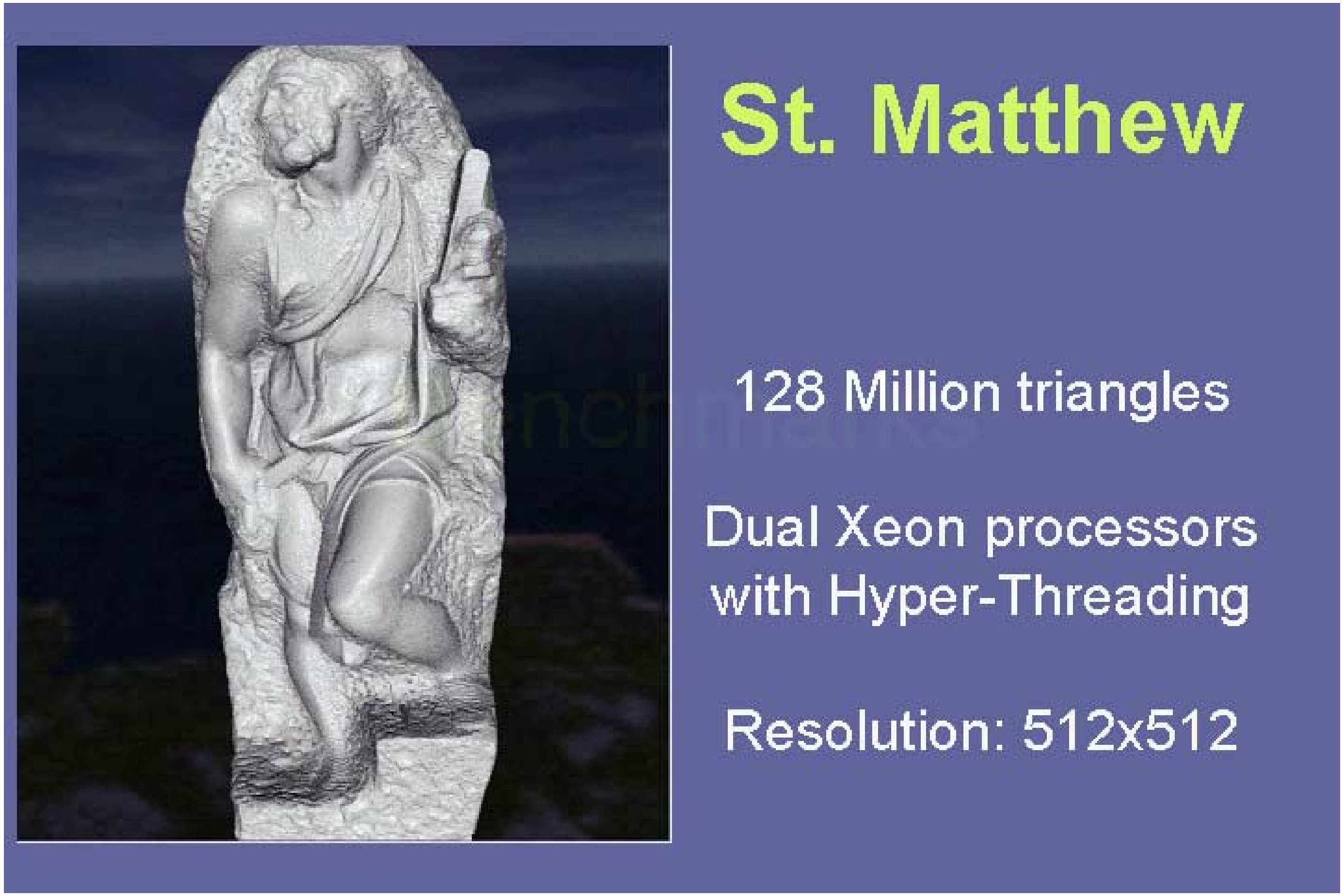
Achieved up to three order of magnitude speedup!

- **Measured with 2GB main memory**



Model complexity (M tri) - log scale

Real-time Captured Video – St. Matthew Model



512 by 512 and 2x2 super-sampling,
4 pixels of LOD error in image space

Related Work

- Interactive ray tracing
- LOD and out-of-core techniques
- LOD-based ray tracing

Interactive Ray Tracing

- **Ray coherences**
  - **[Heckbert and Hanrahan 84, Wald et al. 01, Reshetov et al. 05]**
- **Parallel computing**
  - **[Parker et al. 99, DeMarle et al. 04, Dietrich et al. 05]**
- **Hardware acceleration**
  - **[Purcell et al. 02, Schmittler et al. 04, Woop et al. 05]**
- **Large dataset**
  - **[Pharr et al. 97, Wald et al. 04]**

