# Automatic Creation of Object Hierarchies for Ray Tracing of Dynamic Scenes

- WSCG '07 -

Martin Eisemann, Thorsten Grosch, Marcus Magnor, Stefan Müller

eisemann@cg.tu-bs.de

Computer Graphics Lab,
TU Braunschweig, Germany

Institute for Computational Visualistics,
University of Koblenz-Landau, Germany

# Motivation

## Ray Tracing

- Interactive or real-time frame rates possible
  - [Parker et al. 99], [Wald 04], [Reshetov et al. 05]
- Strongly dependent on acceleration data structures
- Optimized for static scenes

# Ray Tracing of Dynamic Scenes

## Related Work:

[Glassner 88], [Reinhard et al. 00], [Lext et al. 01], [Günther et al. 06], [Lauterbach et al. 06], [Wächter et al. 06], [Wald et al. 06]

## 2 Methods for dynamic scenes:

- Dynamic Goldsmith and Salmon

  , see also [Goldsmith and Salmon 87]

- Loose Bounding Volume Hierarchy

  , see also [Ulrich 00]

# Dynamic Goldsmith and Salmon

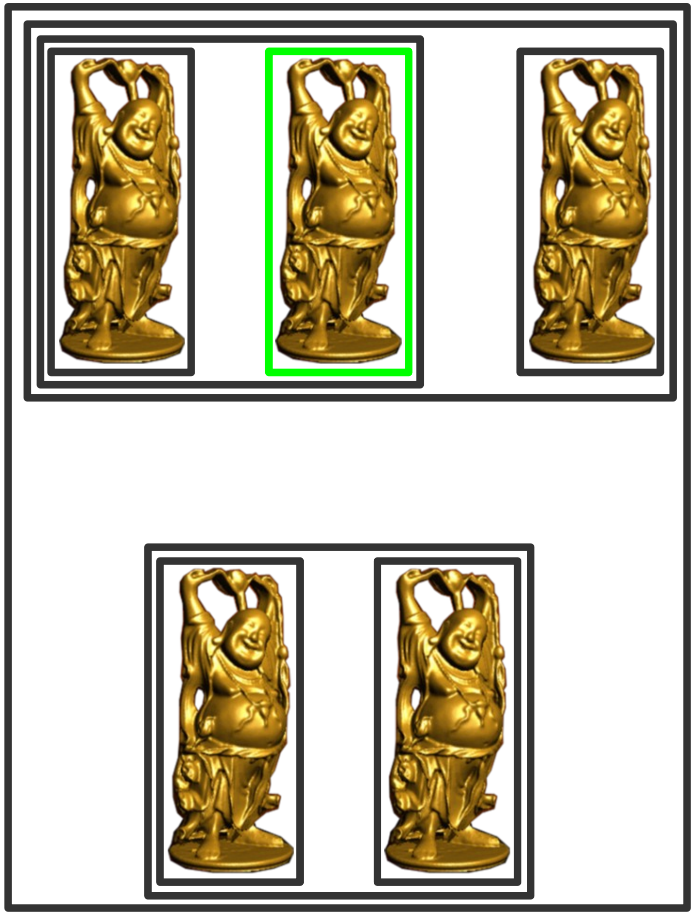
- Goals:
  - Exploitation of localities
  - Prevention of thinning
- Initial build:
  - Minimize amount of expected intersection tests

