

Virtual Reality in Assembly Simulation

Collision Detection,
Simulation Algorithms, and
Interaction Techniques



Contents

1. Motivation
2. Simulation and description of virtual environments
3. Interaction with virtual environments
4. Collision detection
5. Conclusions

Introduction

Simulation

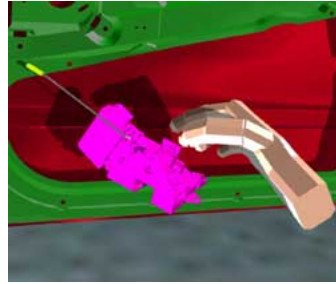
Interaction

Collision Detection

Conclusion

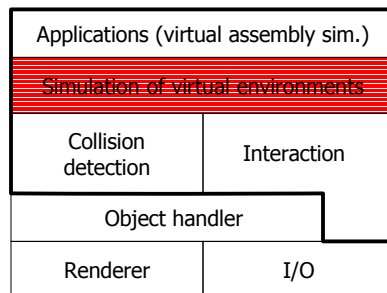
Motivation

- Assembly simulation:
 - One of the most important applications in virtual prototyping
 - One of the biggest challenges for VR technology



Introduction Simulation Interaction Collision Detection Conclusion

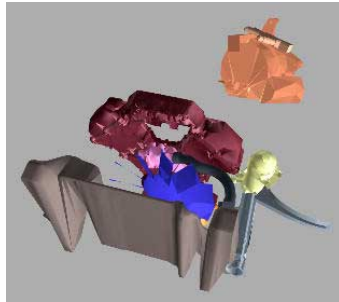
Overview



Introduction Simulation Interaction Collision Detection Conclusion



Simulation of virtual environments



Introduction

Simulation

Interaction

Collision Detection

Conclusion



Requirements

- Event-based
- No programming
- Easy to learn for non-programmers
- Description independent from scene graph