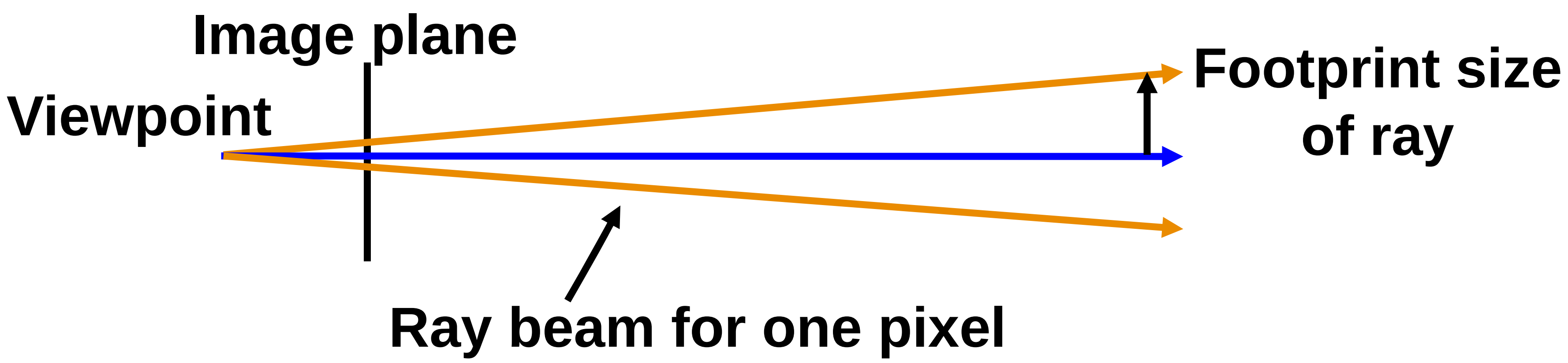
LOD and Out-of-Core Techniques

- **Widely researched**
  - **LOD book [Luebke et al. 02]**
  - **Out-core algorithm course [Chiang et al. 03]**
- **LOD algorithms combined with out-of-core techniques**
  - **Points clouds [Rusinkiewicz and Levoy 00]**
  - **Regular meshes [Hwa et al. 04, Losasso and Hoppe 04]**
  - **General meshes [Lindstrom 03, Cignoni et al. 04, Yoon et al. 04, Gobbetti and Marton 05]**

Not clear whether LOD techniques for rasterization is applicable to ray tracing

LOD-based Ray Tracing

- **Ray differentials [Igehy 99]**
  - **Subdivision meshes [Christensen et al. 03, Stoll et al. 06]**
  - **Point clouds [Wand and Straβer 03]**



