Interactive Collision Detection for Deformable and Fracturing Objects

Sung-Eui Yoon

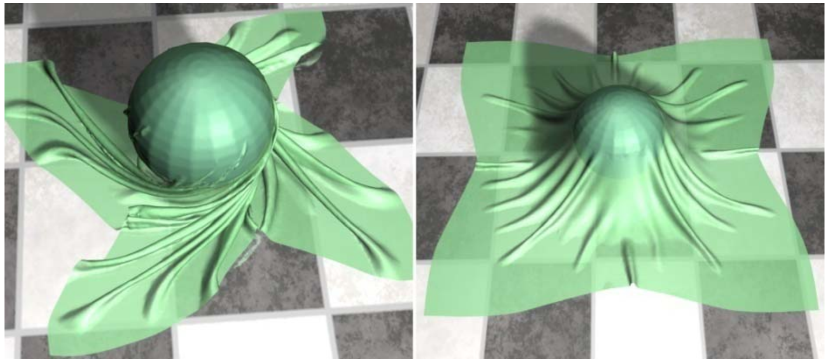
Scalable Graphics Lab.

KAIST

http://sglab.kaist.ac.kr/~sungeui/

Objectives

- Discuss techniques for collision detection for deformable and fracturing models
  - Basic BVH-based collision detection
  - Hybrid parallel collision detection
  - Fracturing-aware collision detection



<Cloth-ball, 94K triangles>

<Breaking dragon, 252K triangles>

Overview

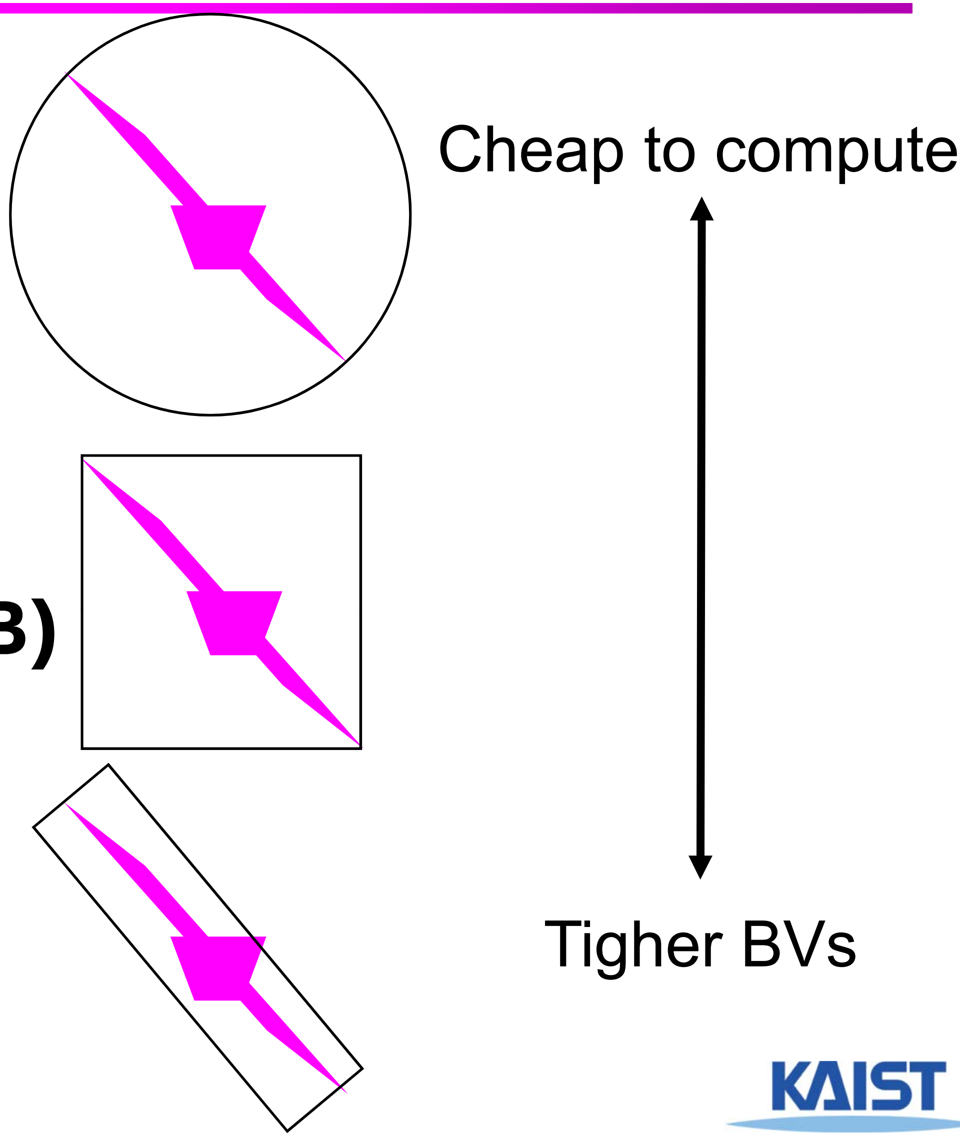
- Background: BVH-based collision detection
- HPCCD: Hybrid parallel continuous collision detection
- FASTCD: Fracturing-Aware Stable Collision Detection

Overview

- Background: BVH-based collision detection
- HPCCD: Hybrid parallel continuous collision detection
- FASTCD: Fracturing-Aware Stable Collision Detection

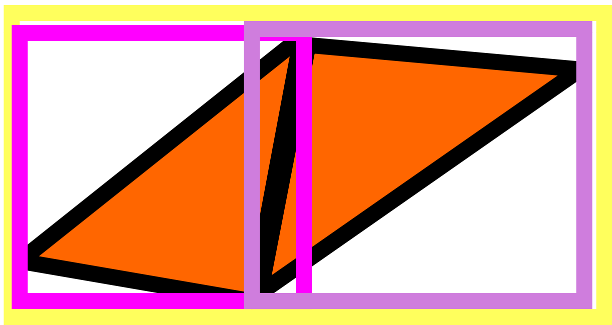
Bounding Volumes

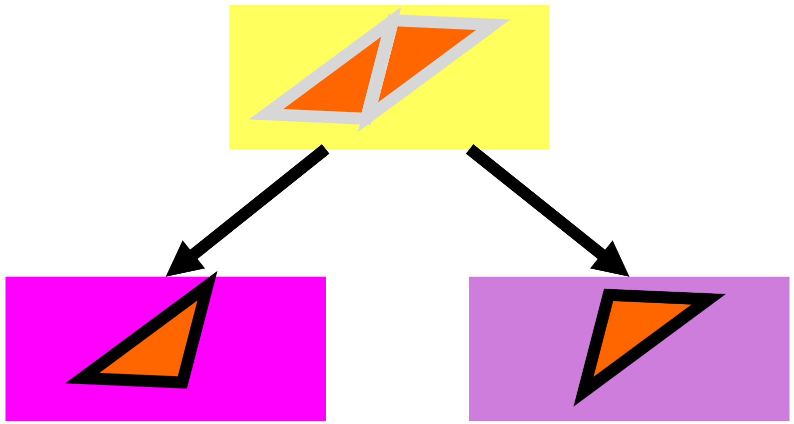
- **Sphere**
- **Axis-aligned bounding box (AABB)**
- **Oriented bounding box (OBB)**



Bounding Volume Hierarchies (BVHs)

- **Organize bounding volumes recursively as a tree**
- **Construct BVHs in a top-down manner**
  - **Use median-based partitioning or other advanced partitioning methods**

