A Benchmarking Suite for 6-DOF Real Time Collision Response Algorithms

David Mainzer, Rene Weller, Gabriel Zachmann

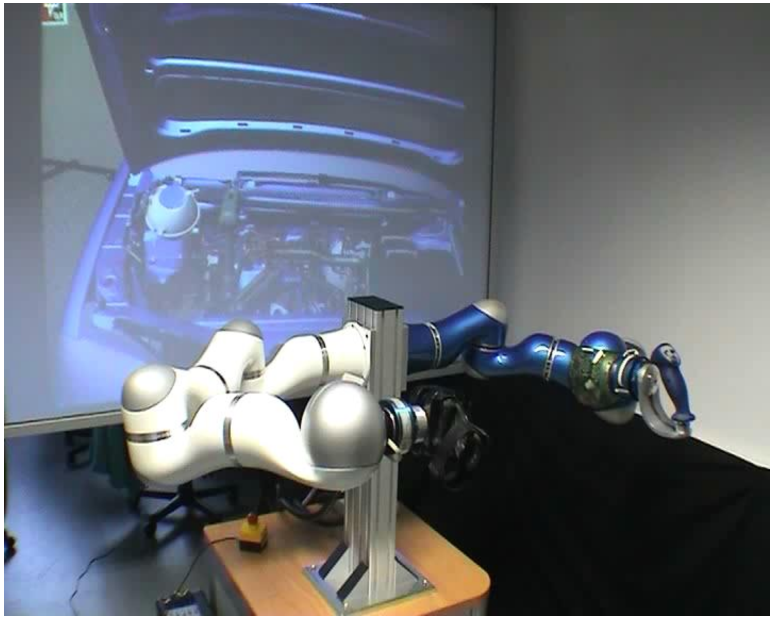
Clausthal University, Germany

Mikel Sagardia, Thomas Hulin, Carsten Preusche

German Aerospace Center (DLR), Germany

VRST 2010, Nov 2010, Hong Kong

Motivation for Collision Detection



- Make virtual environments more realistic
- Basic component of video games, robotic, medical applications

Motivation for Coll.-Det. Benchmark

- Many collision detection libraries exist
  - Different data structures and/or different penetration measures
  - Difficult to compare
- Human perception is very sensitive with forces [Kim et al. 2002]
- Visual and tactical sensations are treated together in a single attentional mechanism ⇒ mismatch can affect suspension of disbelief [Spence & Driver 2000]
- Need stable and continuous forces and torques, even in extreme situations (high impact velocities or large contact areas)
- Force-feedback requires a constant update rate of 1000 Hz
  ⇒ collision detection must be very fast

Previous Work

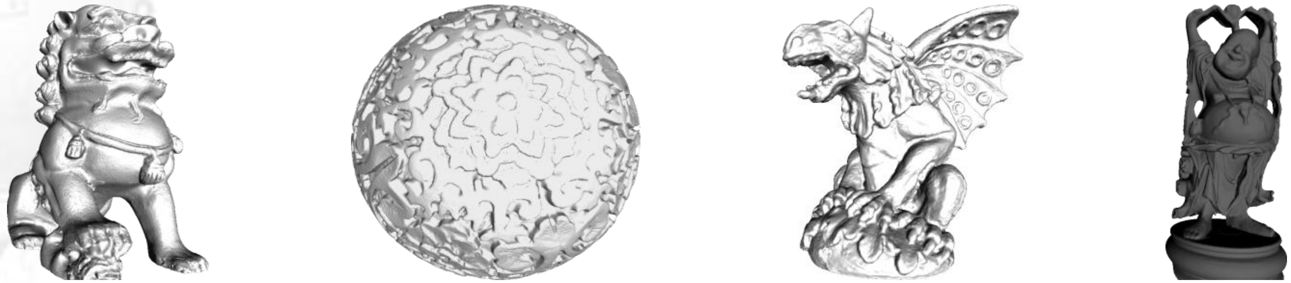
- Collision detection within context of motion planning for rigid and articulated robots in 3D workspace [Caselli et al. 2002]
  - Not of general utility and restricted to fixed set of scenarios
- 3-DOF point-based benchmark [Cao 2006]
  - Attached collision detection libraries to emulated 3-DOF point based haptic device
  - Only suitable for haptic algorithms
- Ground truth data set for haptic rendering [Ruffaldi et al. 2006]
  - Only single point of contact
- Benchmarking suite for collision detection algorithm [Trenkel et al. 2007]
  - Only distance, no comparison of expected and computed response