Outline

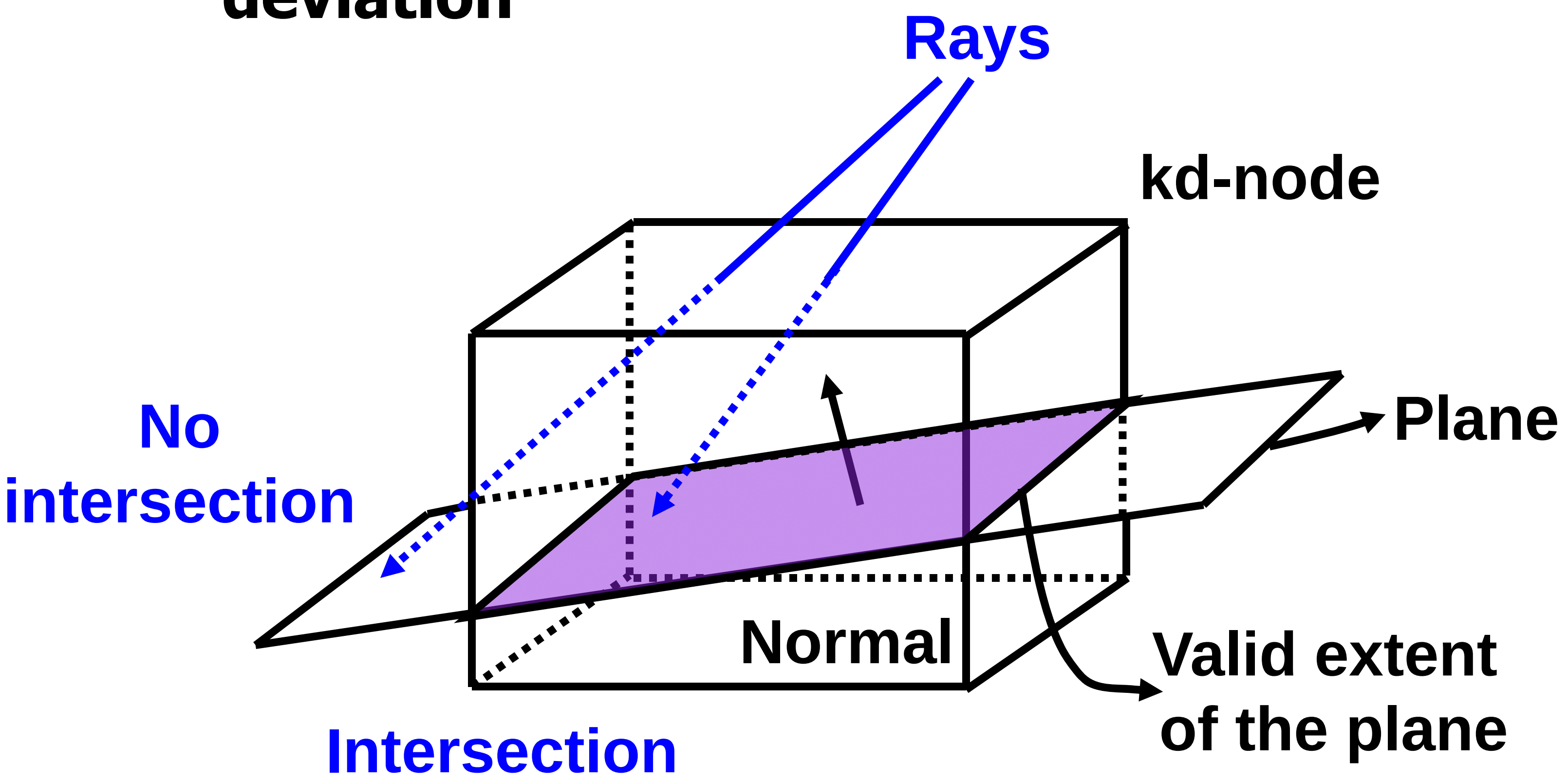
- R-LODs for ray tracing
- Results

Outline

- R-LODs for ray tracing
- Results

R-LOD Representation

- **Tightly integrated with kd-nodes**
  - **A plane, material attributes, and surface deviation**



Properties of R-LODs

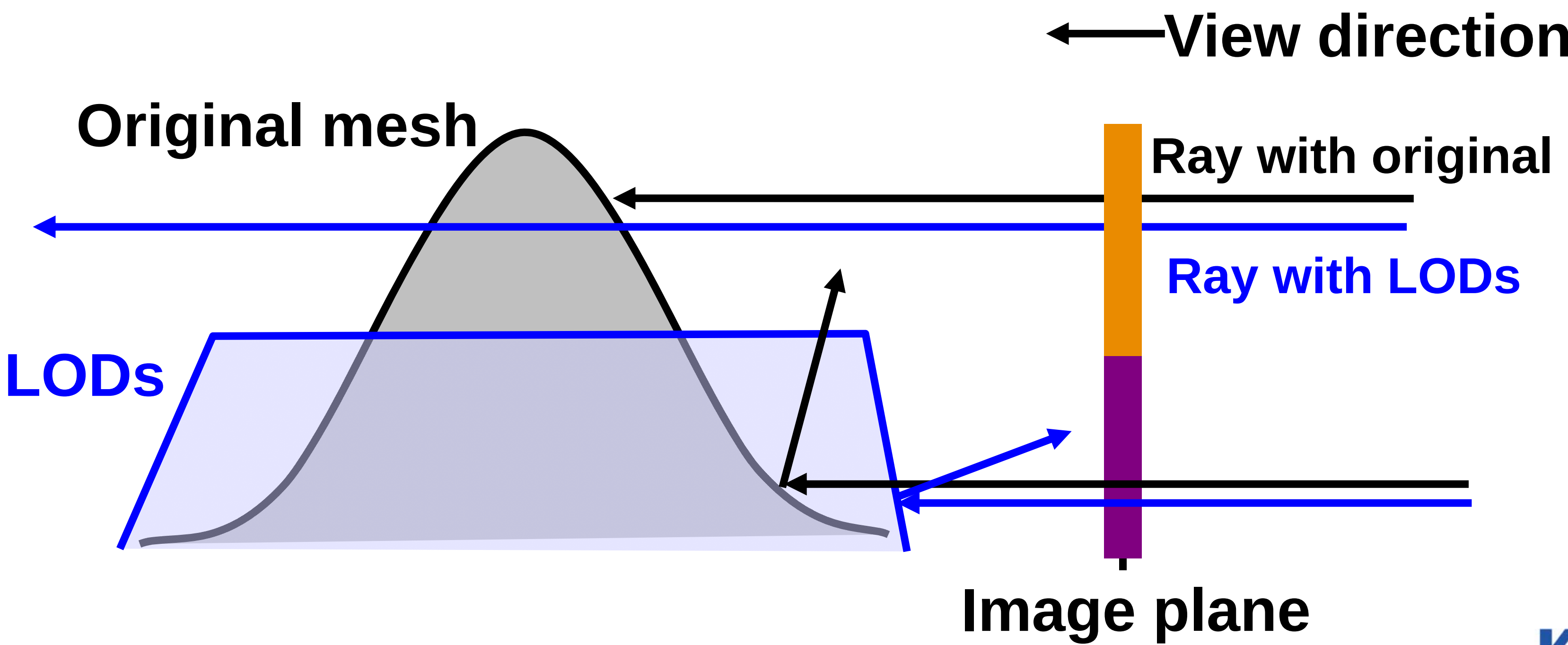
- **Compact and efficient LOD representation**
  - **Add only 4 bytes to (8 bytes) kd-node**
- **Drastic simplification**
  - **Useful for performance improvement**
- **Error-controllable LOD rendering**
  - **Error is measured in a screen-space in terms of pixels-of-error (PoE)**
  - **Provides interactive rendering framework**