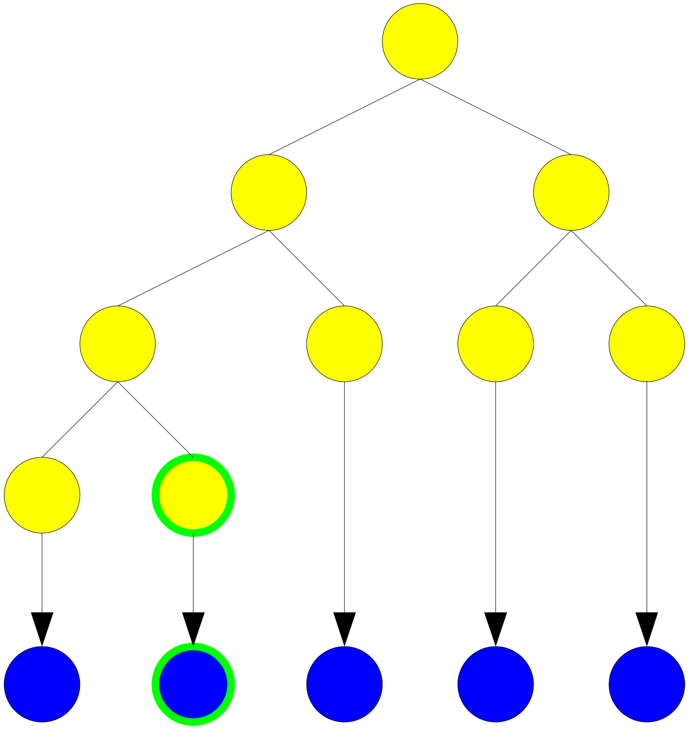
$$C = 1 + \sum_{i=0}^{n} \frac{S(N_i)\#Children(N_i)}{S(Root)}$$

  - Use SAH, Median-Cut, G. & S., ...