Introduction

Simulation

Interaction

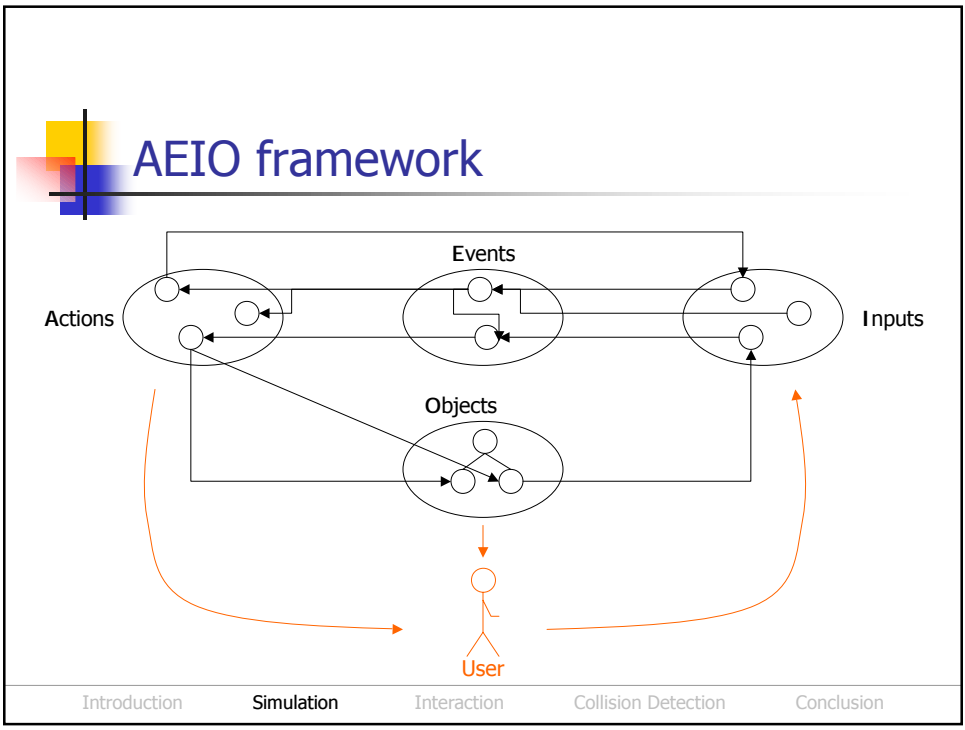
Collision Detection

Conclusion

Other VR Systems

- Academic:
 - DIVE
 - Minimal Reality
- Commercial:
 - Division
 - Sense8

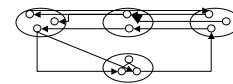
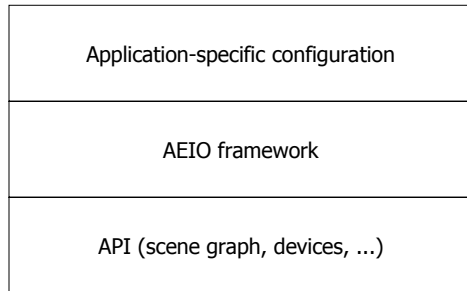
Introduction **Simulation** Interaction Collision Detection Conclusion





Three layers of description

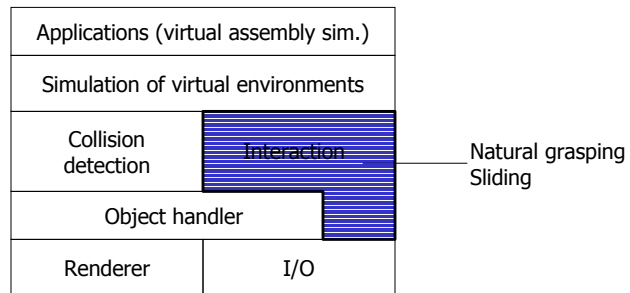
- Layered software architecture



```
trGetTranslation( ... );  
trSetMatrix( ... );
```



Interaction in Virtual Environments





Natural object manipulation

- Types of grasping:
 - Precision grasping
 - 3-point pinch grasping
 - Power grasping



Introduction

Simulation

Interaction

Collision Detection

Conclusion



Natural grasping algorithm

- Kinematic chain induces dependency among joints
- Position of fingers determined by (M,F)
- Iterative interpolation
- Invisible hand for collision detection
- Analysis of contact for grasping

Introduction

Simulation

Interaction

Collision Detection

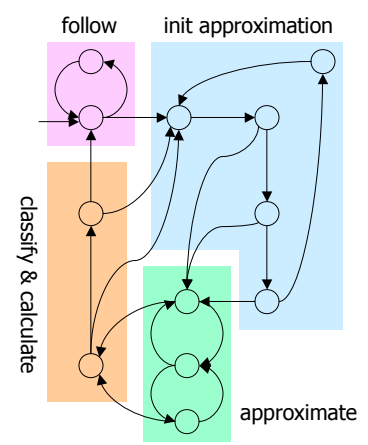
Conclusion

Sliding

- Frequent task: collision-free assembly path
- Sliding simulation
- Problem: no force-feedback
- Different grasping metaphor: "rubber-band"
- Algorithm = FSM (concurrent)
- Approximate contact point
- Calculate new velocities

Sliding algorithm

1. Approximate exact contact point
 1. init
 2. approximate
2. Classify contact
3. Calculate contact normals
4. Calculate new velocities
5. Check stop conditions

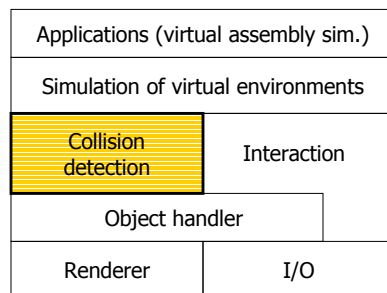


Sliding Video



Introduction Simulation **Interaction** Collision Detection Conclusion

Collision Detection



Introduction Simulation Interaction **Collision Detection** Conclusion



Requirements

- Large class of input geometry
- Real-time
- Additional data structures:
 - Small footprint
 - Construction at startup