Two Main Design Criteria for LOD Metric

- Controllability of visual errors
- Efficiency
  - Performed per ray (not per object)
  - More than tens of million times evaluation

Visual Artifacts

- **Visibility difference** Surface deviation
- **Illumination difference** Projected area
- **Path difference for secondary rays** Curvature difference



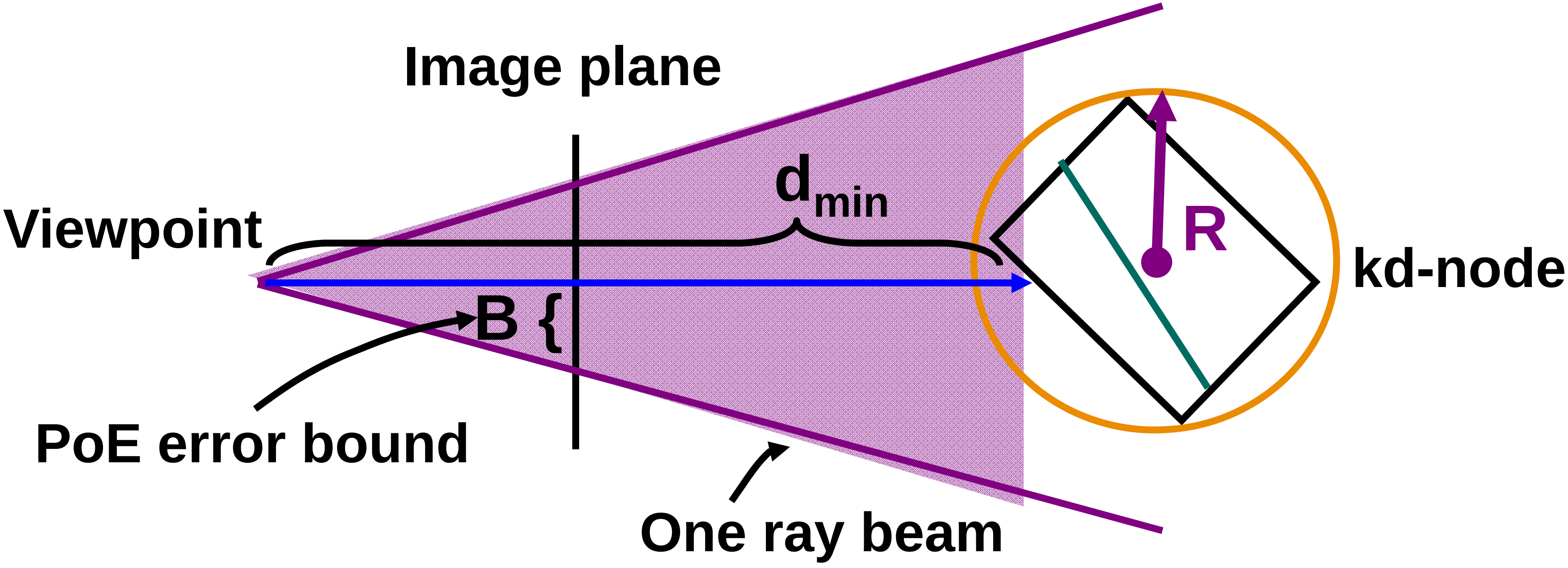
R-LOD Error Metric

- Consider two factors
  - Projected screen-space area of a kd-node
  - Surface deviation

Conservative Projection Method

- **Measures the screen-space area affected by using an R-LOD**

LOD metric: $C(B)\, d_{min} \overset{?}{>} R$



R-LODs with Different PoE Values

PoE: Original 1.85 5 10

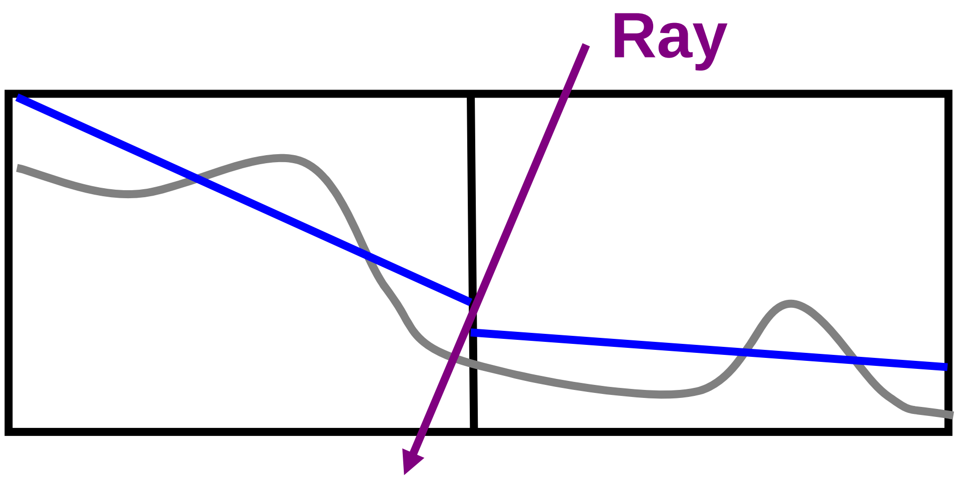
(512x512, no anti-aliasing)