Contribution

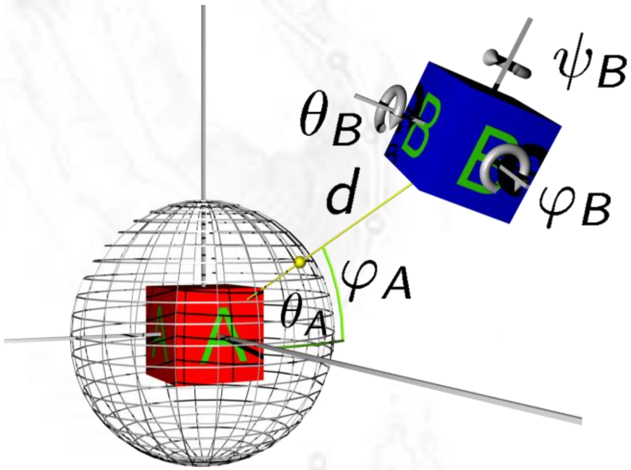
- Our Benchmarking Suite:
  1. Performance benchmark for collision detection algorithms
  2. Evaluation methodology for force and torque quality
     - Analyzes magnitude & direction values with respect to contact models
     - Noise in signals
- Evaluation
  - Compare two rather different collision detection algorithms

Part 1: Performance Benchmark

- Cover a wide variety of different, highly detailed objects e.g.:



- Move objects in vast number of different configurations and perform a collision detection test

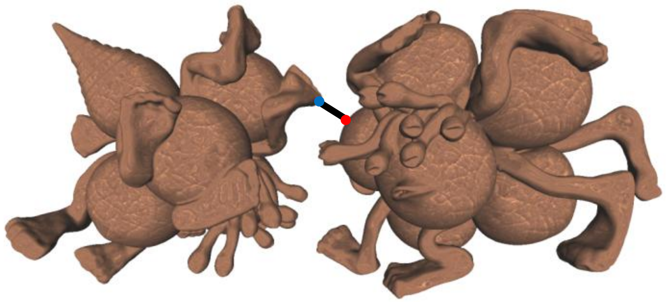


- One configuration consists of 6 parameters:
  - Translation of object B in the coordinate system of object A, given by d, φ_A, θ_A
  - Rotation of object B, given by $\varphi_B, \theta_B, \psi_B$

Performance Benchmark scenarios

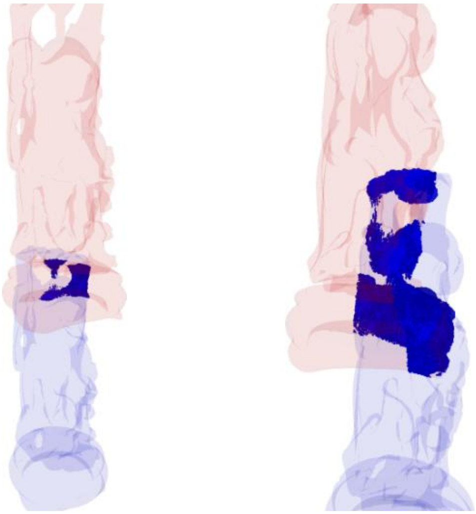
Scenario I

- Situations where objects are in close proximity, but not touching



Scenario II

- Situations where two objects intersect (from light to heavy interpenetration)



Goal:

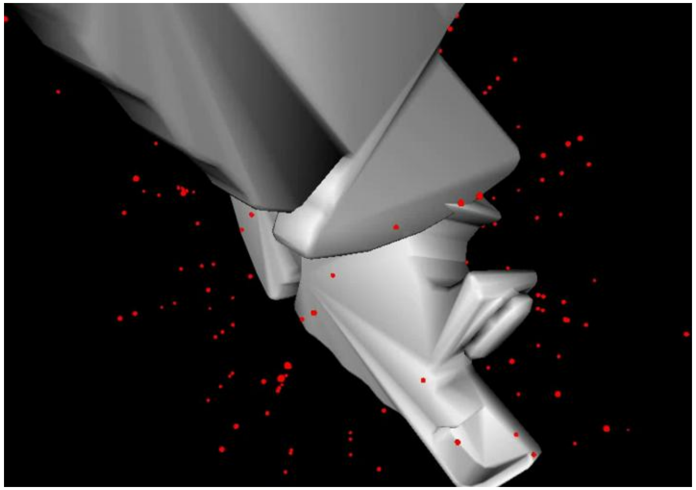
- Max and avg collision detection time
  - Sample configuration space densely

Scenario I

- Scenario I (no intersection)
  - Keep distance d fixed
  - $\Delta\varphi_A = \Delta\theta_A = 15^\circ$ and
    $\Delta\varphi_B = \Delta\theta_B = \Delta\psi_B = 15^\circ$
  - Generate 2M sample configurations for each distance
  - Compute sample configurations for distance from 0% up to 30% of object size (1% steps)

Scenario II

- Scenario II (intersection)
  - Keep intersection volume fixed
  - $\Delta\varphi_A = \Delta\theta_A = 15^\circ$ and $\Delta\varphi_B = \Delta\theta_B = \Delta\psi_B = 30^\circ$
  - For every intersection volume: 270K sample configurations
  - Sample configurations for intersection volume from 0% up to 10% of the total fixed object volume (1% steps)
- Used PC cluster with 25 cluster nodes, each with 4 Intel Xeon CPUs with 16GB of RAM
- 5 600 CPU days = 86 objects