Introduction

Simulation

Interaction

Collision Detection

Conclusion



Issues with collision detection

1. Reduction of polygon-polygon tests
2. Reduction of object-object tests
3. Integration in VR system

Introduction

Simulation

Interaction

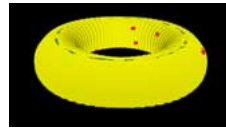
Collision Detection

Conclusion



Hierarchical collision detection

- Strategy:
Divide & Conquer
- Data structure:
Bounding volume hierarchy
- General scheme:
simultaneous traversal



Introduction

Simulation

Interaction

Collision Detection

Conclusion



Boxtree

- Bounding volumes = boxes
- Re-alignment during traversal
- Properties:
 - Split along one axis
 - Re-use computations
- Costs:
 - Node overlap test faster than any other hierarchy
 - Less memory per node than any other hierarchy

Introduction

Simulation

Interaction

Collision Detection

Conclusion

Construction of BV hierarchies

- Speed of collision detection \sim quality of BV hierarchy
- Criteria:
 - Balance depth & number of polygons
 - Minimize volume of overlap
 - Minimize individual volume of BVs
- Strategies:
 - Insertion
 - Bottom-up
 - Top-down

Introduction
Simulation
Interaction
Collision Detection
Conclusion

Constructing Boxtrees

- Algorithm:
 1. Decision: bisection or splitting off empty bbox
 2. Find optimal splitting axis
 3. Distribute polygons
- Performance:

| Polygons | hyperboloid (millisec) | sphere (millisec) |
|----------|------------------------|-------------------|
| 0 | 0 | 0 |
| 2500 | 250 | 250 |
| 5000 | 500 | 500 |
| 7500 | 750 | 750 |
| 10000 | 1000 | 1000 |
| 12500 | 1250 | 1250 |
| 15000 | 1500 | 1500 |
| 17500 | 1750 | 1750 |
| 20000 | 1900 | 2000 |

Introduction
Simulation
Interaction
Collision Detection
Conclusion



DOP-trees

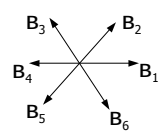
- Generalization of bounding boxes

- Definition:

$$B = \{B_1, \dots, B_k\}, \quad B_i \in \mathbb{R}^3, \quad \|B_i\| = 1, \quad B \text{ fixed}$$

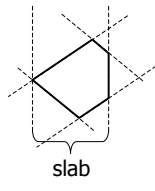
$B_i, B_{i+k/2}$ antiparallel

$$\text{DOP } D = \bigcap_{i=1}^k H_i, \quad H_i : B_i \cdot x - d_i \leq 0$$



- Representation:

$$D = (d_1, \dots, d_k) \in \mathbb{R}^k$$



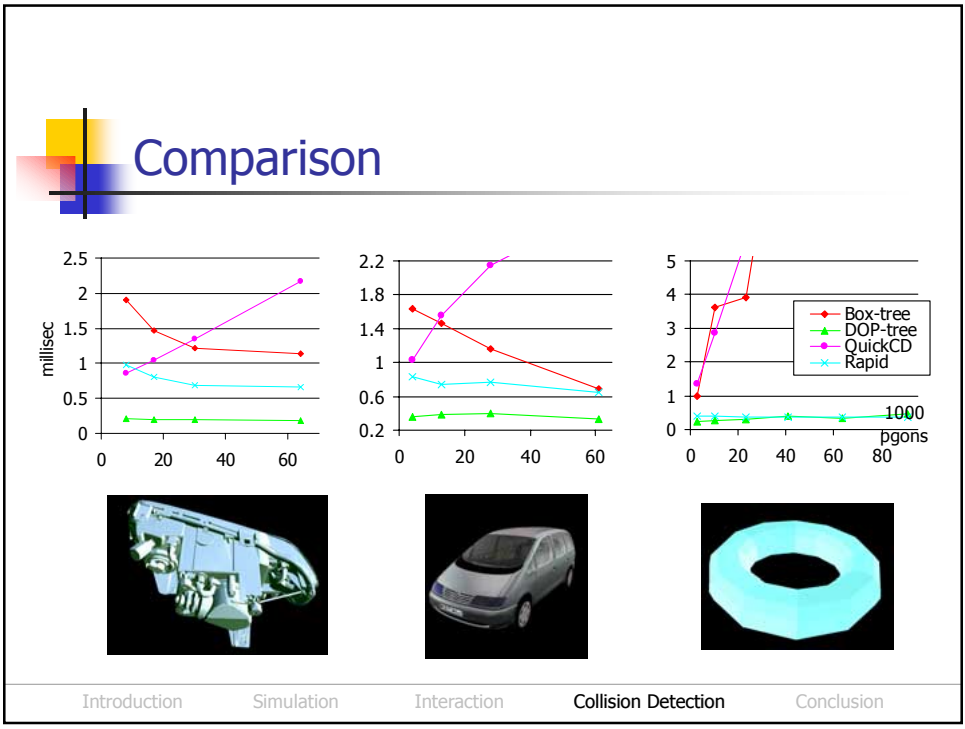
Overlap test of DOPs

- Algorithm for "tumbled" DOPs:

- Object motion (M, o)
- Affine transformation:

$$d_i' = B_i \begin{pmatrix} b_{j_1} \\ b_{j_2} \\ b_{j_3} \end{pmatrix}^{-1} \begin{pmatrix} d_{j_1} \\ d_{j_2} \\ d_{j_3} \end{pmatrix} + B_i o, \quad b_j = B_j M^{-1}$$

- Correspondence j_i identical for all DOPs of a DOP-tree
- Cost: $O(k)$, previously $O(k^2)$



- ## Convex polytopes
- Useful in the collision detection pipeline
 - Linear separability
 - Algorithm:
 - Perceptron learning rule
 - Simulated annealing
 - Hill climbing
 - Incremental
 - Probabilistic (biased towards "collision")
- Introduction Simulation Interaction Collision Detection Conclusion



Separating planes

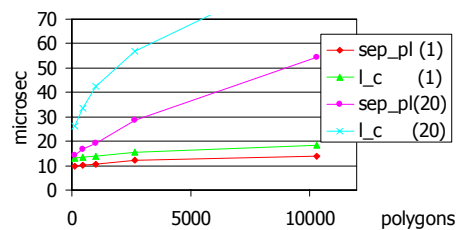
- Linear separability:
 P, Q linearly separable \Leftrightarrow
 $\exists w \in \mathbb{R}^3, w_0 \in \mathbb{R}: \forall i: p^i \cdot w - w_0 > 0, \forall j: q^j \cdot w - w_0 < 0$
 $\exists w \in \mathbb{R}^3, w_0 \in \mathbb{R}: \forall i: (p^i, -1) \cdot (w, w_0) > 0, \forall j: (-q^j, 1) \cdot (w, w_0) > 0$
- Algorithm:
 - Perceptron learning rule
 - Simulated annealing
 - Hill climbing
 - Incremental
 - Probabilistic (biased towards "collision")

Introduction Simulation Interaction Collision Detection Conclusion



Results

- ~2x faster than Lin-Canny
- Sub-linear in the number of polygons
- Less dependency on rotational speed
- Numerically more robust than Lin-Canny
- Less memory requirements



Introduction Simulation Interaction Collision Detection Conclusion



The object level

- Multi-body problem: $n^2/2$ BV tests
- Space-indexing is more efficient if

$$P_s(n) < \frac{n^2}{2} - \frac{T_s(n)}{T_b}$$

- Grid and octree:
 - Incremental
 - Optimal number of cells
 - Grids are more efficient