LOD Metric for Secondary Rays

- **Applicable to any linear transformation**
  - **Shadow**
  - **Planar reflection**

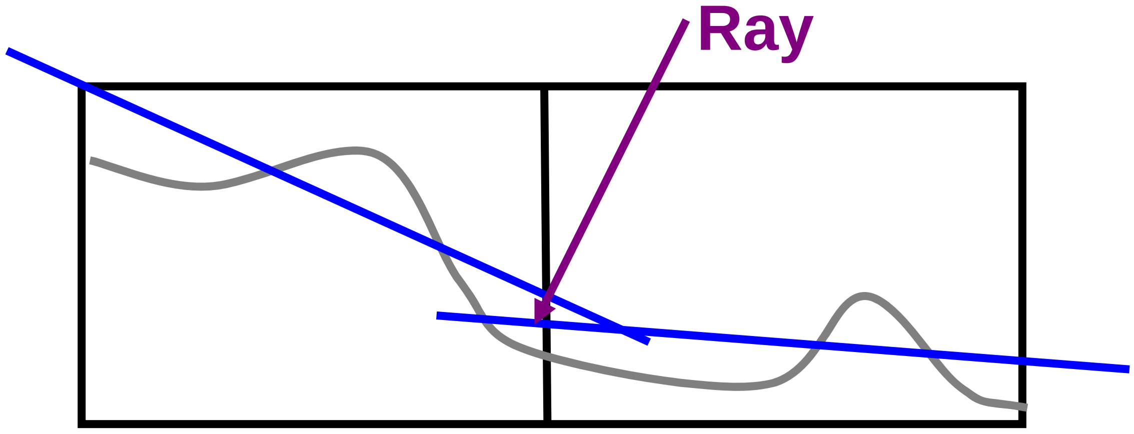
- **Not applicable to non-linear transformation**
  - **Refraction**
  - **Uses more general, but expensive ray differentials [Igehy 99]**

C^0 Discontinuity between R-LODs



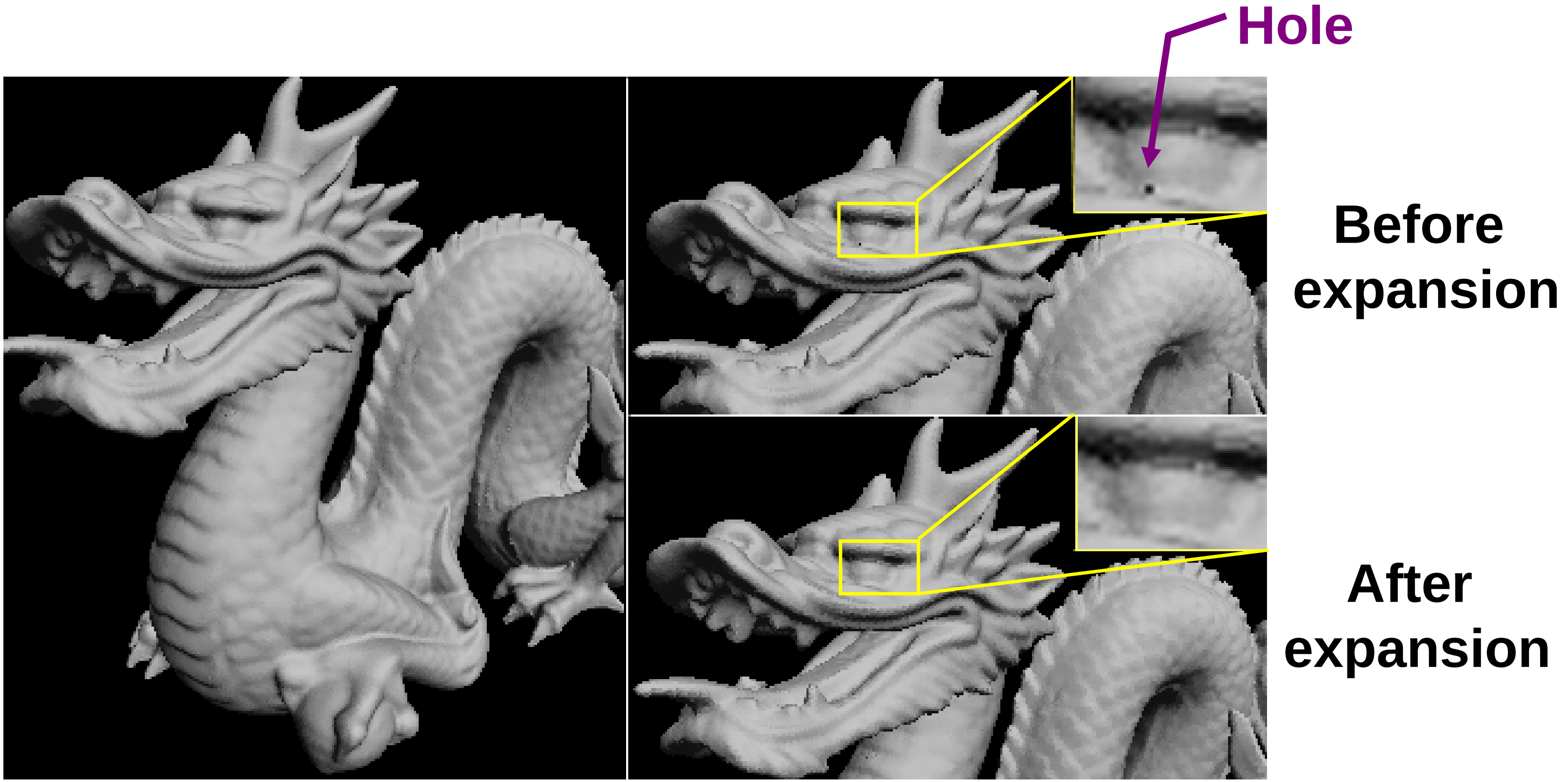
- **Possible solutions**
  - **Posing dependencies** [Lindstrom 03, Hwa et al. 04, Yoon et al. 04, Cignoni et al. 05]
  - **Implicit surfaces** [Wald and Seidel 05]

