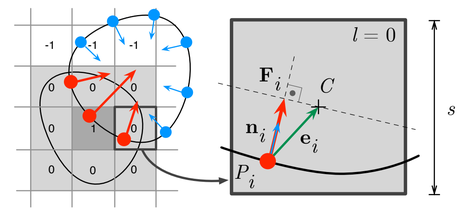
Fast and Accurate Distance, Penetration, and Collision Queries Using Point-Sphere Trees and Distance Fields

Introduction

Collision detection, force computation, and proximity queries are fundamental in interactive gaming, assembly simulations, or virtual prototyping. However, many available methods have to find a trade-off between the accuracy and the high computational speed required by haptics (1 kHz). Our haptic rendering algorithm is based on the Voxmap-Pointshell Algorithm introduced by McNeely et al. New optimized haptic data structures enable an accurate collision response of (partially) arbitrarily complex geometries within 1 ms.

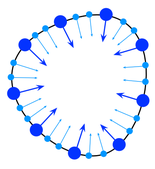
Classical Voxmap-Pointshell Algorithm



$$\mathbf{F}_i = v(\mathbf{x} = P_i)\mathbf{n}_i = \max\{\underbrace{\mathbf{n}_i\epsilon_i}_{\text{(local)}} + \underbrace{ls}_{\text{(global)}}, 0\}\mathbf{n}_i$$

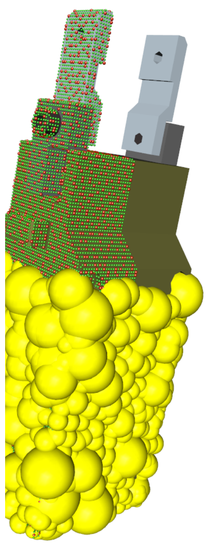
The classical Voxmap-Pointshell Algorithm uses point clouds and voxelized structures for computing collisions. In each cycle, the penetration of the points in the voxelized object is computed. This penetration depends on the discrete voxel layer in which the point is. The single collision force of one point is its normal scaled by its penetration (v), and the sum of all single forces gives the total collision force. The cross-products of forces and points yield torques.

New Haptic Data Structures



From *Pointshells* to Point-Sphere Trees

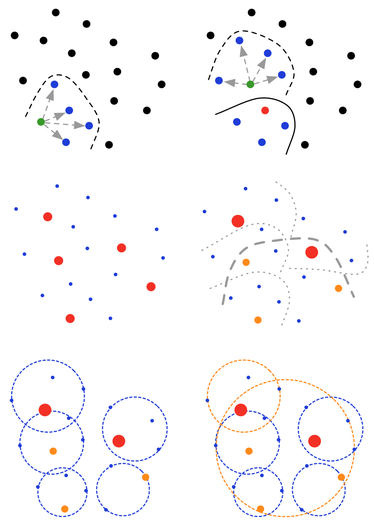
Pointshells are point-sampled representations of objects. All points are uniformly distributed on the surface and each one has an inwards pointing normal vector. Our approach generates point-sphere trees able to (*i*) **localize likely colliding regions** and (*ii*) sample the object with **several resolutions**.



1. Neighbor points of the initial *point-soup* are organized in clusters.

2. The parent point of each cluster belongs to the upper level in the tree, which is also clustered.

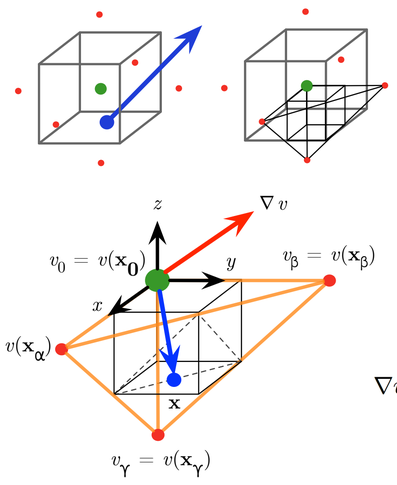
3. All points and children points within a cluster are enclosed with a minimal sphere.



From *Voxmaps* to Interpolated Distance Fields

Voxmaps are voxelized representations of objects. Each voxel contains its voxel layer value l (surface voxels: $l = 0$). The voxel value multiplied by the voxel size (s) gives the approximate distance from the voxel to the surface.

Our approach introduces voxelized distance fields, where each voxel contains the exact distance (v) from the voxel center to the surface. These structures enable **accurate penetration and distance queries**.