Introduction

Simulation

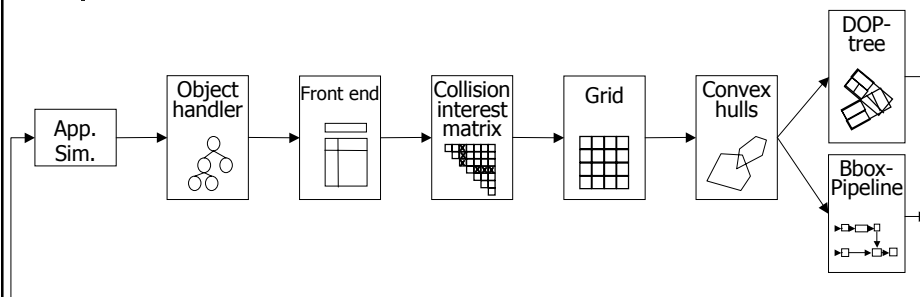
Interaction

Collision Detection

Conclusion



Collision detection pipeline



Introduction

Simulation

Interaction

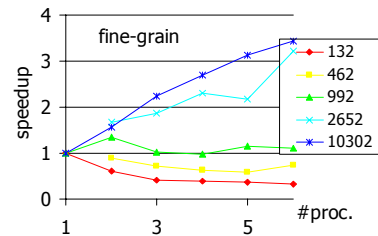
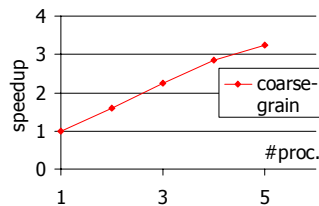
Collision Detection

Conclusion



Parallelization

- Types:
 - Coarse-grain (object level)
 - Fine-grain (polygon level)



Introduction Simulation Interaction Collision Detection Conclusion



Collision detection results

- Incremental algorithm for convex objects:
 - 10,000 polygons \approx 12 microseconds
- Enclosing DOPs in $O(k)$, previously $O(k^2)$
- Hierarchical algorithms:
 - 50,000 polygons \approx 1 millisecond
- Collision detection pipeline,
 - parallelized and concurrent
- Integrated in *Virtual Design 2*

Introduction Simulation Interaction Collision Detection Conclusion



Conclusions

- Framework for simulation of virtual environments
- Interaction:
 - Natural grasping
 - Sliding simulation
- Collision detection:
 - New hierarchical and convex algorithms
 - Collision pipeline
 - Parallelization
- Virtual assembly simulation application

Introduction

Simulation

Interaction

Collision Detection

Conclusions



Future Work

- Collision detection:
 - Increasing demands on collision detection
 - Collision detection for flexible objects
 - Local criterion for global optimization?
 - Incremental algos for polygon soups
 - Estimation of penetration depth?
 - Collision detection with max. allowed penetration depth
 - Combination of C-Space and DOP-trees?
 - CD in hardware
 - Subdivision surfaces


Introduction

Simulation

Interaction

Collision Detection


Conclusion



Future work

- Natural grasping:
 - Simulation of finger tissue for precise manipulation
 - Complex manipulations ("twiddling")
 - Deformable hand
 - Sliding of hand
- VR without Devices
 - Virtual dataglove
 - Computer-vision based head-tracking
 - Body tracking

Introduction Simulation Interaction Collision Detection Conclusions



Future work

- Interaction in VR
 - Selection (dense and crowded environments)
 - Navigation and interaction in virtual cities
- Experimentation in VR
 - Relativistic VR
 - Non-physical laws (r^2 instead of r^3)
 - Non-euclidian geometry

Introduction Simulation Interaction Collision Detection Conclusion



Thankx.

Introduction Simulation Interaction Collision Detection **Conclusions**