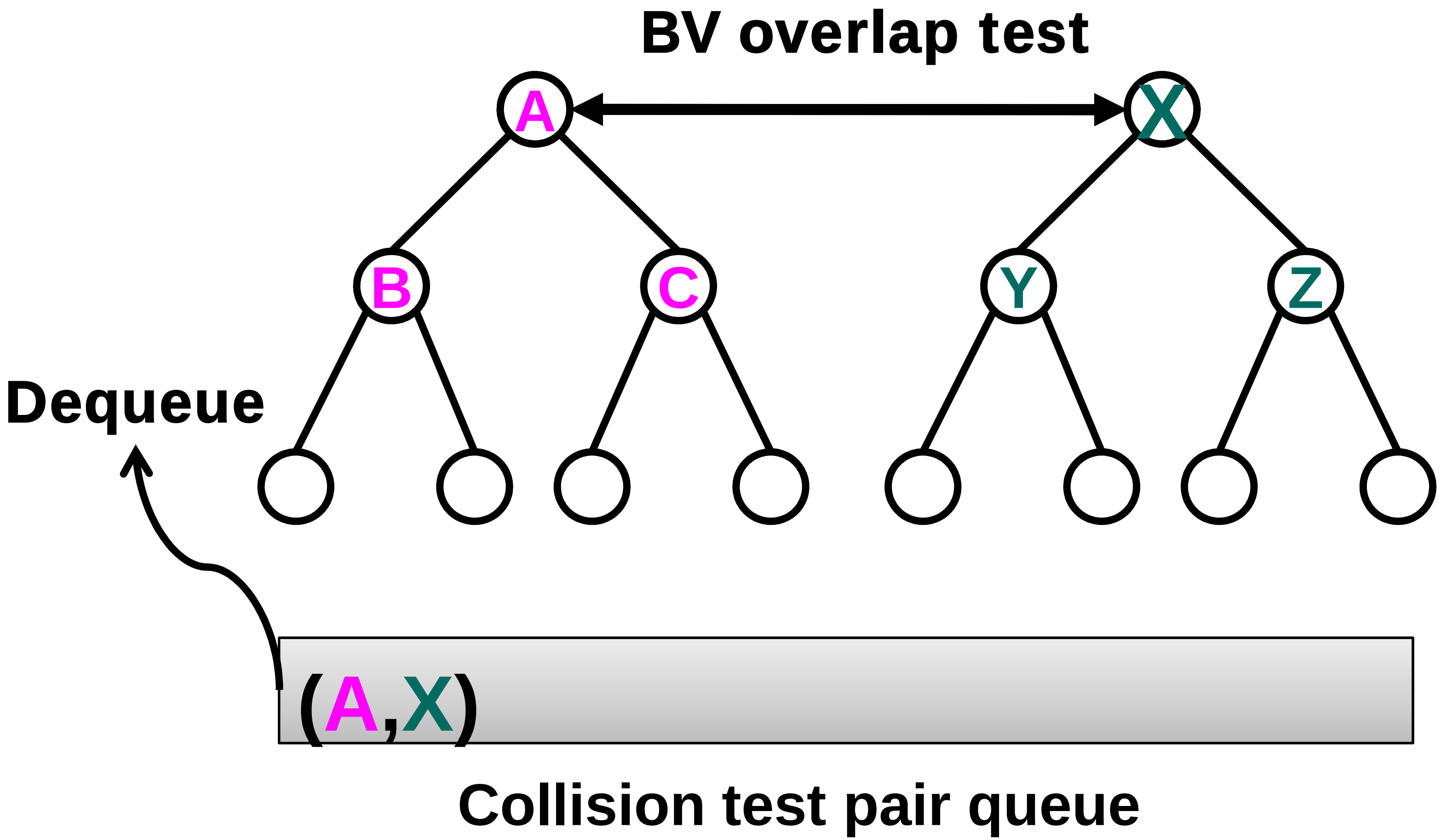




A BVH

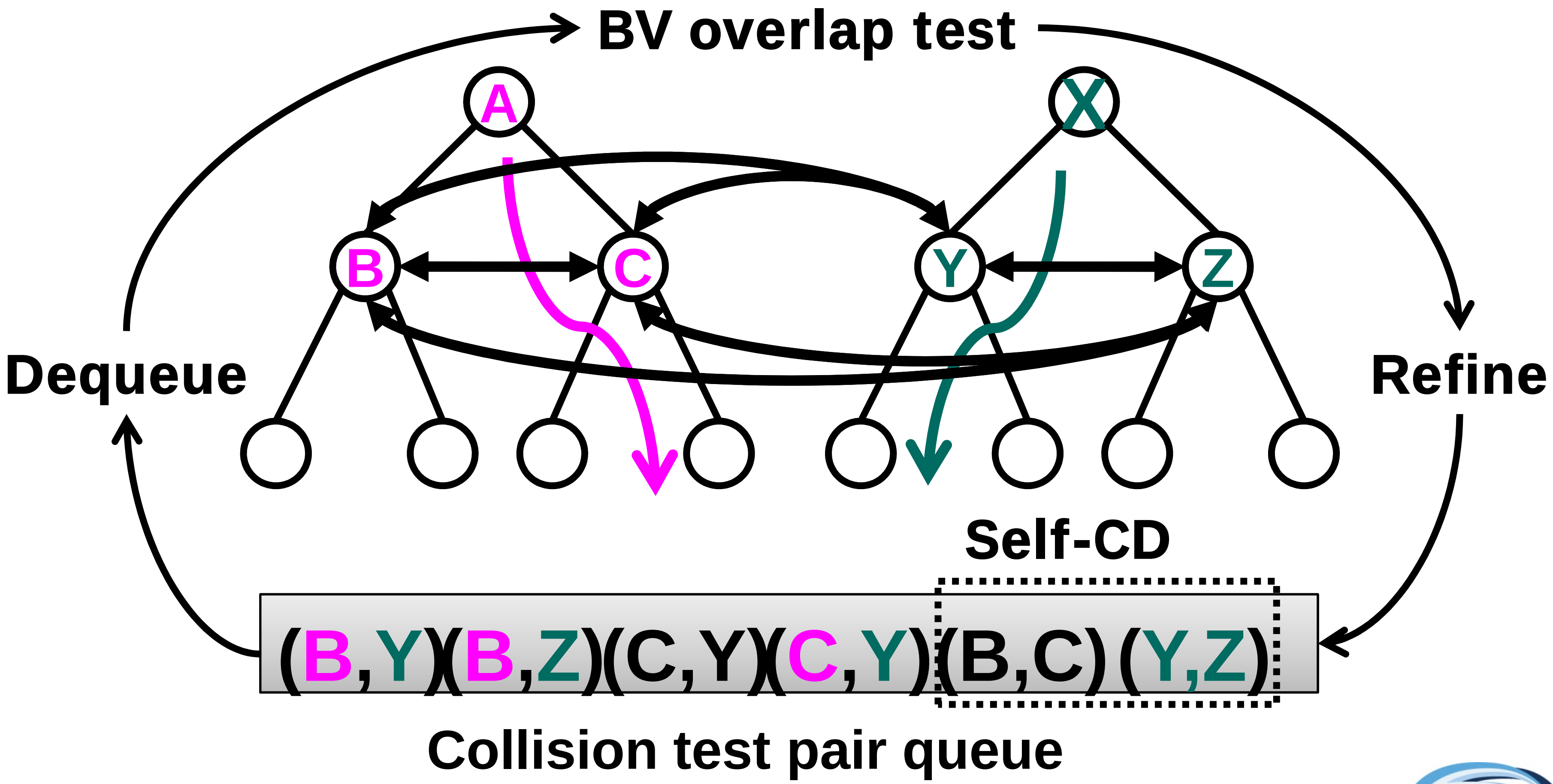
BVH-based Collision Detection

- BVH traversal



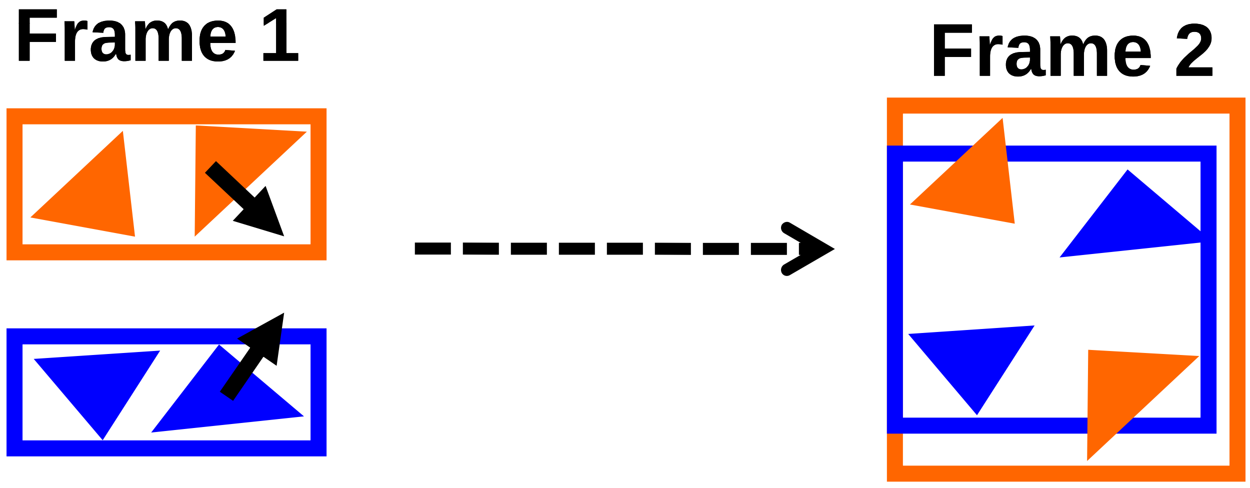
BVH-based Collision Detection

- **BVH traversal**



BVH Update

- Reconstruct BVHs from scratch
- BV Refitting
  - Refit BVs with deformed vertices
  - Performed efficiently in a bottom-up traversal
  - Can have loose BVs when deformation levels are high



