Kinetic Separation Lists for Continuous Collision Detection of Deformable Objects

René Weller
Clausthal University, Germany
weller@in.tu-clausthal.de

VRIPHYS ’06, November 2006, Madrid
Motivation

- Environments with dynamically deforming objects play an important role in many applications
  - Medical simulation
  - Animations (Games/Movies)
  - Cloth simulation
CD for Deformable Objects

- Most current techniques use BVHs
- The pre-processed hierarchy becomes invalid when the object deforms
- Problem of adjacency when using BVHs for self collision detection
- Swept volumes for continuous CD
Swept-Volume Continuous CD
Problems

- Discrete time sampling
  - Many update operations/ collision checks
- No adequate use of spatial and temporal coherence
- Other approaches:
  - Restriction of deformation schemes [James and Pai, 2004]
  - Chromatic decompositions [Govindaraju et al. 2005]
  - Kinetic sweep-and-prune-algorithm [Coming, Staadt, 2006]
Our Approach

- Motion in the physical world is normally continuous
- Changes in the combinatorial **structure** of the BHVs and collisions occur only at **discrete time points**
  
  → We store only the combinatorial structure of the BVH and use an event based approach for updates
  
  → We maintain the combinatorial structure of the recursion tree

- Collision detection is reduced to the discrete problem of determining changes in our separation list
Event Based Continuous Collision Detection

Event-Queue
Advantages

- Valid BVHs and separation list at every point in time
- Independent of query sampling frequency
- Collisions are reported in the right order
- Can handle all kinds of objects
  - polygon soups, point clouds, and NURBS models
- Can handle insertions/deletions during run-time
- Inter-object and self-collision detection
- Can handle all kinds of deformations
  - Only a flightplan is required for every vertex
  - These flightplans may change during simulation
Recap: Kinetic AABB Tree

- Kinetization of the AABB tree
- Pre-processing: Build the tree by any algorithm suitable for static AABB trees
- Store with every node the indices of those points that determine the BV
Recap: Kinetic AABB Updates

Motivation

Recap

Kinetic Separation List

Results

Conclusions
Recap: KDS terminology

- **KDS** are a framework for designing and analyzing algorithms for objects in motion [Basch et al. 1997]
- KDS framework leads to event-based algorithms that samples the state of parts of a system only as often as necessary for a special task (e.g. a bounding box)
- The task is called the **attribute**
- A KDS consists of **certificates**
- Certificate failures are called **events**
- If the attribute changes at the time of an event, the event is called **external**, otherwise **internal**
Kinetic Separation List

- Kinetic AABB tree utilizes coherence only for updates
- Kinetic separation list uses event-based approach also for collision detection
  - Between pairs of objects
  - Self-collision detection
- Kinetization of the „moving front“ algorithm
Definition/Initialization of the Separation List

- Separation list contains highest non-overlapping BVs and overlapping leaves
Initializing events: BVs overlap
Initializing events: Fathers do not overlap
Simulation Loop

while simulation runs

determine time t of next rendering
e ← min event in event queue

while e.timestamp < t

processEvent(e)

e ← min event in event queue

render scene
Event-Handling: BVs overlap

Separation-List

... 

B 2 --- C 2 --- B 3 --- C 3
Event-Handling: Fathers do not overlap
Event-Handling: Topology of BVs change

Separation-List

... A 1 ...
Quality of a KDS

- A KDS is **compact**, if it requires only little space
- A KDS is **responsive** if we can update it quickly in case of a certificate failure
- A KDS is **local**, if one object is involved in not too many events
- A KDS is **efficient**, if the overhead of internal events with respect to external events is reasonable
Analysis

- **Worst case:**
  - Theorem 1: Our kinetic separation list is compact (O(n^2)), local (O(n)), responsive (O(1)) and efficient. Furthermore, the kinetic separation list is valid at every point of time.

- **Average Case:**
  - Theorem 2: Our kinetic separation list is compact (O(n)), local (O(1)), responsive (O(1)) and efficient.
Test Scenes
Results

- Time for updates and collision check

![Graph showing time for updates and collision check](image-url)
Results

- Self Collision

![Graph showing results for Self Collision]
Results
Conclusions

- A novel data structures for inter- and intra-collision detection between deformable object
- Efficiency due to event based approach
- Well suited for collision response
- Up to 50 times faster than swept volume approach in practically relevant scenarios
Future Work

- Use our kinetic data structures also for other kinds of primitives like NURBS
- Utilize our data structures for other kinds of motion
  - physically-based simulations
  - other animation schemes
Acknowledgements

- Gabriel Zachmann, Clausthal University of Technology
- Johannes Mezger, University of Tübingen
- Stefan Kimmerle, University of Tübingen
- DFG grant ZA 292/1-1 ("Aktionsplan Informatik")