Expansion of R-LODs



- Expansion of the extent of the plane
  - Inspired by hole-free point clouds rendering [Kalaiah and Varshney 03]
  - A function of the surface deviation (20% of the surface deviation)

Impact of Expansions of R-LODs

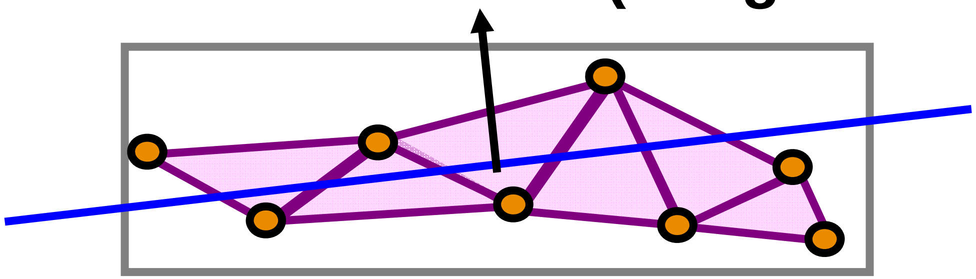


Original model

PoE = 5
at 512 by 512

R-LOD Construction

- Principal component analysis (PCA)
  - Compute the covariance matrix for the plane of R-LODs



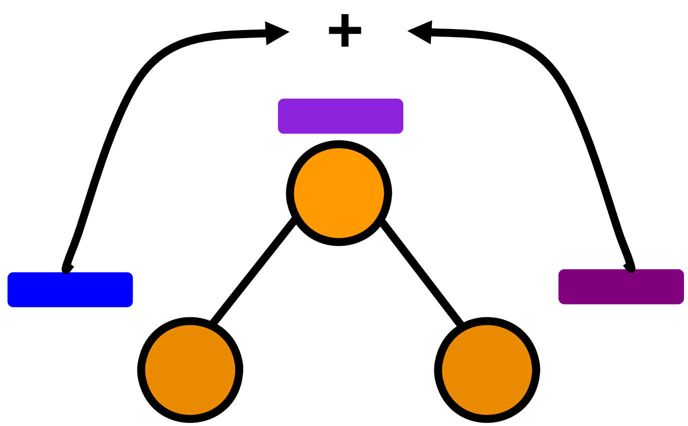
- Hierarchical PCA computation
  - Has linear time complexity
  - Accesses the original data only one time with virtually no memory overhead

Hierarchical PCA Computation with Linear Time Complexity

$$\sigma_{xy} = \sum_{k=1}^{n} (x_k - \mu_x)(y_k - \mu_x),$$

where x_k **,** y_k **are x, y coordinates of kth points**

$$\sigma_{xy} = \boxed{\sum_{k=1}^{n} x_k y_k} - \frac{2}{\boxed{n}} \boxed{\sum_{k=1}^{n} x_k} \boxed{\sum_{k=1}^{n} y_k} + \frac{2}{n^2} \sum_{k=1}^{n} x_k \sum_{k=1}^{n} y_k$$



KAIST

Utilizing Coherence

- **Ray coherence**
  - **Using LOD improve the utilization of SIMD traversal/intersection**
- **Cache coherence**
  - **Use cache-oblivious layouts of bounding volume hierarchies [Yoon and Manocha 06]**
  - **10% ~ 60% performance improvement**