Discrete vs. Continuous

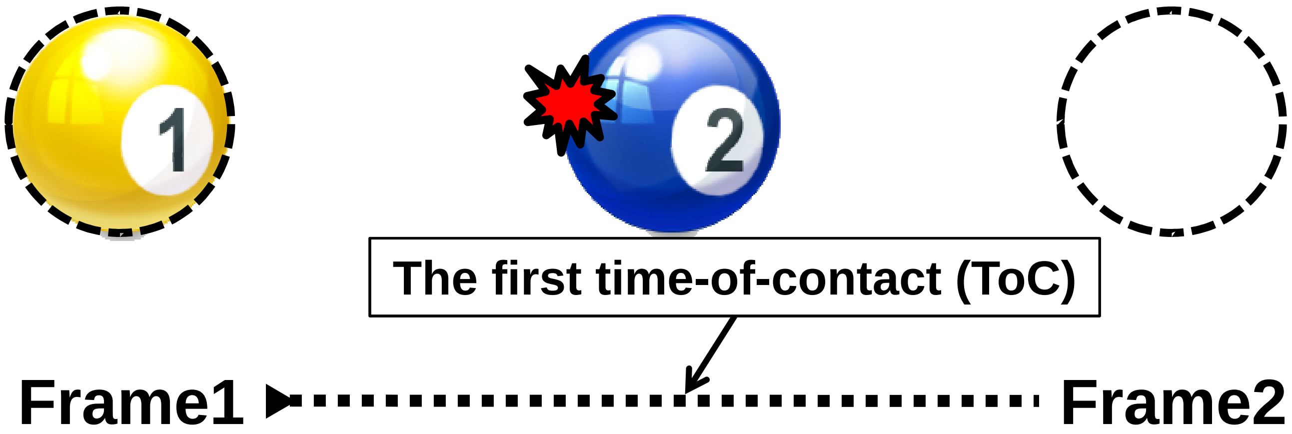
- **Discrete collision detection (DCD)**
  - **Detect collisions at each frame**
  - **Fast, but can miss collisions**

Miss collisions

Frame1 **Frame2**

Discrete vs. Continuous

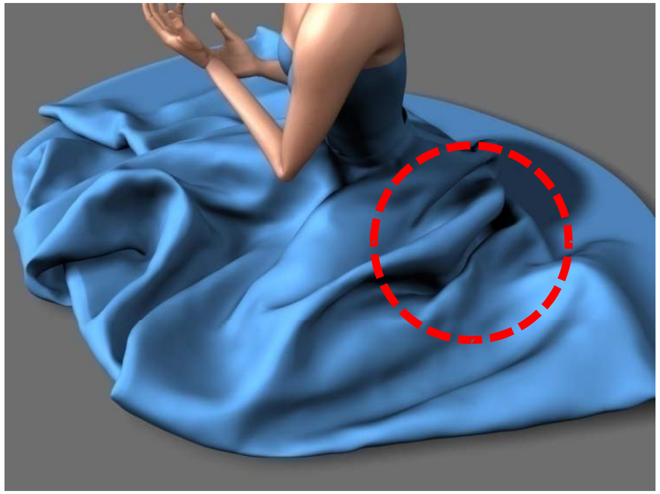
- Discrete collision detection (DCD)
- Continuous collision detection (CCD)
  - Identify the first time-of-contact (ToC)
  - Accurate, but requires a long computation time
  - Vertex-face & edge-edge elementary tests [Provot96]



Inter- and Self-Collisions

- **Inter-collisions**
  - **Collisions between two objects**

- **Self-collisions**
  - **Collisions between two regions of a deformable object**
  - **Takes a long computation time to detect**



From Govindaraju's paper

Overview

- Background
- HPCCD: Hybrid parallel continuous collision detection
- FASTCD: Fracturing-Aware Stable Collision Detection

Parallel Computing Trends

- Many core architectures
  - Multi-core CPU architectures
  - GPU architectures
- Heterogeneous architectures
  - Intel's Larabee and AMD's Fusion
- Designing parallel algorithms is important to utilize these parallel architectures

Recent Parallel Collision Detection Methods

- **CPU-based CD method**
  - **Tang et al., Solid and Physical Modeling, 2009**

- **gProximity: GPU-based CD method**
  - **Lauterbach et al., Eurographics 2010**

- **HPCCD: Hybrid parallel CD method**
  - **Kim et al., Pacific Graphics 2009**

Recent Parallel Collision Detection Methods

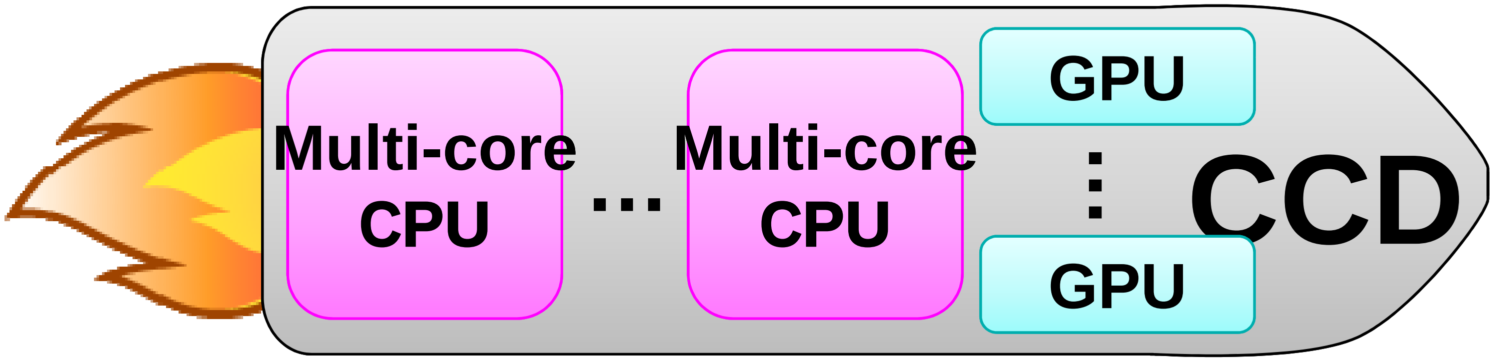
- **CPU-based CD method**
  - **Tang et al., Solid and Physical Modeling, 2009**

- **gProximity: GPU-based CD method**
  - **Laterbach et al., Eurographics 2010**