# Dynamic Goldsmith and Salmon

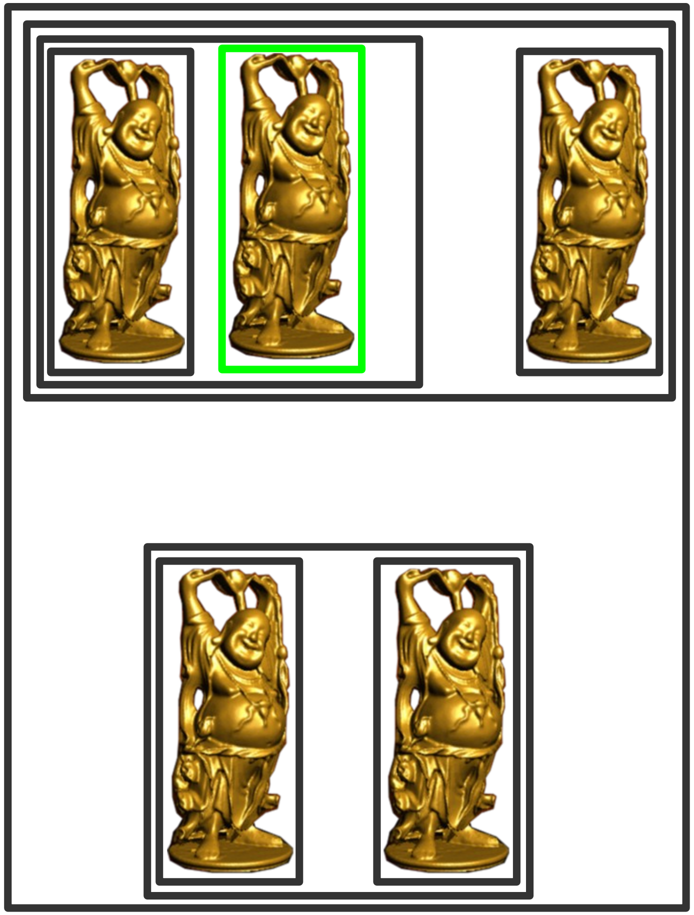
## Exploitation of localities

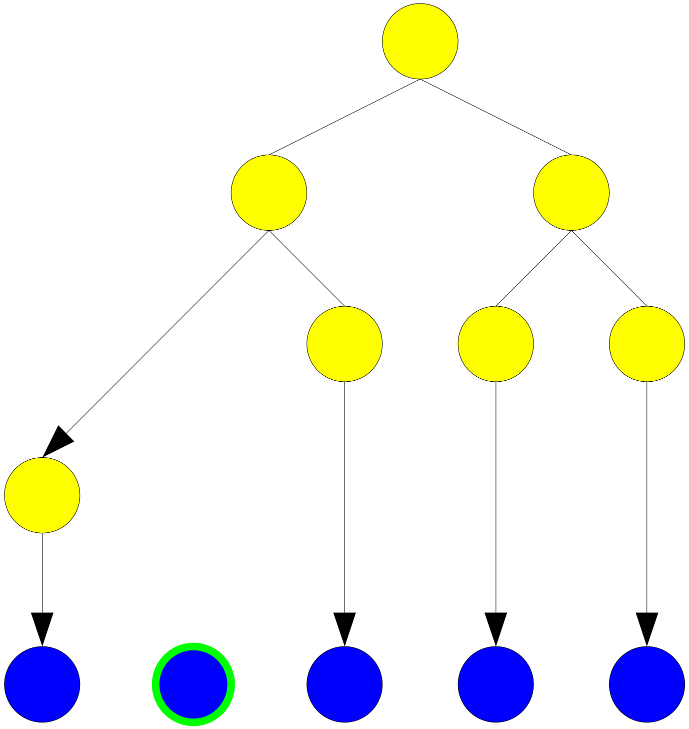




# Dynamic Goldsmith and Salmon

## Exploitation of localities

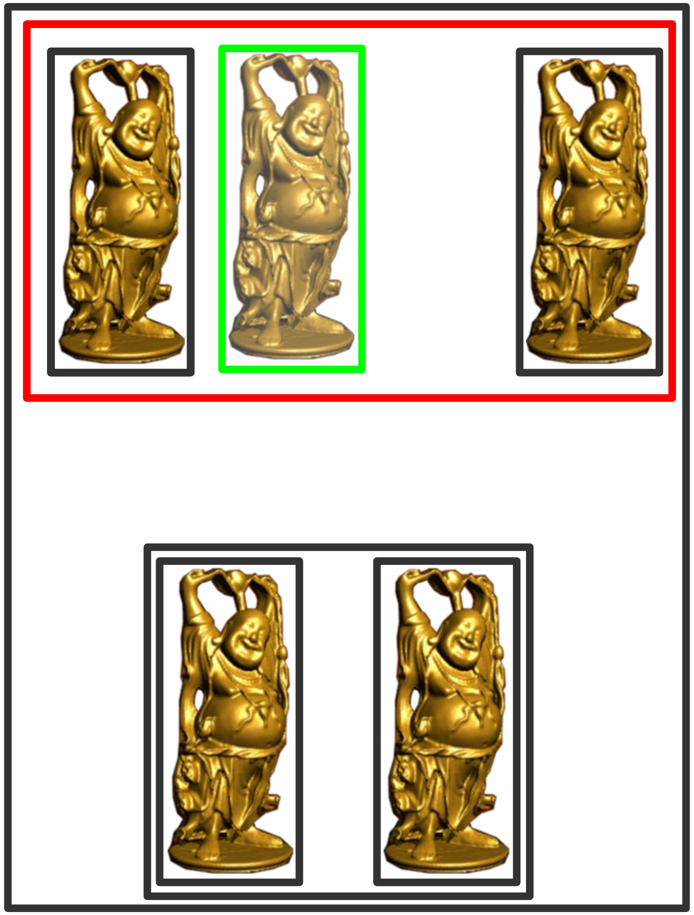


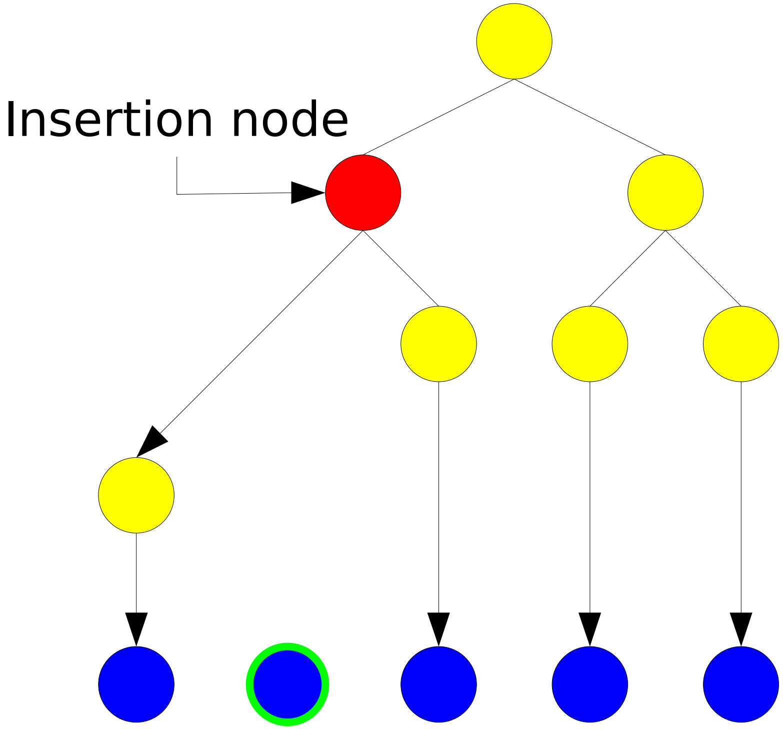


1. Deletion