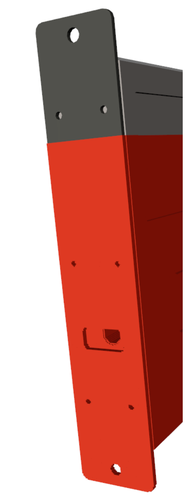


1. After detecting the voxel in which the point is, the optimal (closest) voxel neighbor set is chosen.

2. A local gradient of the distance field is computed in the neighborhood. This gradient is used to locally interpolate the penetration of the point.

$$\nabla v = [\frac{v_\alpha - v_0}{\alpha}, \frac{v_\beta - v_0}{\beta}, \frac{v_\gamma - v_0}{\gamma}],\ \alpha, \beta, \gamma \in \{-1, 1\}$$

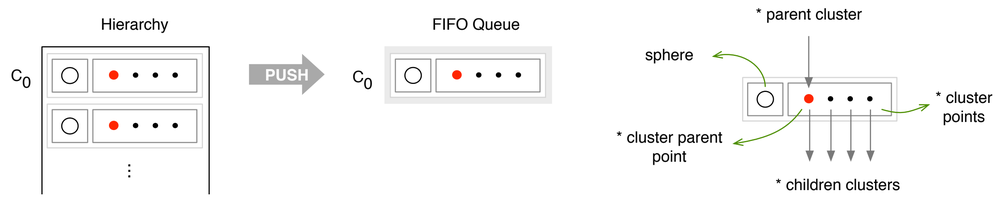
$$v(\mathbf{x}) = \frac{1}{s}[\nabla v \cdot (\mathbf{x} - \mathbf{x}_0)] + v_0$$



Collision Detection and Force Computation

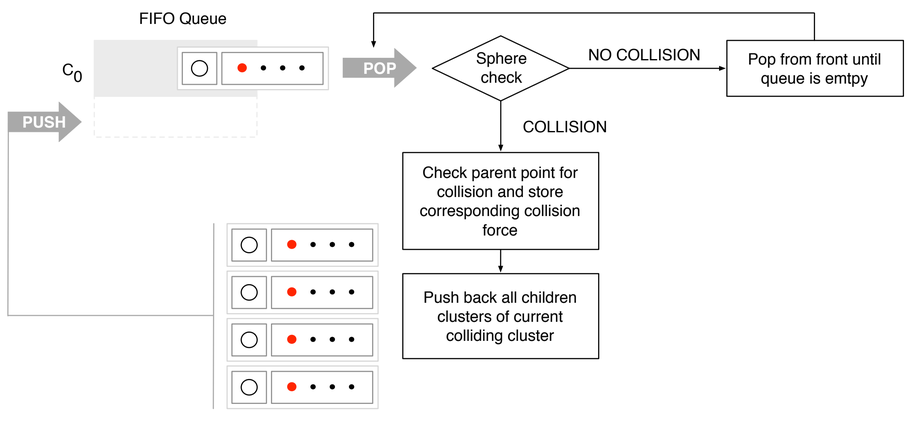
t_i ——— $t_i + 1$ ms

1. The uppest cluster with the sphere that encloses all points is pushed to the queue.



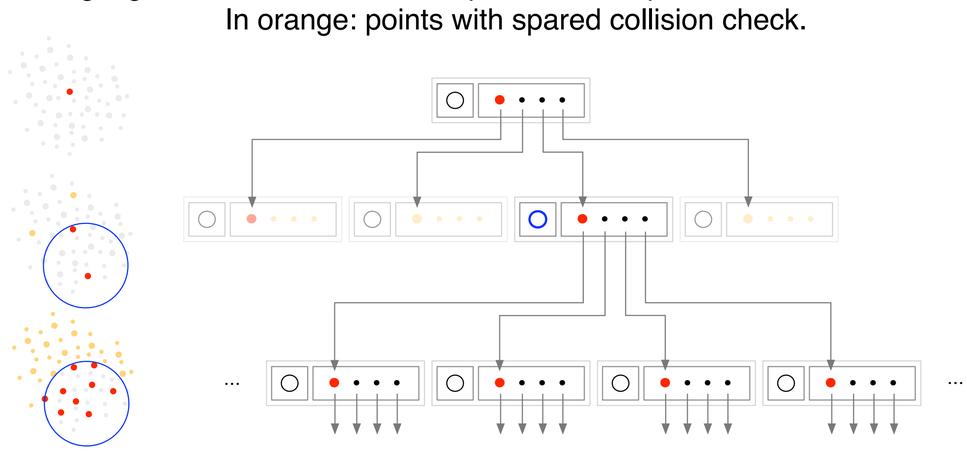
2. The queue containing likely colliding clusters is popped until it is empty. The algorithm checks whether each popped cluster's sphere is in collision; if so, the parent point of the cluster is checked for collision and children clusters are pushed to the queue. The single collision force that corresponds to each colliding point is the normal vector scaled with the locally interpolated distance to the surface.