Benchmarking procedure

Main steps:

1. Load the set of configurations for one object
2. For each object-object distance/intersection volume, start timing, set the transformation matrix of the moving object and perform a collision test
3. Get a max and avg collision detection time

- Overall we performed 65 million different collision detection tests with one collision library

Part 2: Quality Benchmark

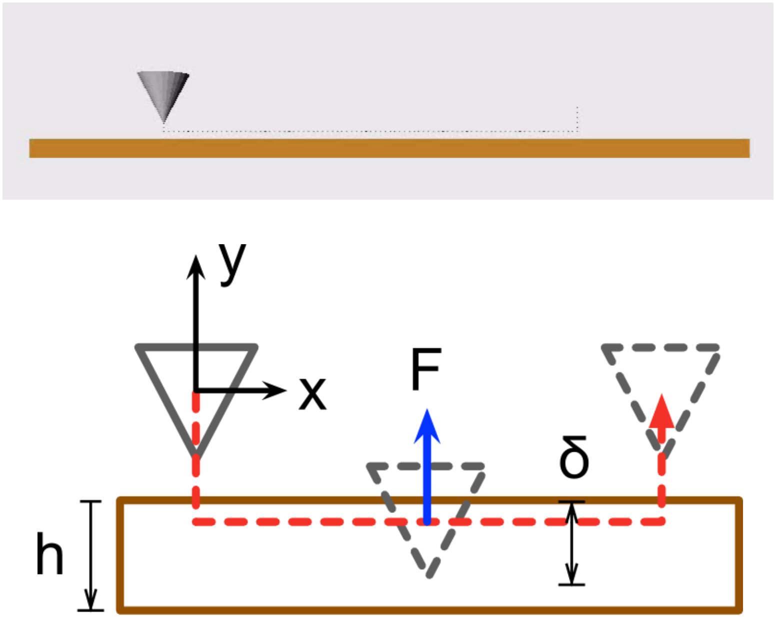
Scenarios in this benchmark should meet two requirements:

- Simple enough so that it is possible to provide an analytical model
- Suitable abstraction of the most common contact configurations in force feedback or physically-based simulations

Quality Scenario I

Reasons for this scenario:

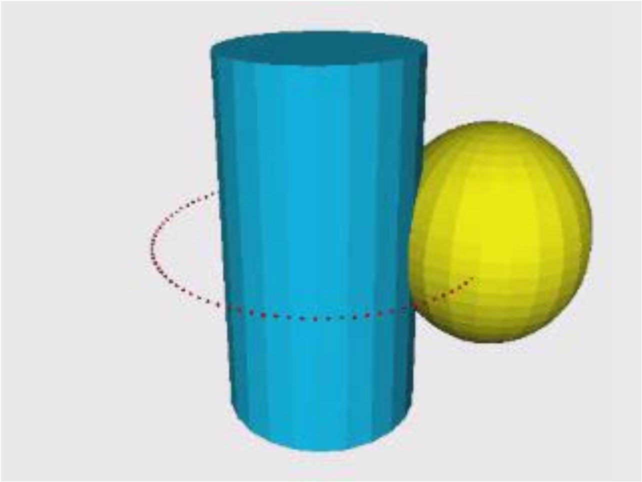
- Evaluation of behavior with flat surfaces or sharp corners
- Evaluates how algorithms handle the *tunneling effect* (h → 0)



Analytical (ideal) model:

- Expected direction of F: +y; no torques
- F = const, while cone slides on the block

Quality Scenario II

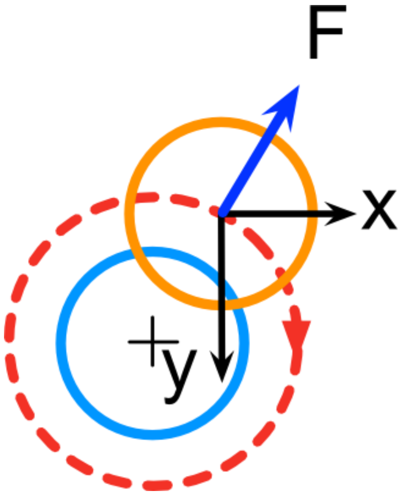


Reasons for this scenario:

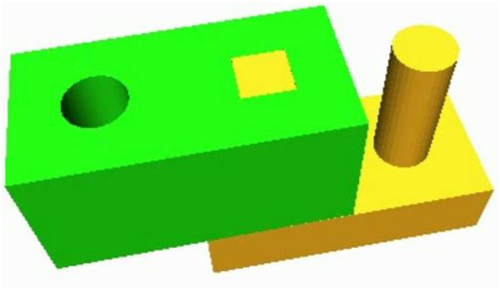
- Evaluation of behavior with smooth rounded surfaces