# Dynamic Goldsmith and Salmon

## Exploitation of localities

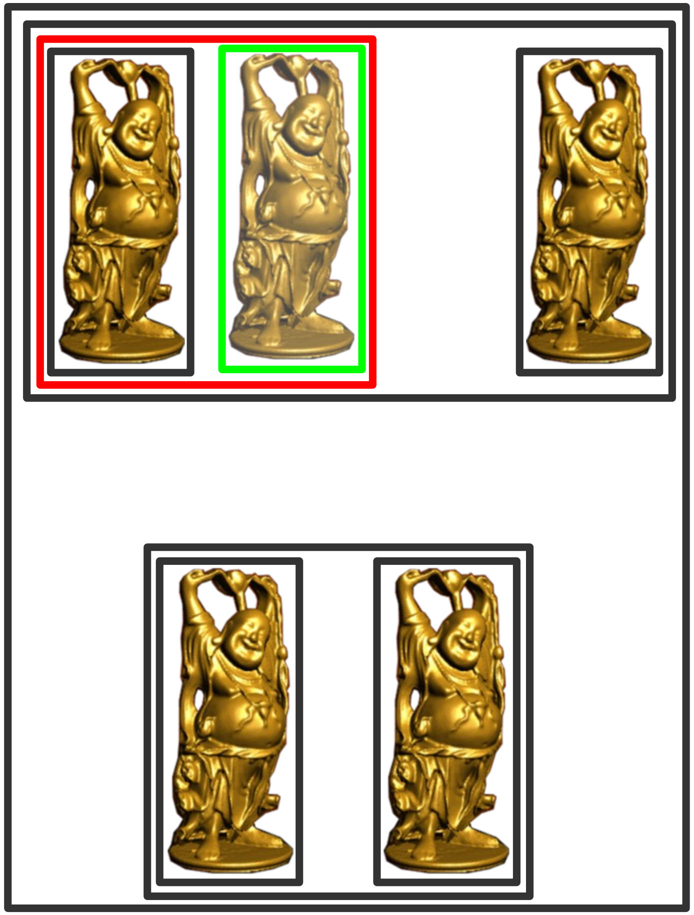


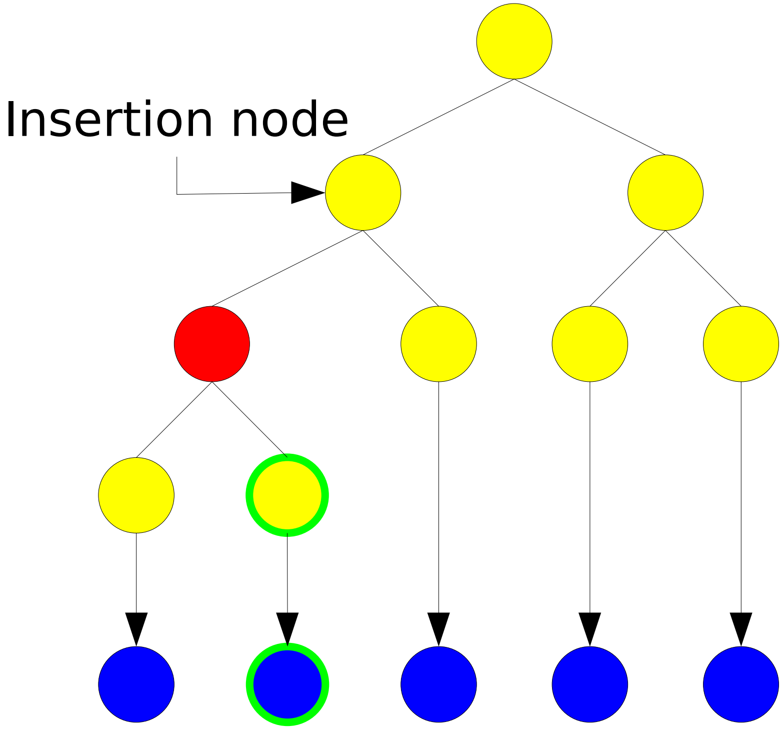


1. Deletion
2. Find new insertion node

# Dynamic Goldsmith and Salmon

## Exploitation of localities





1. Deletion
2. Find new insertion node
3. Reinsertion

# Dynamic Goldsmith and Salmon

Thinning