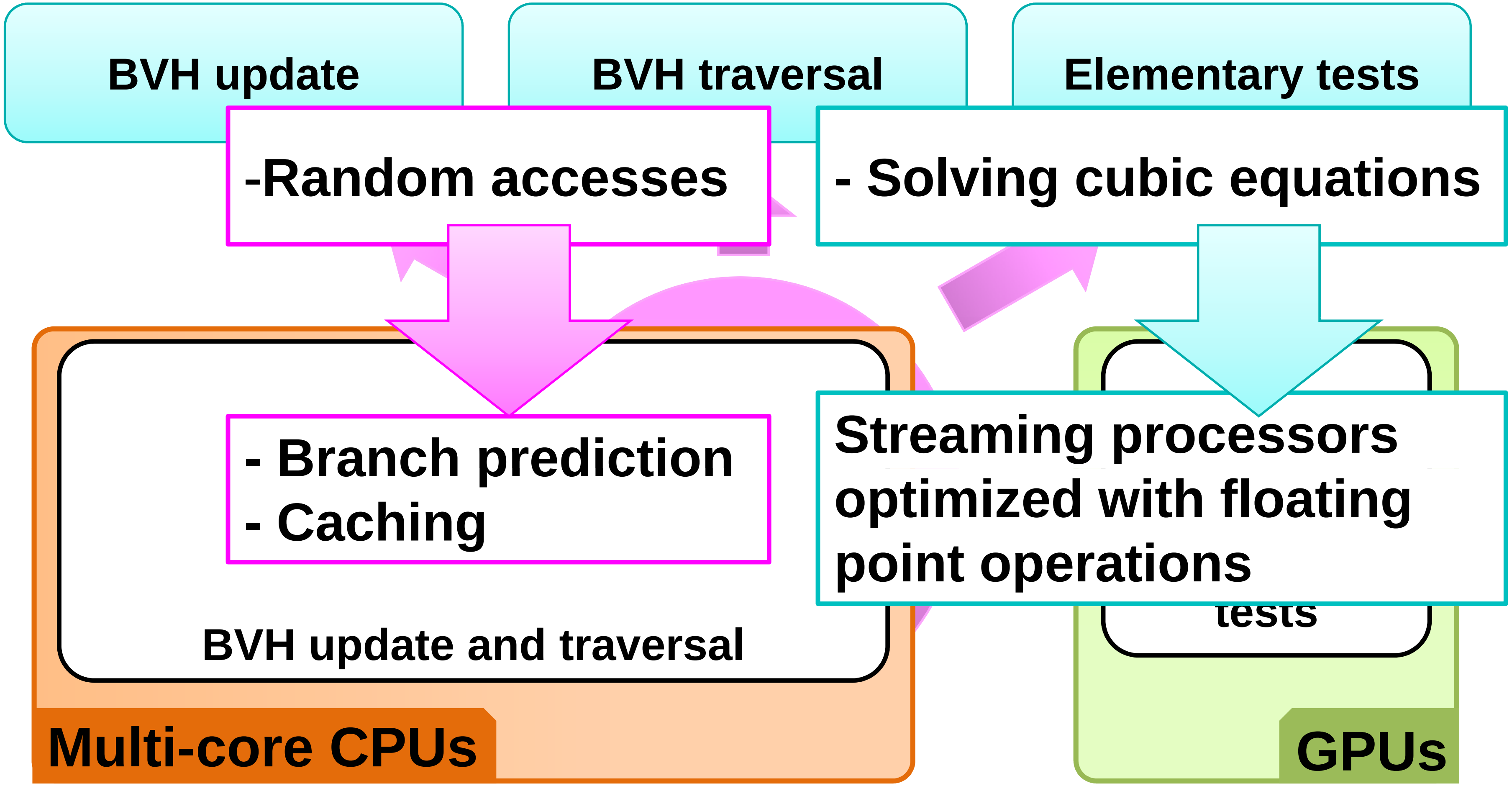
- **HPCCD: Hybrid parallel CD method**
  - **Kim et al., Pacific Graphics 2009**

HPCCD: Hybrid Parallel CCD

- Utilize both multi-core CPUs and GPUs
  - No locking in the main loop of CD
  - GPU-based exact CD between two triangles
- High scalability & interactive performance



Task Distribution

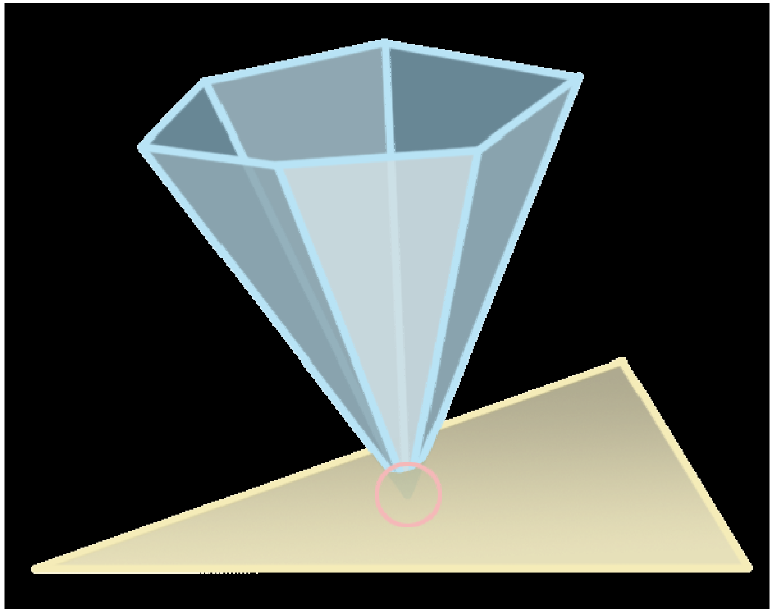


Testing Environment

- **Machine**
  - **One quad-core CPU (Intel i7 CPU, 3.2 GHz)**
  - **Two GPUs (Nvidia Geforce GTX285)**
- **Run eight CPU threads by using Intel's hyper threading technology**

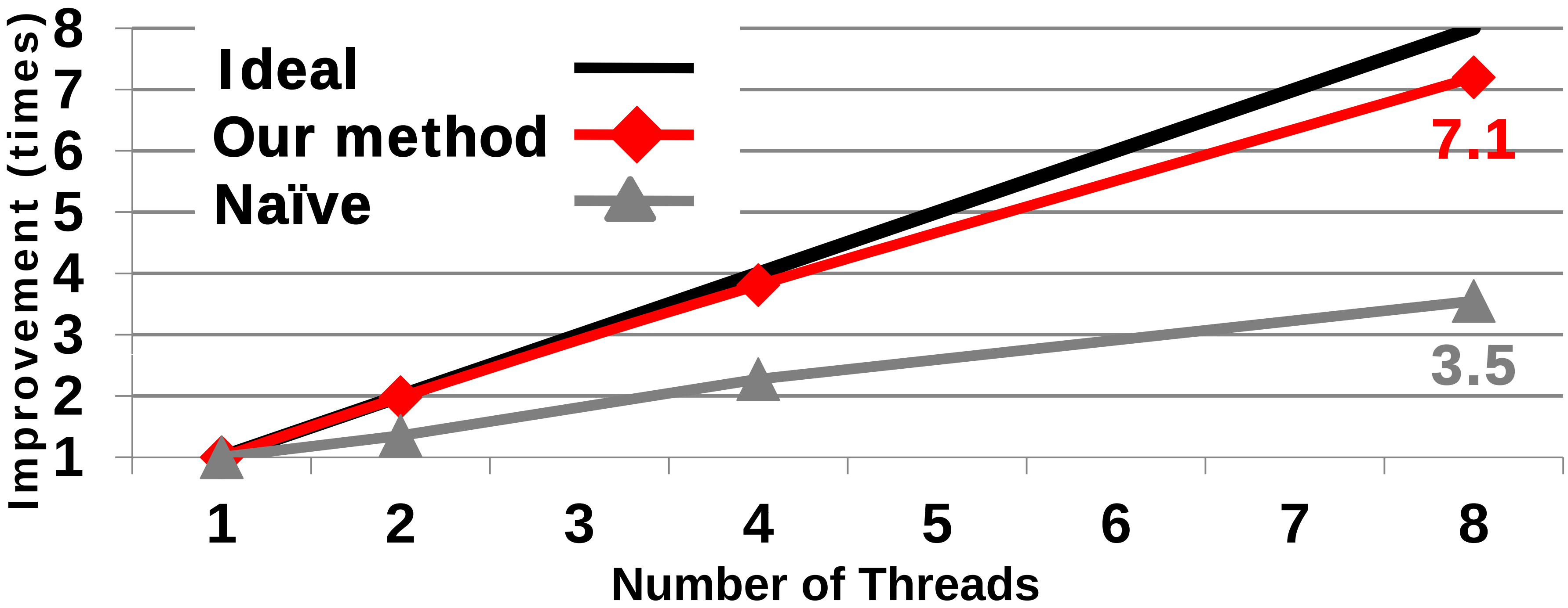
BVH-based CCD

- Axis-aligned bounding boxes
- Feature based BVHs [Curtis et al., I3D 08]
  - Assign each features (e.g., vertex and edge) to each triangle
  - Drastically reduce many redundant tests