Analytical (ideal) model

- Expected direction of F: from cylinder center to sphere center; no torques
- |F| = const, while sphere revolves around cylinder



Quality Scenario III

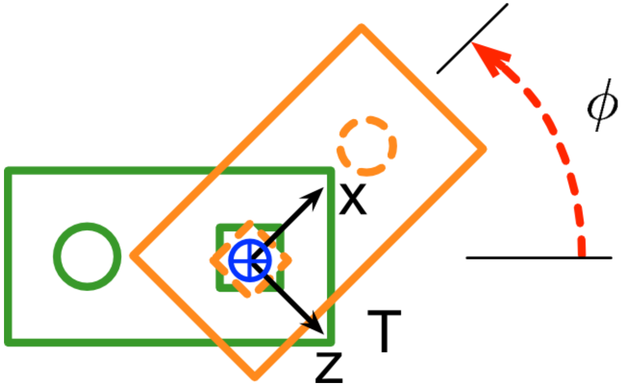


Reasons for this scenario:

- Evaluation of behavior with large contact areas

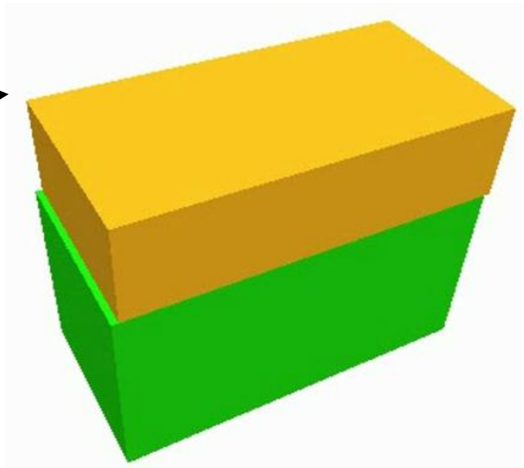
Analytical (ideal) model

- Expected direction of T: +z; no forces
- |T| should increase as ϕ increases



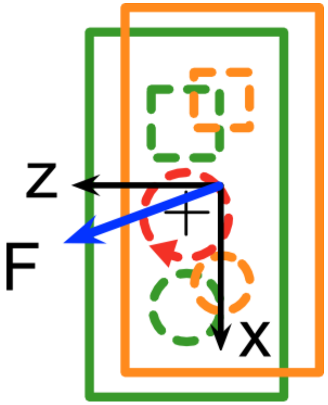
Quality Scenario IV

pins object →



Reasons for this scenario:

- Evaluation of behavior with small displacements around a configuration in which two concave objects are in large surface contact



Analytical (ideal) model

- Expected forces and torques are those that bring *pins* object towards the central axis (push *pins* object back to resting configuration)
  - Expected direction of F: sinusoid in XZ plane

Benchmarking procedure

Main steps:

1. Measured (m) and recorded values in each time stamp k: forces $\mathbf{F}_k^{\mathrm{m}}$, torques $\mathbf{T}_k^{\mathrm{m}}$, penalty values q_k^{m} (volume, penetration), computation time t_k
2. Computation of ideal (i) force $\mathbf{F}_k^{\mathrm{i}}$ and torque $\mathbf{T}_k^{\mathrm{i}}$ (volume based and penetration based model)
3. Compare ideal (i) and measured (m) values

Proposed quality measures

1. Deviation of magnitude of measured (m) forces from ideal (i) forces (RMSE)

$$\sigma_F = \frac{1}{N}\sqrt{\sum_{k=1}^{N} \left(\|\hat{\mathbf{F}}_k^{\mathrm{i}}\| - \|\hat{\mathbf{F}}_k^{\mathrm{m}}\| \right)^2}, \quad \hat{\mathbf{F}} = \frac{\mathbf{F}}{\|\mathbf{F}\|_{\max}}$$

where N being total number of time stamps

2. Deviation for the direction

$$\gamma_F = \frac{1}{N}\sum_{k=1}^{N} \arccos \frac{\mathbf{F}_k^{\mathrm{i}} \mathbf{F}_k^{\mathrm{m}}}{\|\mathbf{F}_k^{\mathrm{i}}\| \cdot \|\mathbf{F}_k^{\mathrm{m}}\|}$$

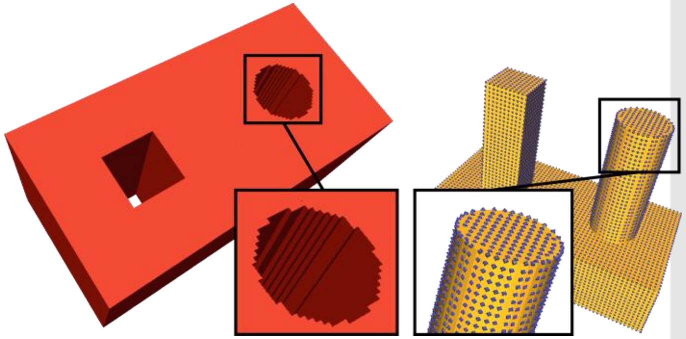
3. Similarly for torques
4. Amount of noise by short time Fourier transform