2.1 Algorithm Flowchart:

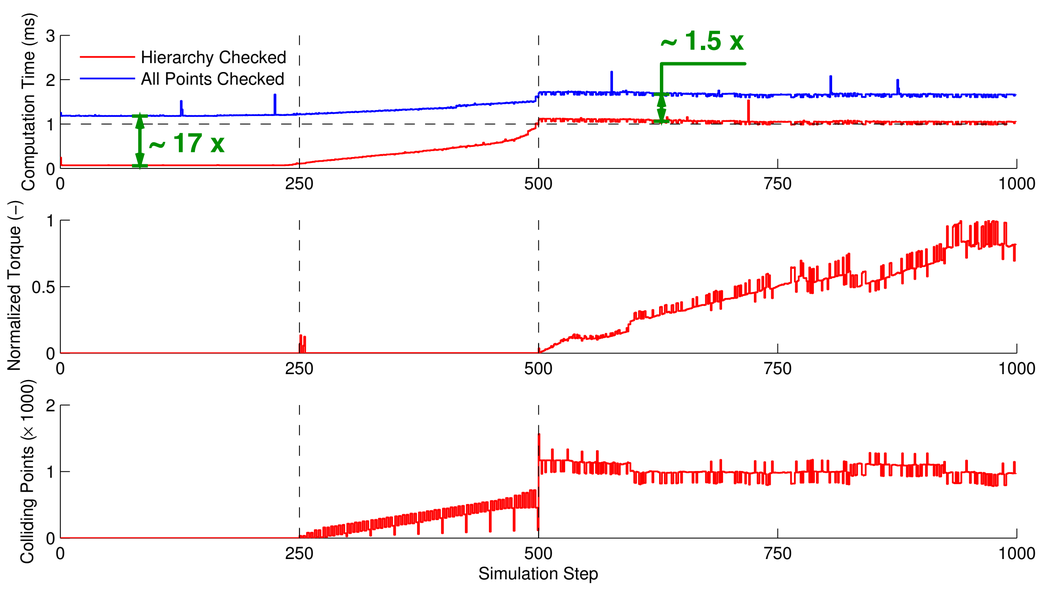


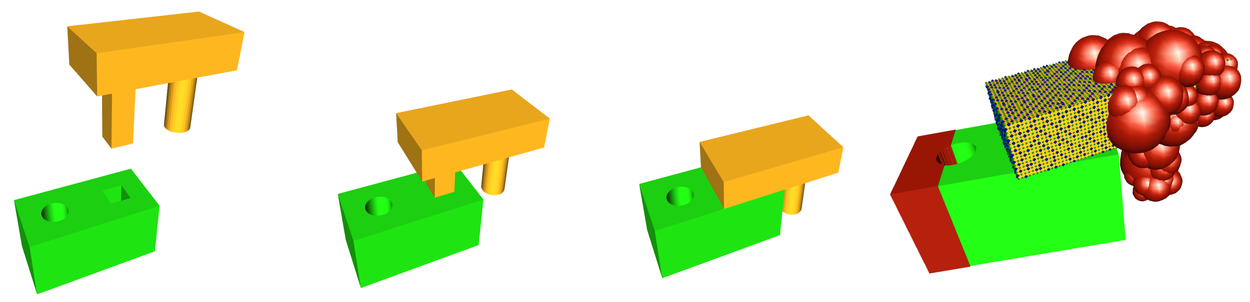
2.2 Traversal in the Point-Sphere Hierarchy:

Sphere checks yield likely colliding regions, whereas the point-tree enables using higher resolutions where required. In red: points checked for collision; In orange: points with spared collision check.



Experimental Results





1. Steps 0 - 250
The *pin* object (yellow) approaches *hole* object (green). As desired, only insignificant computation times are registered in the hierarchy traversal.

2. Steps 250 - 500
As expected, computation time increases with the contact points (and surface), but almost no torques appear in the axis of the pin.

3. Steps 500 - 5000
The number of colliding points increases suddenly when the two faces of the objects collide. Then, the *pin* object is rotated around its pin without increasing the computation time. The torque in the axis of the pin increases with the rotated angle, as expected. Data structures are partially displayed.

Note: The experiment was performed with an Intel(R) Xeon(R) X5550 @ 2.67GHz and 4 GB memory. The voxelmap has 281x161x161 voxels and the pointshell 8691 points, with 3 children/cluster.

DLR Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

Mikel Sagardia, Thomas Hulin
Institute of Robotics and Mechatronics, DLR
SIGGRAPH 2013