- Number of objects in a node decreases
- Surface area stays constant

- Needs quality criterion

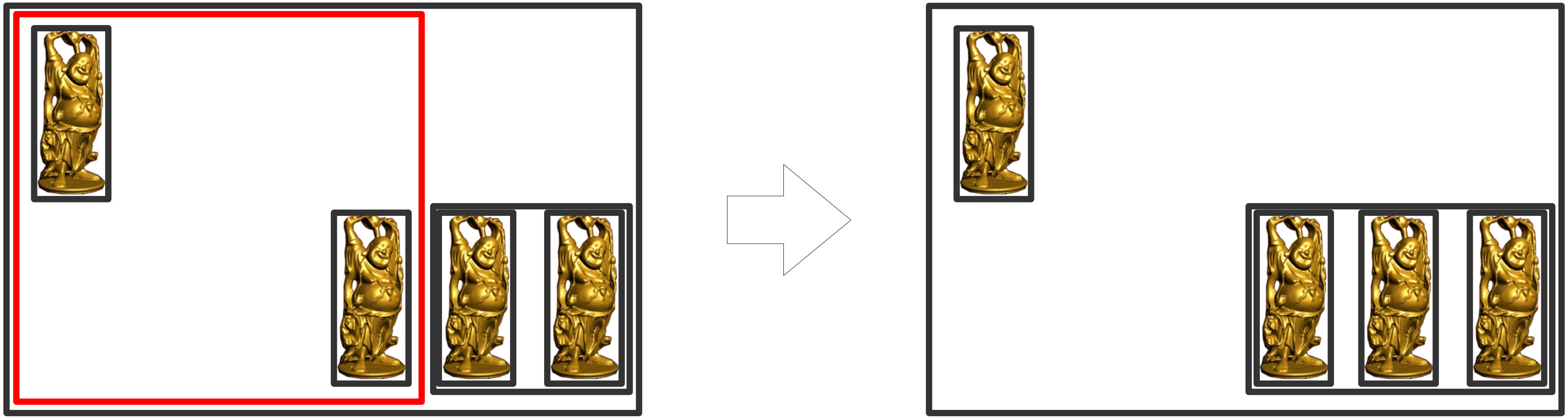
$$Q(N) = \frac{S(N)}{\#Objects(N)}$$

- Initial calculation of Q(N) for every node

# Dynamic Goldsmith and Salmon

Thinning

- If threshold is exceeded during animation:
  1. Delete node
  2. Reinsert child nodes