Evaluated Algorithms

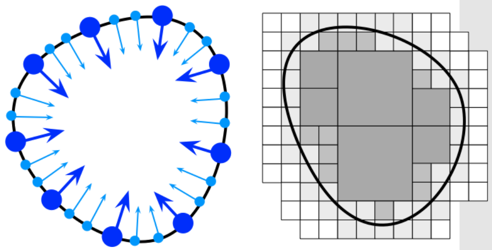
- Quite different algorithms Voxmap-Pointshell (VPS) and Inner Sphere Tree (IST)
- Both Penalty based haptic rendering method

	IST	**VPS**
Penalty value	Intersection volume	Penetration depth
Data structure	Sphere packing	Voxmap & Pointshell

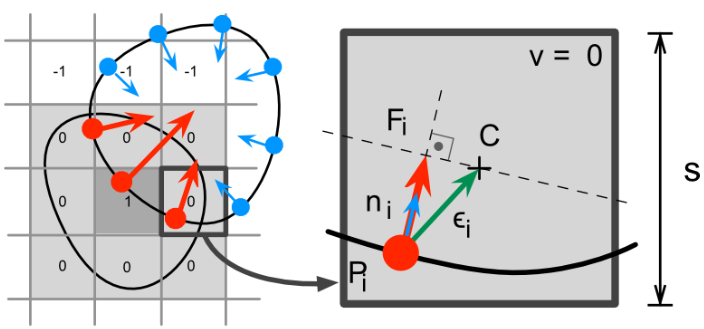
Voxmap-Pointshell algorithm



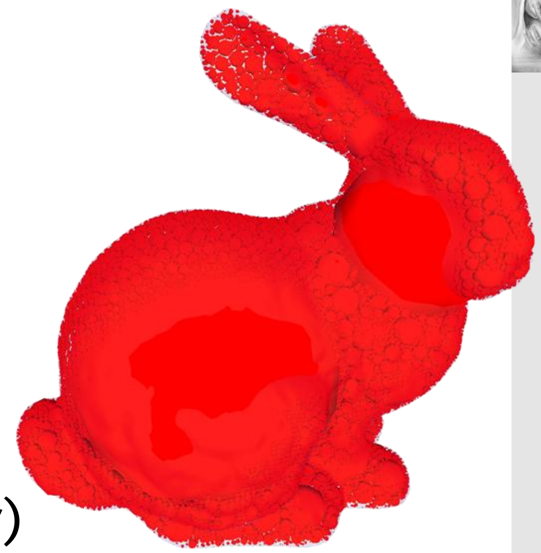
- Two types of data structure
  (generated offline)
- Voxmap:
  - 3D grid: each voxel stores discrete distance value $v \in \mathbb{Z}$ to surface
- Pointshell:
  - Set of points uniformly distributed on the surface



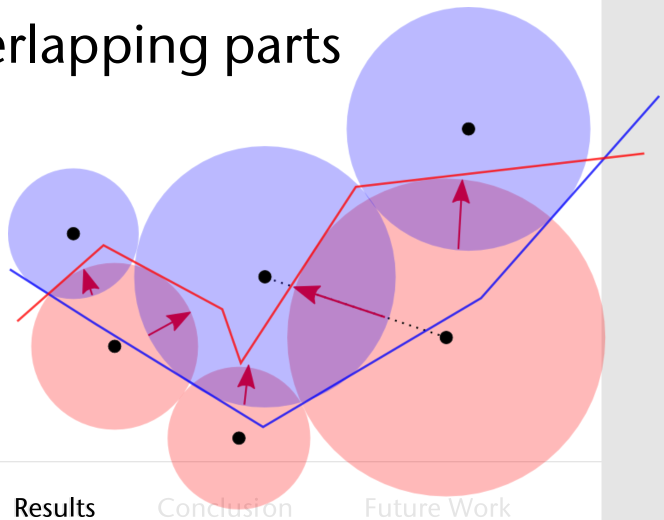
- Likely colliding points are checked for collision ($v \geq 0$)
- F = normal vectors $\mathbf{n}_i$ of colliding points P_i are summed
- Penalty value = penetrated distance