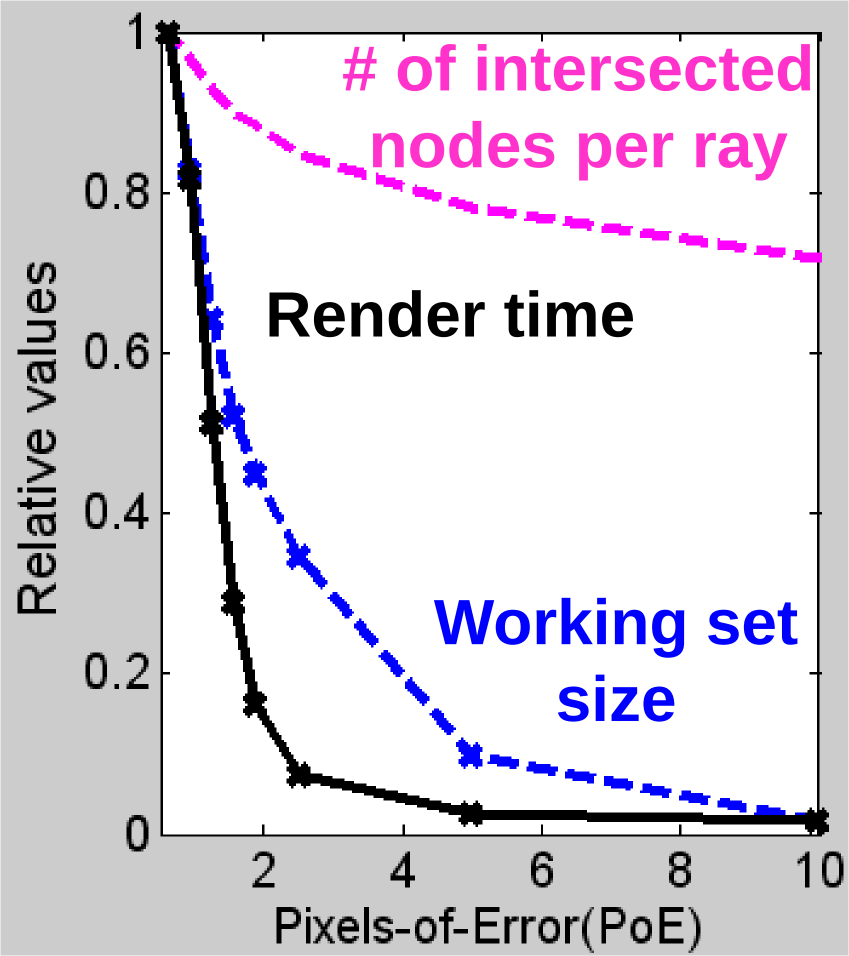
Outline

- R-LODs for ray tracing
- Results

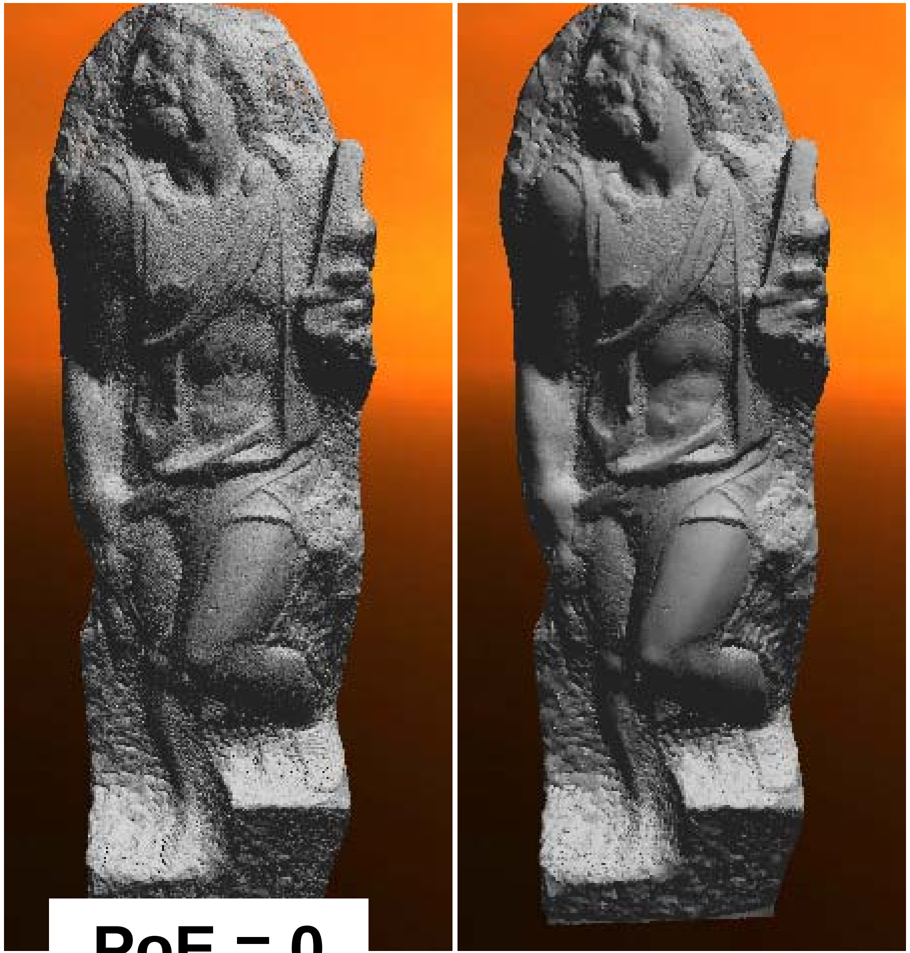
Implementation

- **Uses common optimized kd-tree construction methods**
  - **Based on surface-area heuristic [MacDonald and Booth 90, Havran 00]**
- **Out-of-core computation**
  - **Decompose an input model into a set of clusters [Yoon et al. 04]**