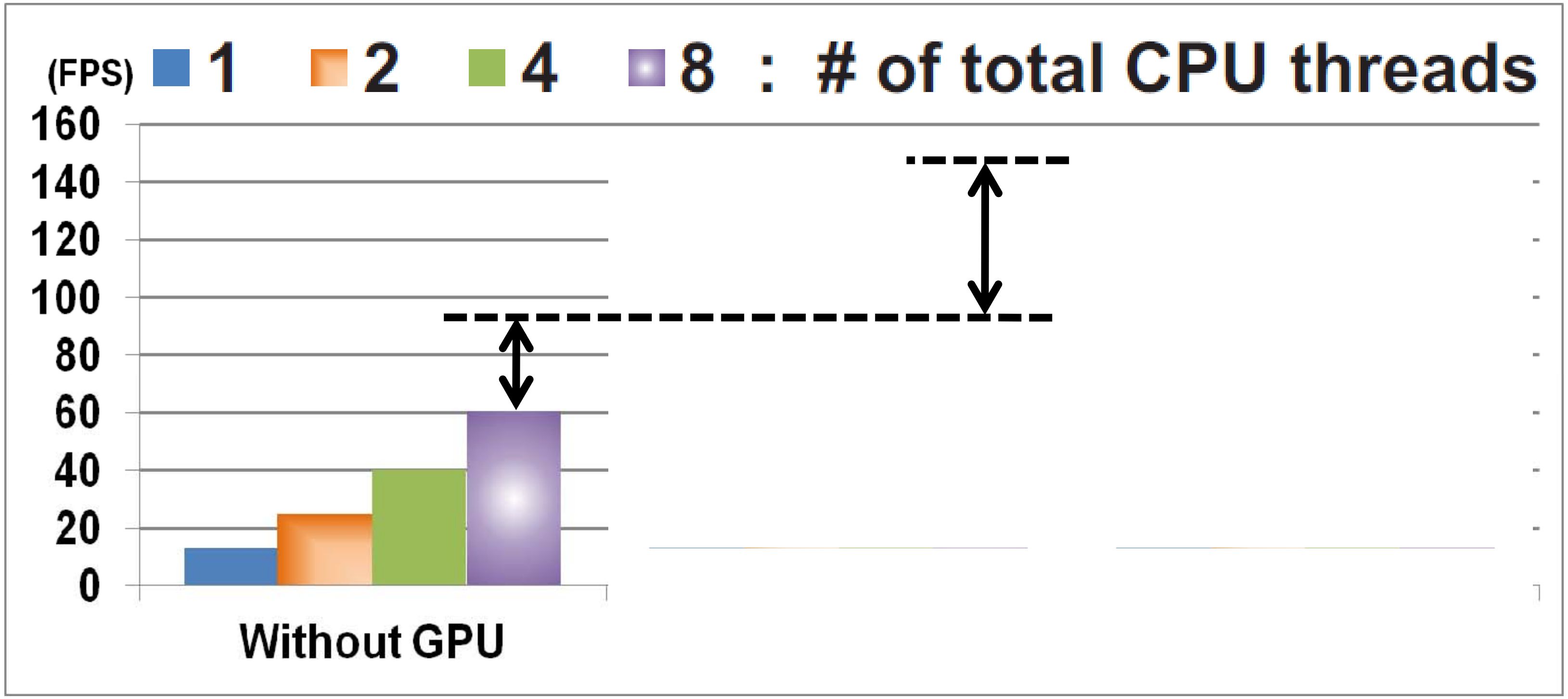
Results

Results of a CPU-based Parallel CCD



- **Remove locking in the main loop of CD**
- **Employ efficient dynamic load-balancing based on inter-CD task units**

Results of HPCCD



- **As the number of GPUs is increased, we get higher performances**

Limitation

- **Low scalability for small rigid models**

Summary

- A **hybrid parallel** algorithm
  - Utilize both multi-core CPUs and GPUs

The implementation code is available as **OpenCCD library (http://sglab.kaist.ac.kr/OpenCCD)**

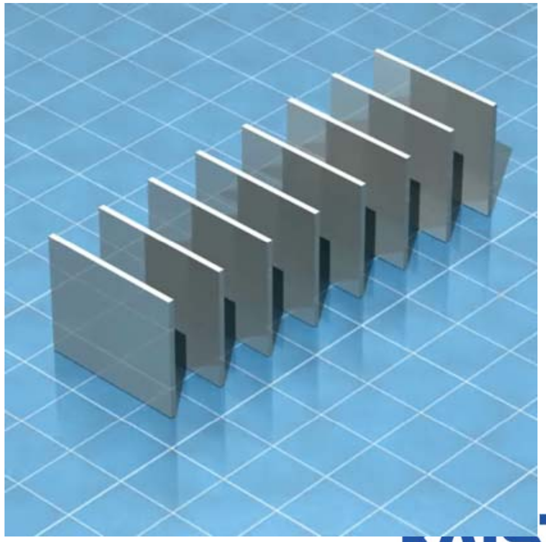
- Interactive performance
  - Show 19-140 FPS for various deformable models consisting of tens or hundreds of thousand triangles

Overview

- **Background**
- **HPCCD: Hybrid parallel continuous collision detection**
- **FASTCD: Fracturing-Aware Stable Collision Detection**

CD for Fracturing Models

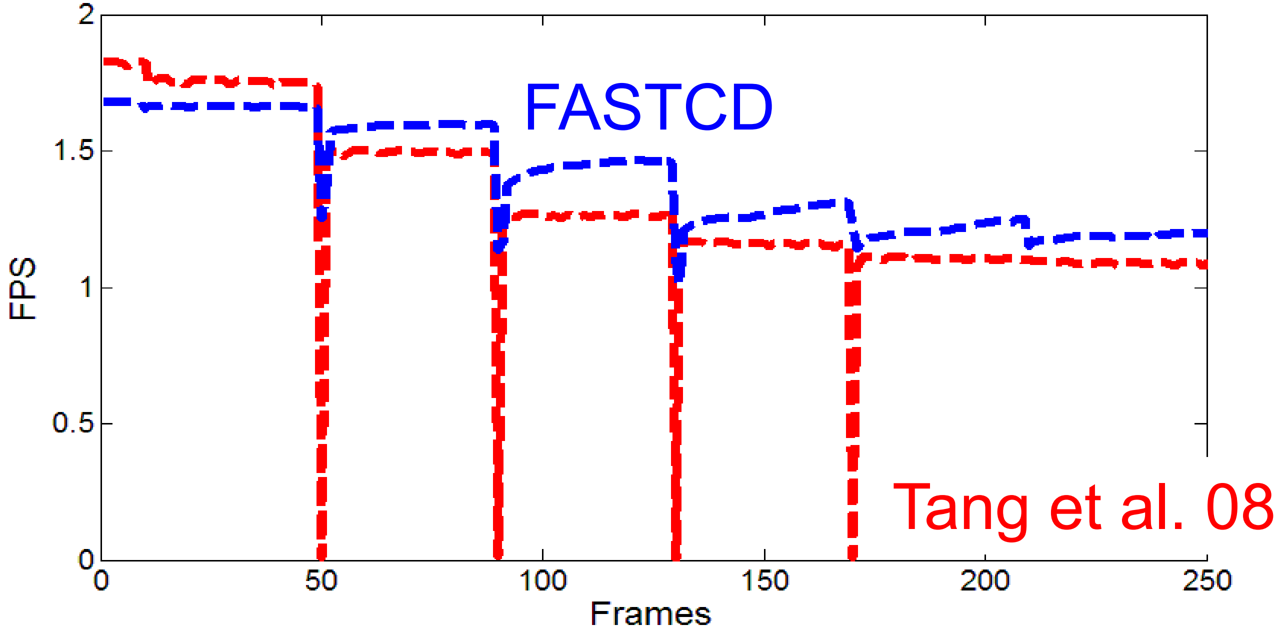
- More widely used in various applications to create more realistic interactions
- Fracturing
  - Changes the connectivity of a mesh: pre-computed hierarchies show low culling ratios
  - Places many objects in close proximity: CD cost is increasing
- Fracturing is one of the most challenging scenarios of collision detection