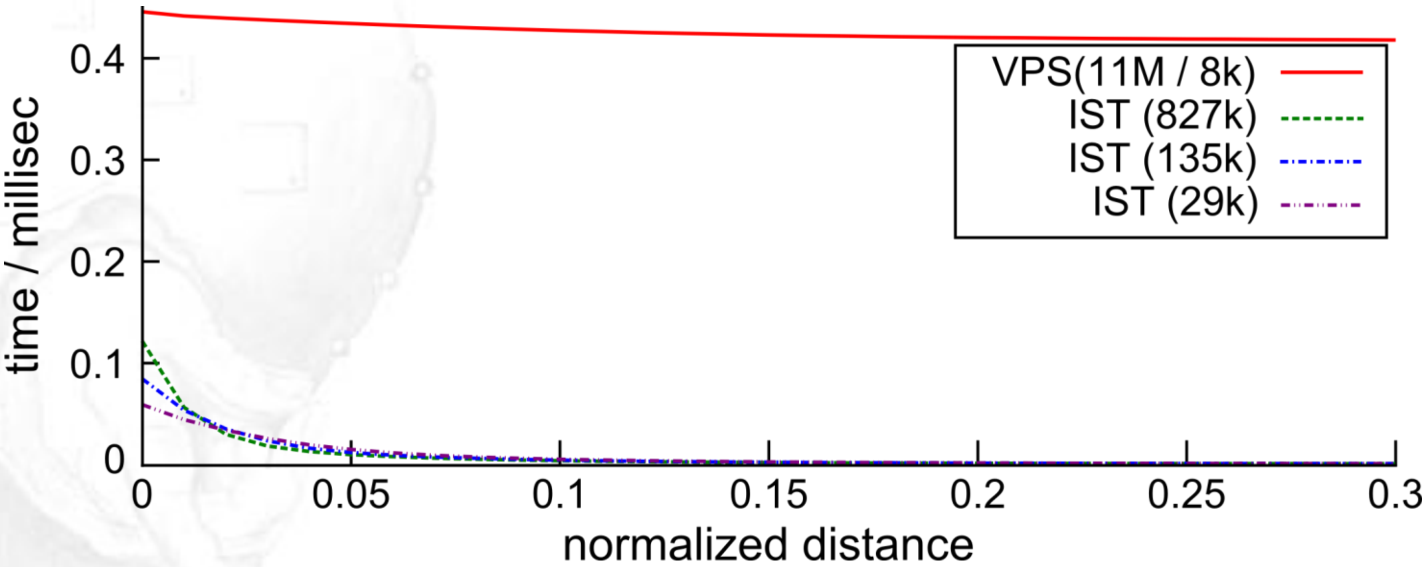
Inner Sphere Tree algorithm

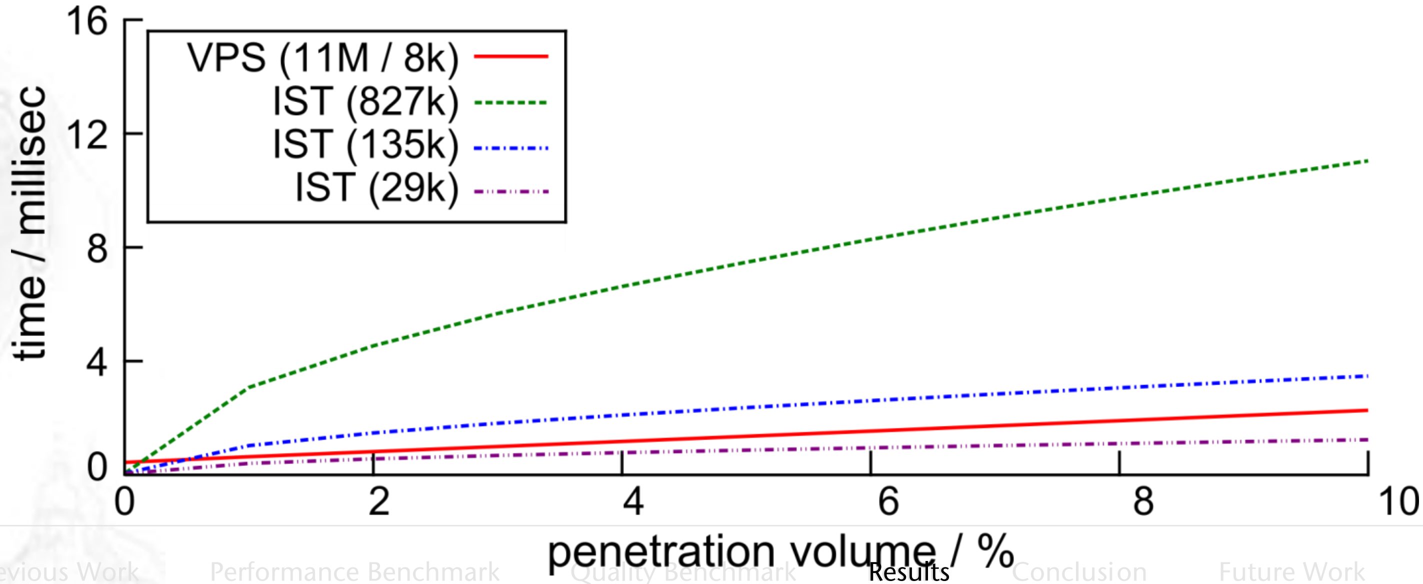


- Provides hierarchical bounding volumes from *inside* of an object
- Fill interior of model with non overlapping spheres (approximate object's volume closely)
- Independent of geometry complexity (only depend on approximation error)
- Penalty value = penetration volume computation ⟶ corresponds to water displacement of overlapping parts (physically motivated)