Impact of R-LODs



10X speedup

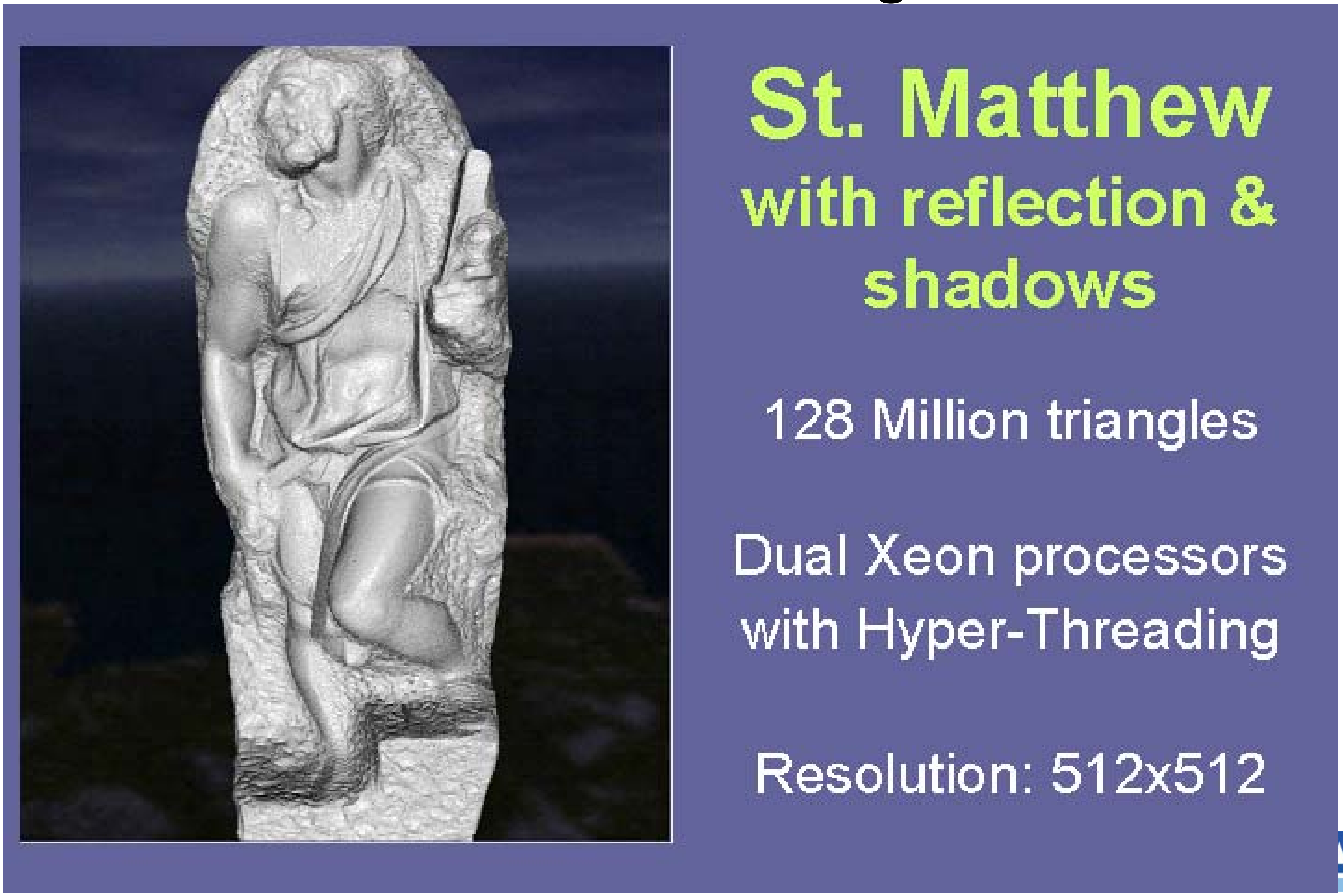


PoE = 0 (No LOD)

PoE = 2.5

Real-time Captured Video – St. Matthew Model

512 x 512, 2 x 2 anti-aliasing, PoE = 4



Pros and Cons

- Limitations
  - Does not handle advanced materials such as BRDF
  - No guarantee there is no holes

- Advantages
  - Simplicity
  - Interactivity
  - Efficiency

Conclusion

- LOD-based ray tracing method
  - R-LOD representation
  - Efficient LOD error metric
  - Hierarchical LOD construction method with a linear time complexity
  - Reduce the working set size

Ongoing and Future Work

- Investigate an efficient use of implicit surfaces
- Allow approximate visibility
- Extend to global illumination

At the Next Class

- **Will discuss cache-coherent layouts**