Our Approach

- FASTCD: Fracturing-Aware Stable CD
  - Incrementally update meshes and BVHs by utilizing topological changes of models
  - Design a simple self-CD culling method without much pre-computations

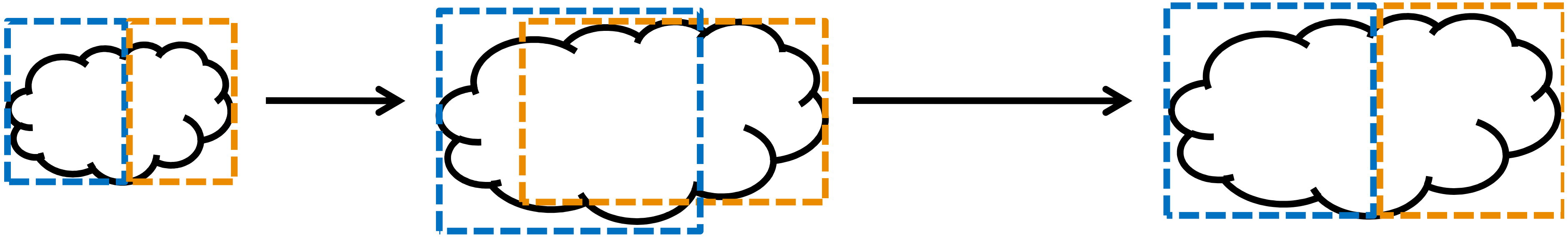
CCD Performance with the Breaking Dragon Model



- **FASTCD shows stable performance**

Selective Restructuring of BVHs

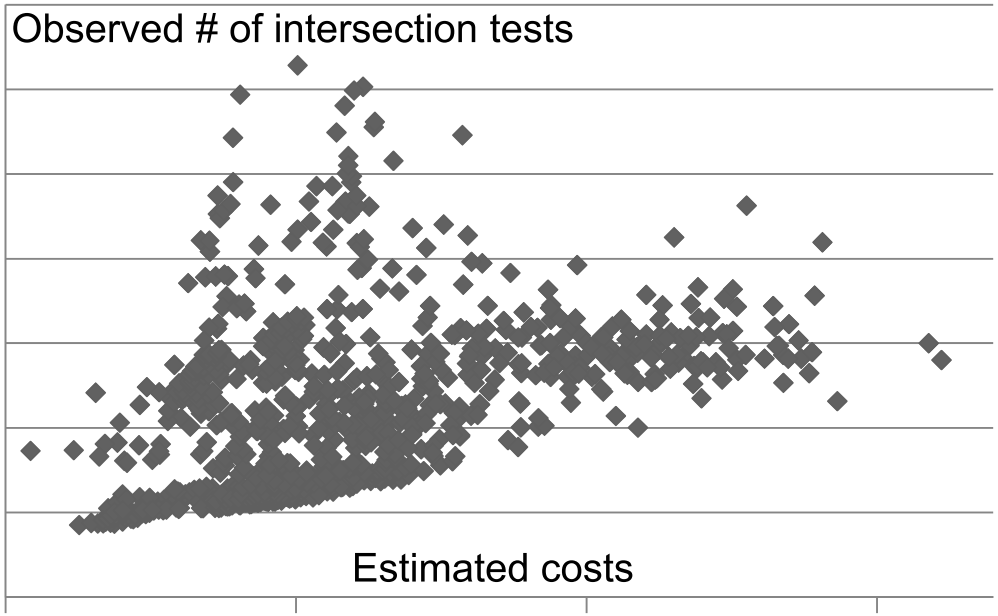
- **As models deform, culling efficiency of their BVHs can be getting lower**
  - **Selective restructuring can address the problem**

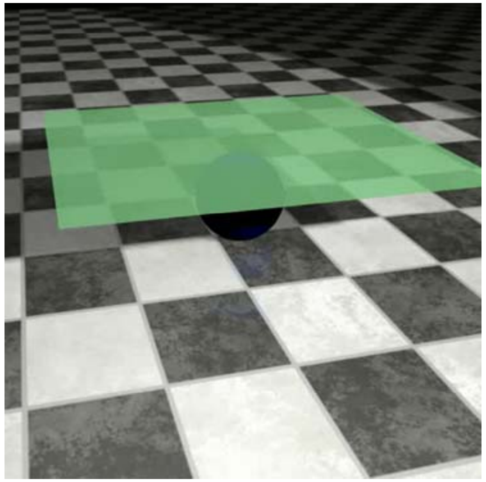


- **How to determine a culling efficiency of a BVH?**
  - **Heuristic metrics have been proposed**
  - **LM metric : [Larsson and Akenine-Möller 2006]**
- **A cost metric that measures the expected number of intersection tests is proposed**

Metric Validation

- **Estimated # of tests vs. Observed # of tests**

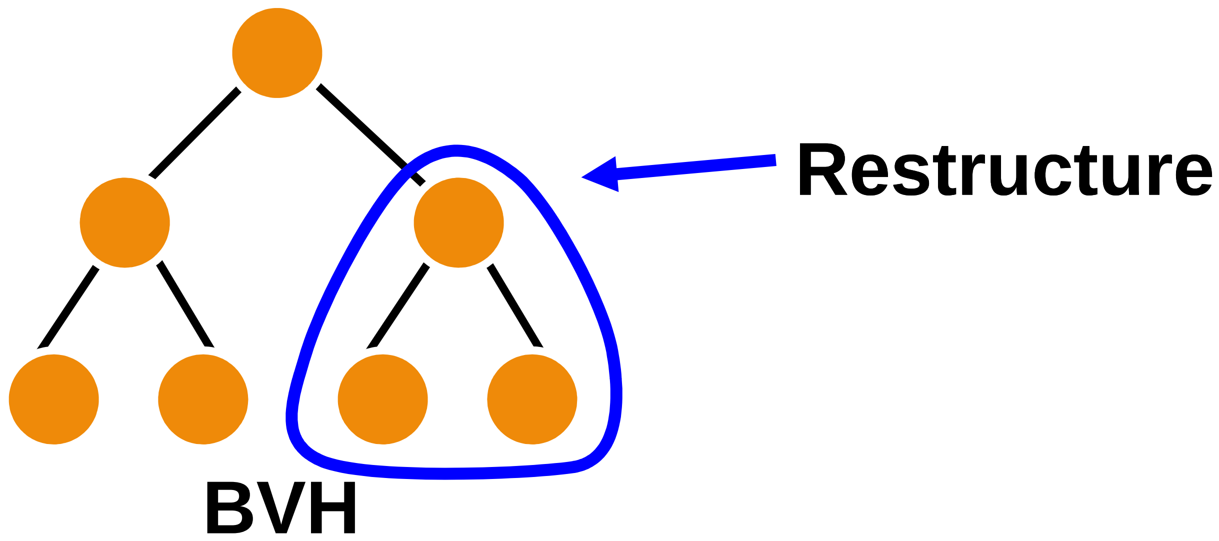




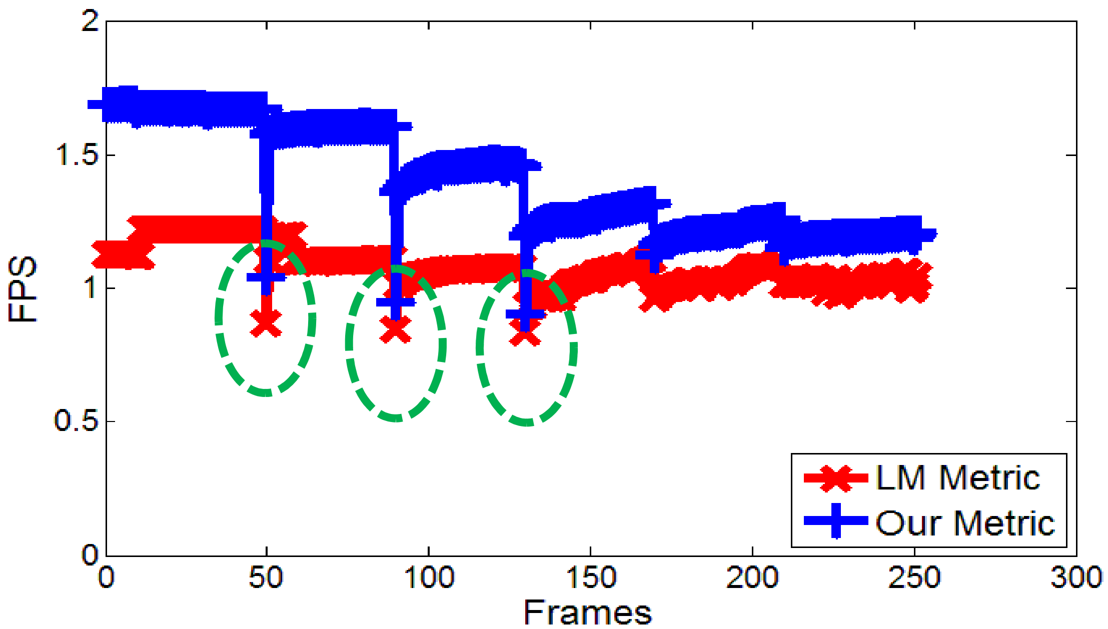
- **Linear Correlation : 0.71**
  - **Tested with various models (0.28 ~ 0.76 , average 0.48)**