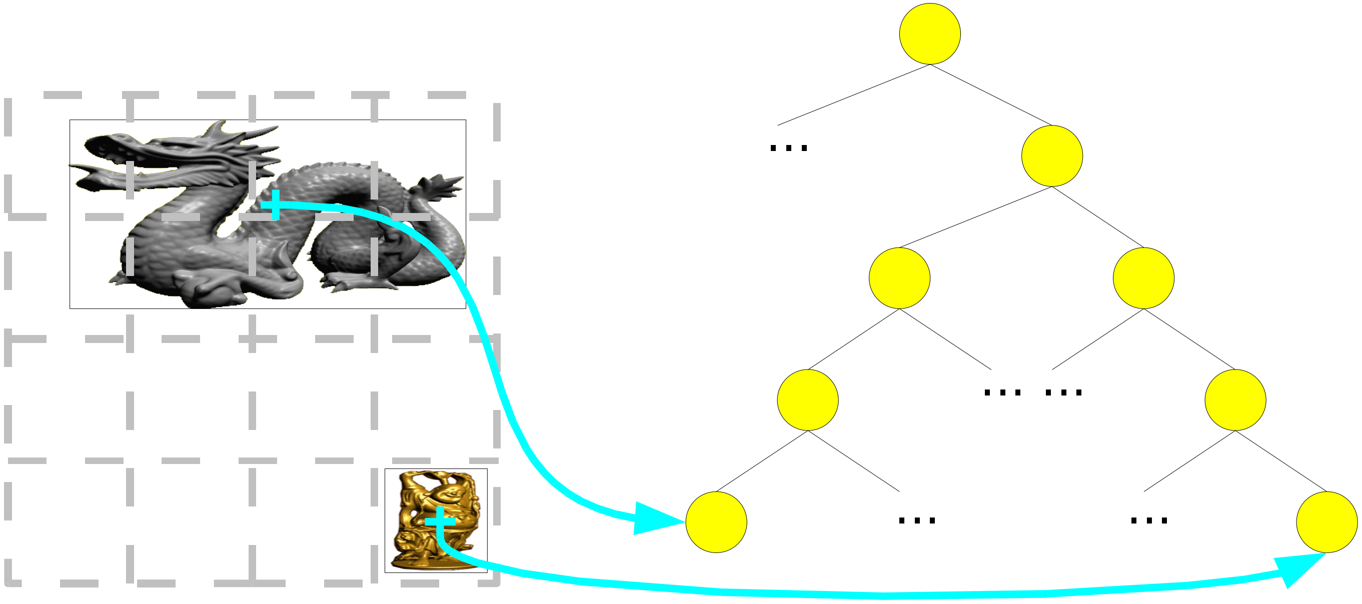


# Loose Bounding Volume Hierarchy

- Reconstruction in O(n)
- Hybrid between spatial acceleration data structure and BVH
  - Spatial median-cut with alternating axes
  - User-defined depth of 3N

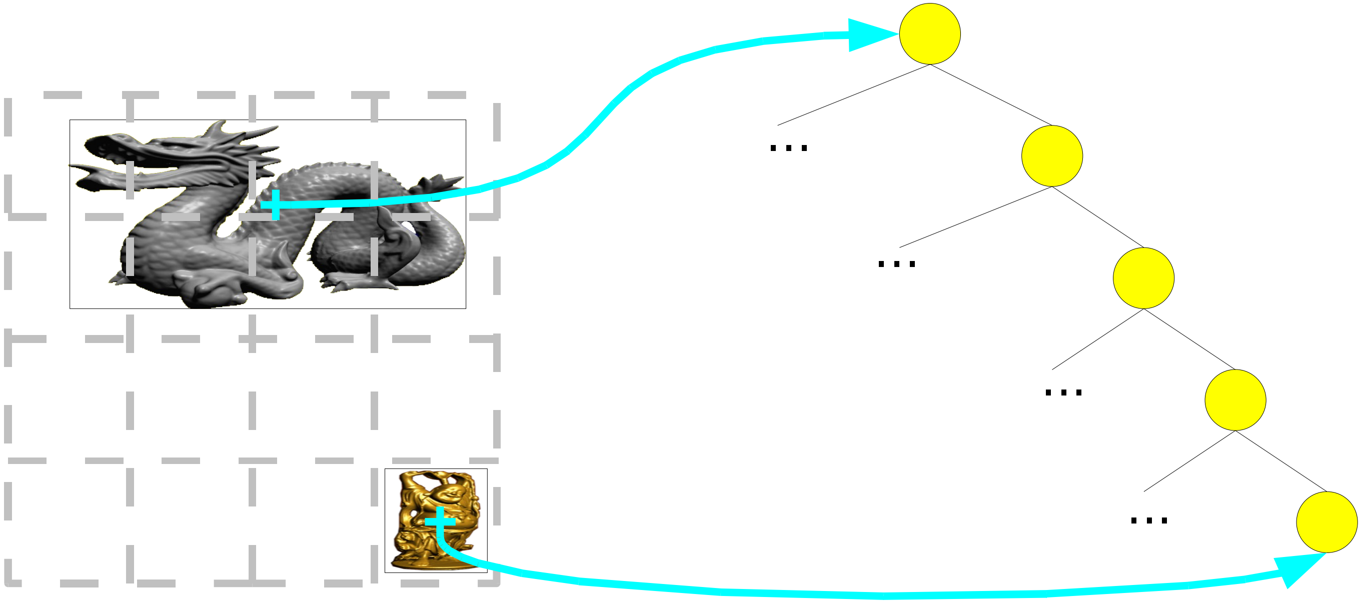
# Loose Bounding Volume Hierarchy

- Lowest level of BVH is a pseudo-uniform grid
- Resolution $2^N$ x $2^N$ x $2^N$