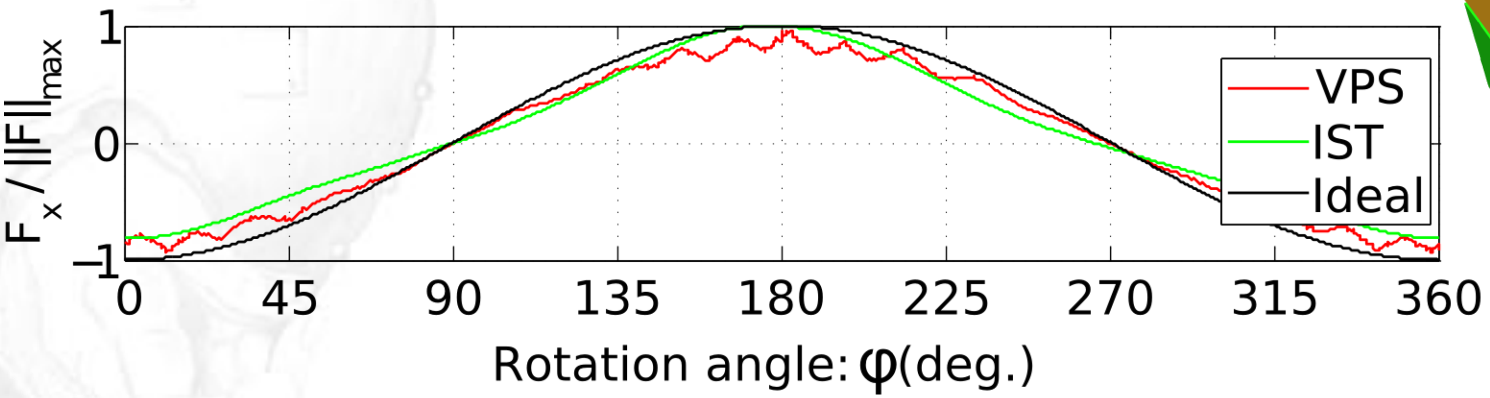


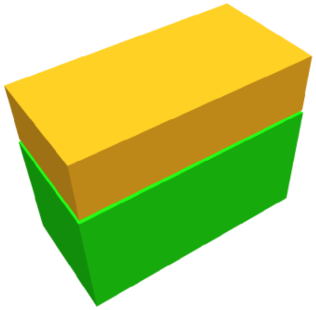
Results: Performance Benchmark

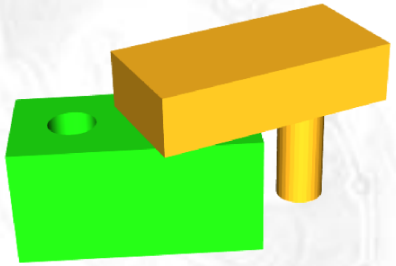


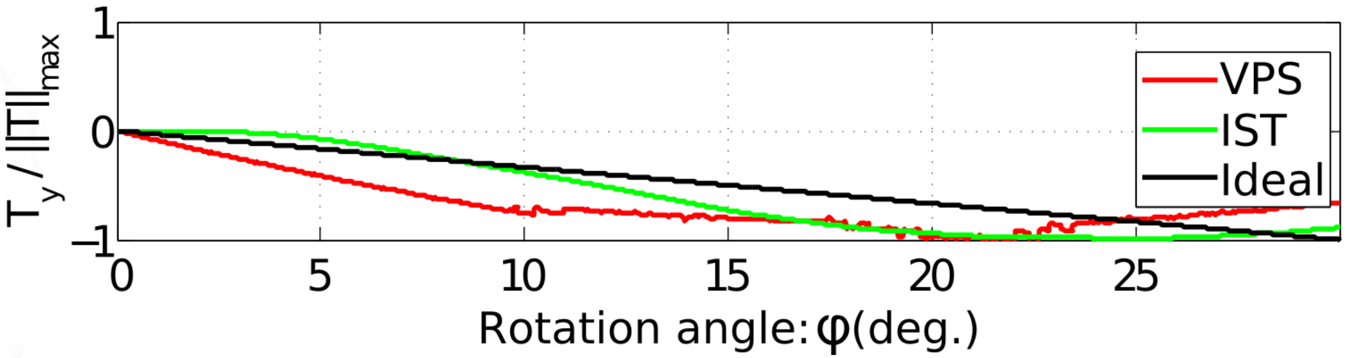


Results: Quality Benchmark

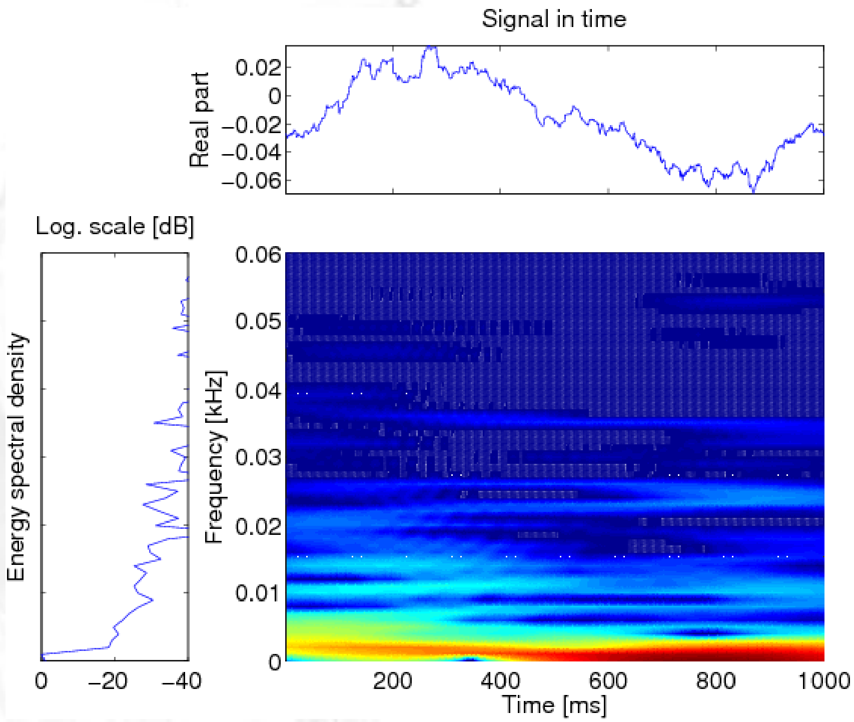




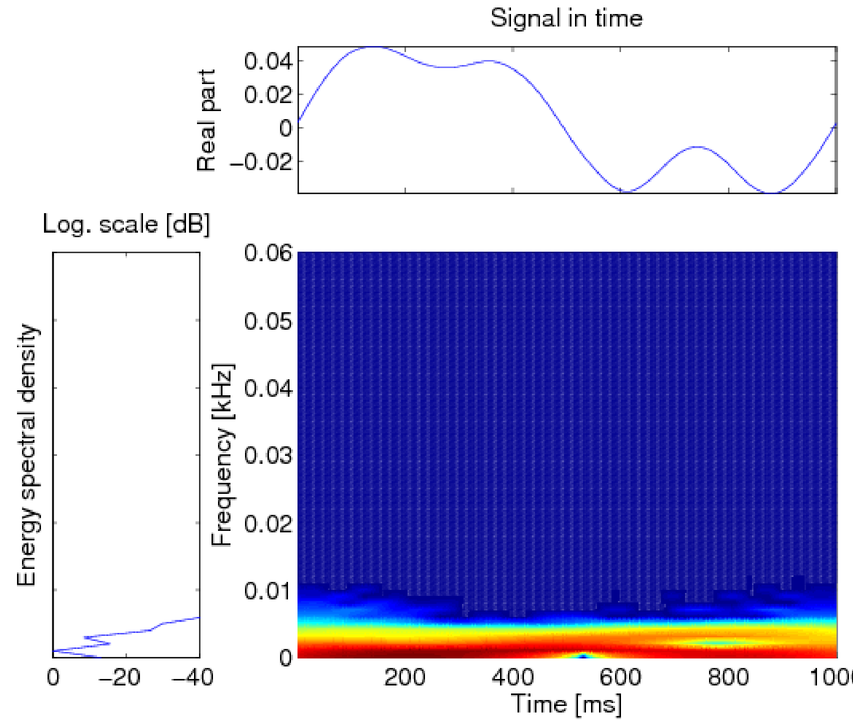




Results: Quality Benchmark



VPS



IST

- Color code intensity of frequency (dark blue represents intensity of zero)

Conclusions

- Easy to benchmark quite different collision detection algorithms
- Benchmark both performance and quality
- Cover wide range of scenarios
- Benchmark and configurations published as open source (soon)

  (http://cg.in.tu-clausthal.de/research/colldet_benchmark/index. shtml)

Future Work

- Weighting of different measurements → ranking of algorithms
- Standardized benchmarking suite for deformable objects is still missing
- Benchmarking of more algorithms

Acknowledgments

- DFG grant ZA292/1-1
- BMBF grant Avilus / 01 IM 08 001 U.