BVH Selective Restructuring

- Restructure only subsets of BVHs after refitting BVs



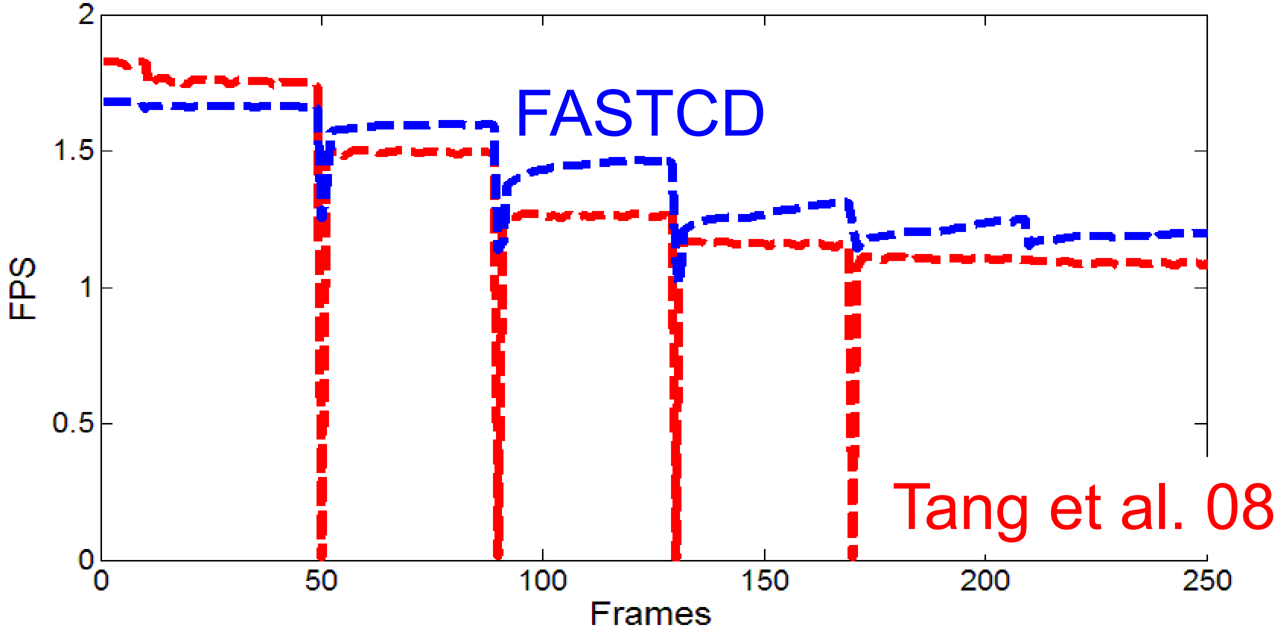
Result of Selective Restructuring



- **LM metric : [Larsson and Akenine-Möller 2006]**
- **Performance degradations at topological changes → unstable**
- **Proposed fast BVH construction methods**

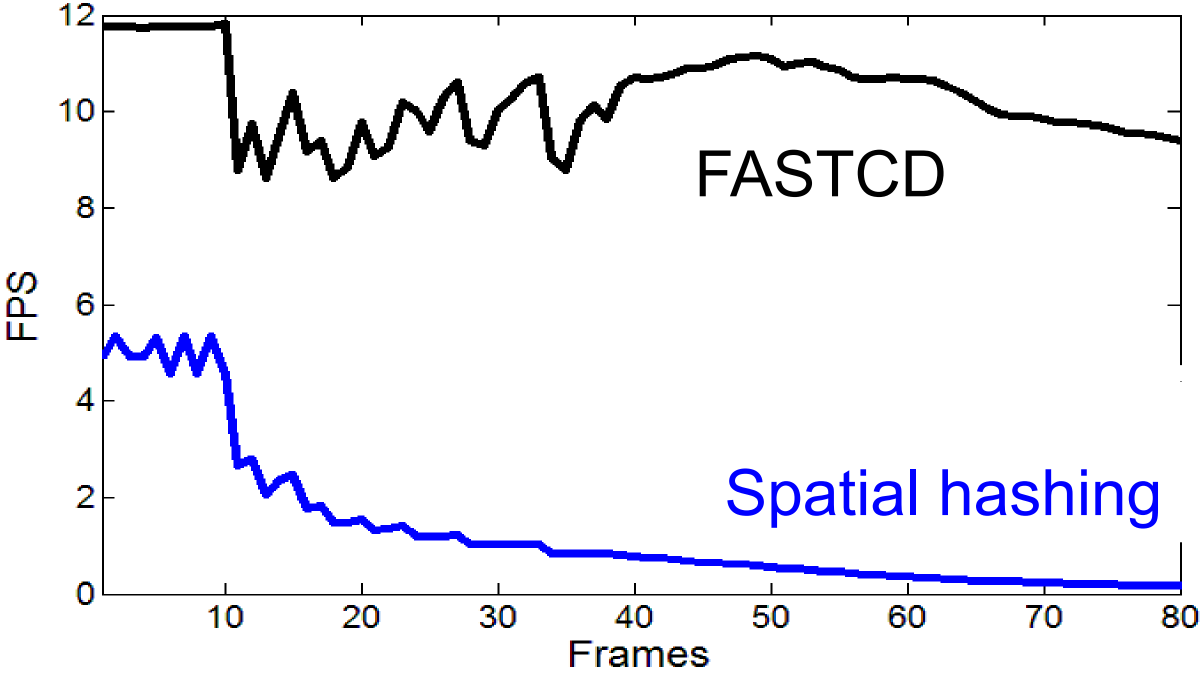
KAIST

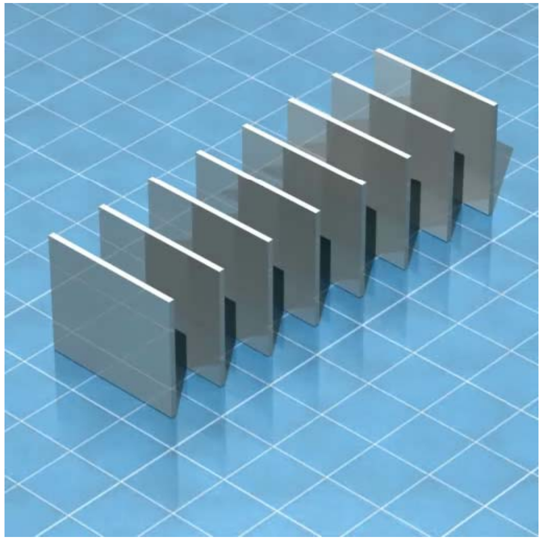
Comparison on CCD



- **FASTCD shows more stable performance**

Comparison (Discrete CD)





- **20x faster than optimized spatial hashing [Teschner et al, 2003]**
- **Stable performance**

Summary

- Presented two recent BVH-based methods for interactive CD among large-scale deforming models
  - HPCCD: Hybrid Parallel Continuous Collision Detection
  - FASTCD: Fracturing-Aware Stable Collision Detection

◆ The code of HPCCD is available as OpenCCD library

◆ Two fracturing models are available (http://sglab.kaist.ac.kr/FASTCD/)