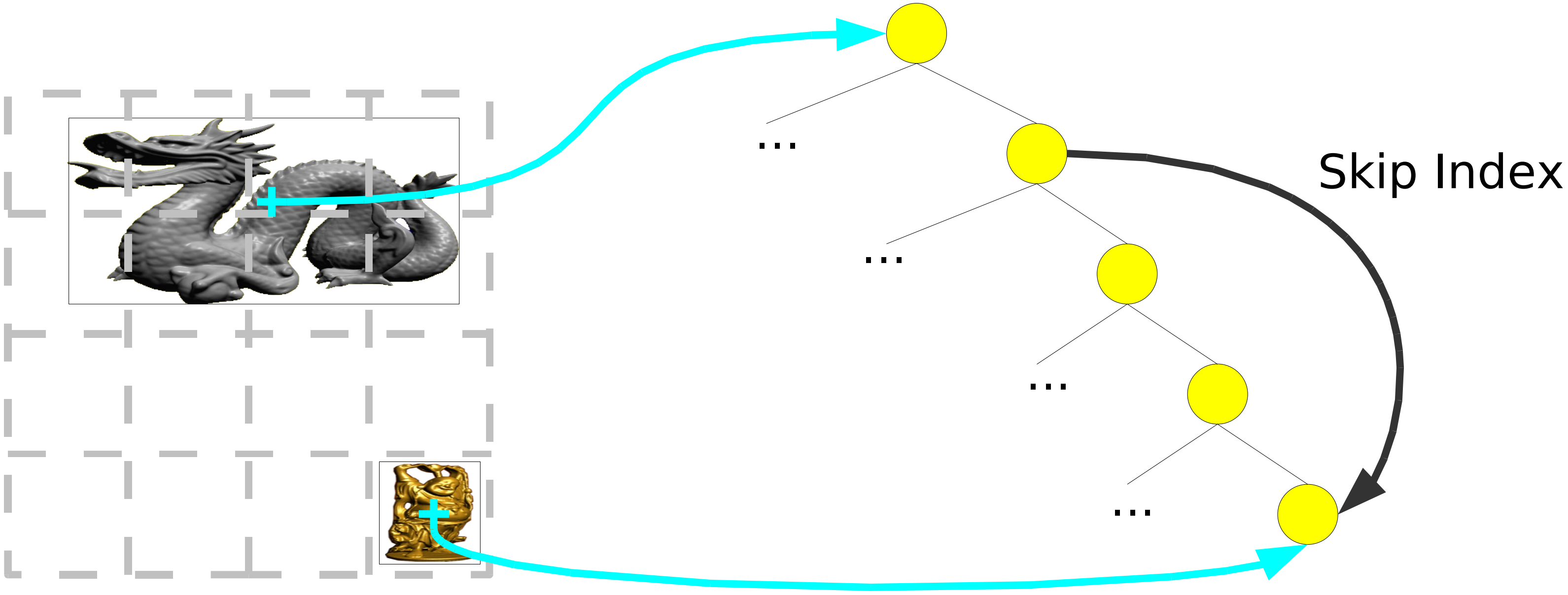
# Loose Bounding Volume Hierarchy

## Wide Object Isolation



# Loose Bounding Volume Hierarchy

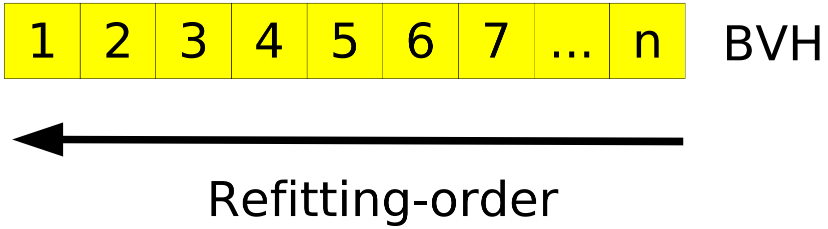
## Skip Indices



# Loose Bounding Volume Hierarchy

Refitting by backward iteration

- Adjusts BVs
- Sets skip indices
- Marks empty nodes



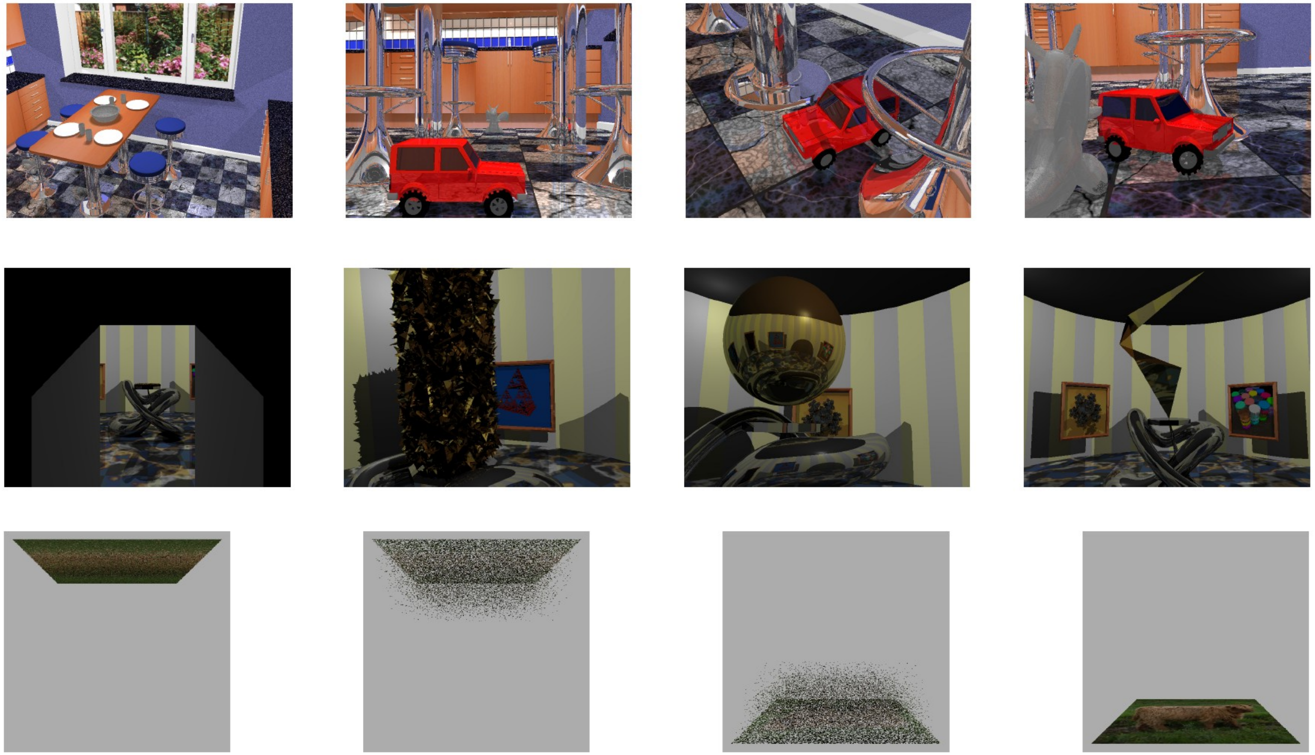
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | … | n | BVH |
|---|---|---|---|---|---|---|---|---|---|

Refitting-order

# Test Results

Test scenes:

between 5 and 149.058 animated objects



# Test Results

## Dynamic G. & S.

## Loose BVH

| Update-Phase | RT-Phase | | Update-Phase | RT-Phase |
|---|---|---|---|---|
| 6x – 103x | 1.0x – 1.9x | **speed-up** | 11.2x – 18.5x | 0.5x – 7.0x |
| 17ms – 907ms | 6.0s – 11.4s | **avg. timings** | 125ms – 404ms | 3.2s – 11.7s |

- Local movement
- up to a few hundred objects
- Good overall performance

- Fast and constant updates
- Several thousand objects
- Teapot in the stadium problem

# End

http://graphics.tu-bs.de/people/eisemann