Future Directions

- Various parallel proximity queries and their applications
  - g-Planner (GPU-based motion planner), AAAI 10
  - Hybrid parallel proximity queries, under progress
  - Their applications to time-critical applications (e.g., robot motion planning)

- Volumetric representations
  - VolCCD, Tang et al. 2010, under progress (zoomed view)

Acknowledgements

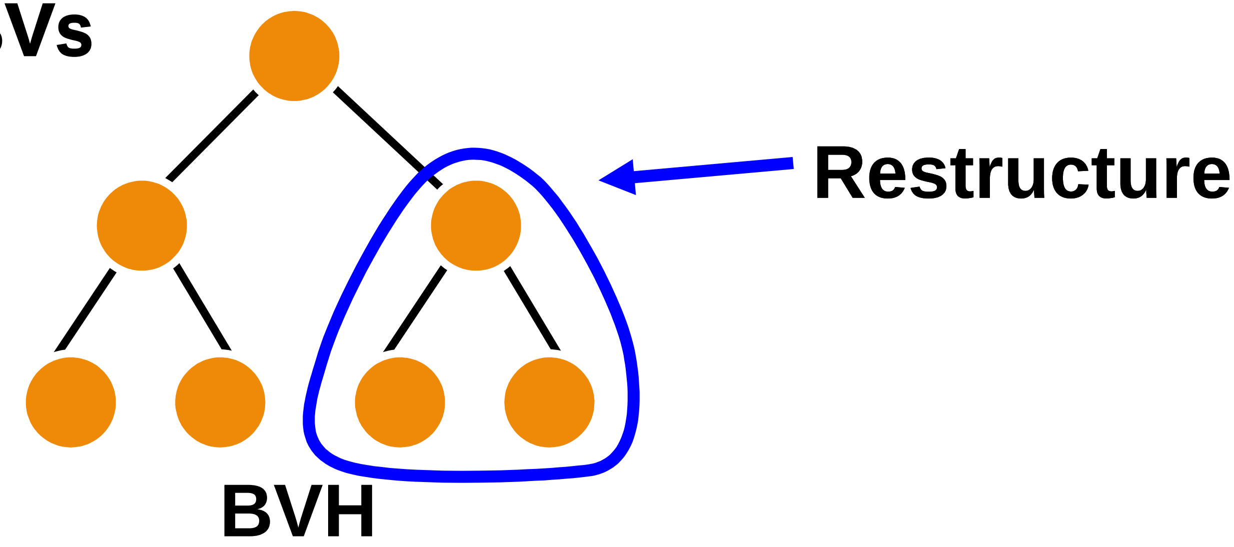
- Research collaborators
  - DukSu Kim, JaePil Heo, John Kim, Miguel Otaduy, Tang Min, JeongMo Hong, JoonKyung Seong, JaeHyuk Heo

- Funding sources
  - Ministry of Knowledge Economy
  - Microsoft Research Asia
  - Samsung
  - Korea Research Foundation

Unused slides

BVH Selective Restructuring

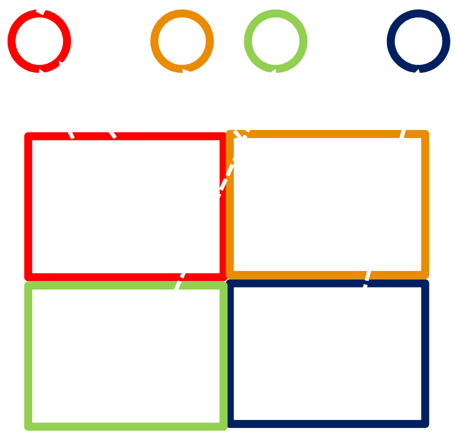
- Restructure only subsets of BVHs after refitting BVs



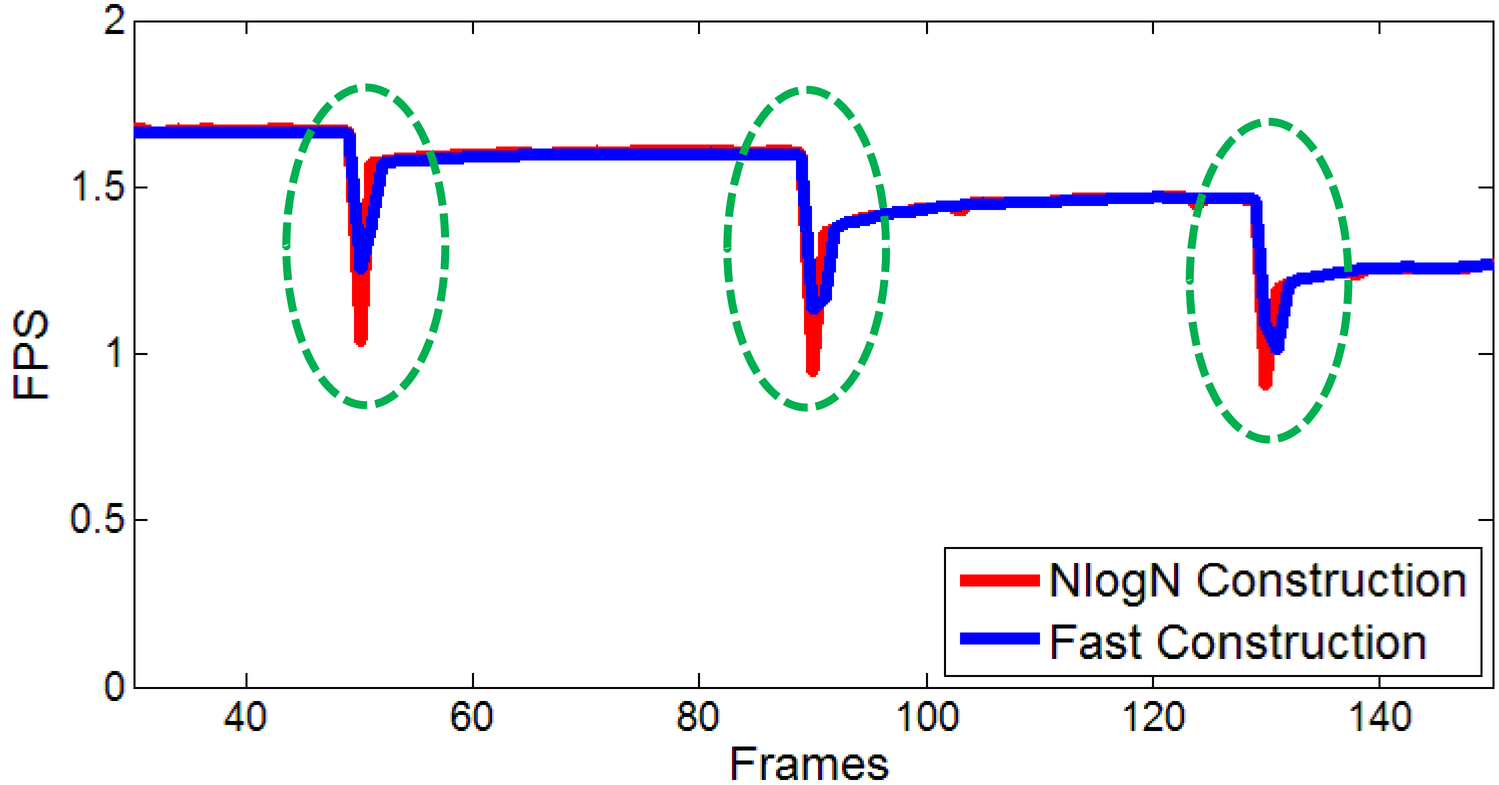
- Requires a metric indentifying such subsets
  - Volume ratios of BVs of parent and child BVs [Zachmann 02, Larsson et al. 06, Yoon et al. 06]

Fast BVH Construction Method

- At a fracturing event, BVHs for fractured parts should be updated for high culling efficiencies
  - Causes noticeable performance degradations
- Propose a BVH construction method based on grid and hashing, instead of typical NlogN methods
- Constructed hierarchies have low culling efficiencies, but use less construction times
  - Improve the overall performance at fracturing events



Result of Fast BVH Construction



- **Performance degradations at fracturing events are reduced**

Background

- BVH-based collision detection
- BVH construction

- Updates BVHs as models deforms
  - Reconstruction from scratch
  - Refitting