Haptic simulation of bone placement based on voxel models

E. Ruffaldi, M. Bergamasco, F. Barbagli, D. Morris, S. Girod, K. Salisbury

e.ruffaldi@sssup.it

28th June 2007
Summary

• Problem Statement
• Volume Based Collision Detection
• Volume Collision Response
• Optimization
• Conclusions
Corrective jaw surgery (orthognathic surgery) aims at correcting acquired or inherited dentofacial defects.

The standard technique consists in the creation of a plaster cast, that is then broken in a position and then aligned using a mechanical tool.

Such procedure allows only one break of the plaster cast because of the cost of the casts itself.
Virtual Osteoectomy

Improve the procedure first by 3D tools, and then by Haptic manipulation

- Bone Cut
- Alignment
- Plaque placement
Haptic Feedback

• Kinesthetic and Tactile feedback can be applied in various aspect of Medicine:
  – Training, for simulating the interaction with tools
  – Robotic Minimally Invasive Surgery, for enhancing the operation feedback
  – Planning, for improving the interaction with the virtual elements

• Although there are many types of Haptic Interfaces the most common are the tool and the wearable ones
Haptic interaction approaches

- The Haptic Feedback given to the user can be associated to a direct interaction of the haptic tool with a virtual object or generated for guidance and signaling.
- Initial Haptic Systems provided tool based interactions for the exploration of rigid surfaces or their manipulation.
- The advancement of the interfaces and of the computational power allows the interaction with deformable surfaces and the manipulation of objects with complete computation of the interaction.
- The 6DOF haptic rendering is in particular useful for the manipulation of virtual objects.
Objectives of the Work

- Provide a 6-DOF system for the haptic rendering of the contact between two complex bone models
- The source of data is a highly complex volumetric model, acquired with X-Ray
- Improve over the state of the art, the Voxel Point Shell algorithm
- The Haptic Rendering poses the strict requirement of a high frequency rate (1 kHz)
Overview of the solution

- Visualization
- Volume representation
- Collision Detection
- Force computation
- Visualization
Challenges

- Real-Time Interaction with Volumetric Data (haptic and visual)
  - Size of the data set
- Evaluation of the accuracy of the planning
  - Correctness of the alignment using reference points
  - Interaction is not necessary to be realistic
Overview of the solution

- Collision Detection
- Volume representation
- Force computation
- Visualization

28th June 2007 – CARS, Berlin, Germany
Volume Representation

- A Voxel representation of an object is a uniform space partitioning
- Arbitrary object geometry, also from imaging technologies
Voxel Content

- Every spatial cube can be empty or full. Additionally we define Surface and Proximity voxel
- Every voxel stores optionally distance, gradient, nearest primitive
Volume Storage

- Uniform storage
  - Expensive
  - Not friendly to memory access
- Hashing
  - Improves over uniform
  - Still complex for spatial computations
- Hierarchical using Octrees
  - Optionally they can be generalized to more than 8 children per level
Octrees

Example of Voxel model with side 128
Corresponding to 7 levels
Overview of the solution

- Collision Detection
- Volume representation
  - octree
- Force computation
- Visualization

28th June 2007 – CARS, Berlin, Germany
Reference Algorithm: Voxel PointShell

Test each surface point of the probing object against the static voxels of the world
Cost is limited by the number of points in the Shell

[McNeeily 1999, 2005]

The force contribution of each collision is computed by a Tangent Plane Force Model
Reference Algorithm – Voxel Point Shell
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit Sphere Tree

- Global collision scheme with reduced memory storage
- Same structure for collision and rendering
- Reference system invariant
Implicit circles for quadtree

An example with a two level quadtree

Voxel side $s$
Circle radius is $s/2 \sqrt{2}$

The four offsets of the children are precomputed and independent to the current radius

Child center is relative to parent center at level $L$ by $2^{L-2}$

Level = 2

radius = $s \ 2^{L-1} \sqrt{2} = s/2$
center = origin
Implicit Sphere Tree Example

Example of Voxel model with side 128
Collision Detection Benchmarking

![Collision Detection Graphs]

28th June 2007 – CARS, Berlin, Germany
Optimizing the Descent

Node Occupancy optimization

Level Skipping

L=0

L=1

L=2
Overview of the solution

Collision Detection
- implicit sphere tree

Force computation

Volume representation
- Octree based

Visualization
Collision Response

- Given the set of collision points there are several methods for computing the resulting force
- In general the idea is to compute a Dynamic Simulation of the two Rigid Bodies
- The three main models are:
  - Constraint Based
  - Impulsive
  - Penalty Based
    - Simultaneous
    - Concurrent
Collision Response

- In this context we opted for Simultaneous Impulsive Collision Response
- The deepest contact pair is selected and used for computing the Impulsive Response
- Eventually friction can be applied for simulating surface properties

\[
\dot{u}_n = \frac{-(1+\epsilon)u_n}{1/m_a + 1/m_a + 1/I_a \left| r_a \right|^2 \sin^2 \theta_a + 1/I_b \left| r_b \right|^2 \sin^2 \theta_b}
\]

separating contact points
Given the collision set
Find the deepest pair and apply the impulse
With a virtual integration step
Do again until all non separating pairs are Processed or some threshold is reached
Virtual Coupling

- The force computed by the 6-DOF Collision Response could be applied directly to the body (Direct Rendering)
  - The varying number of contact points would produce a varying stiffness
  - Performance depends on the speed of the Collision Detection
- Virtual Coupling overcomes these problems by smoothing the interaction (Colgate95, McNeely99)
- In VC the body and the haptic handle are connected by a damped 6-DOF spring
Conclusion

Visualization

• octree

Collision Detection
• implicit sphere tree

Force computation
• simpulse based
• separating points

Volume representation
• octree
Additional applications

The described algorithm can be applied to other cases of bone manipulation for planning.

Additionally, it can be used in the context of Virtual Prototyping for the assessment of mechanical parts.
Challenges

- In the context of the specific application for operation planning
  - Identification of peculiar points
  - Automatic alignment
  - Visualize Collision Detection
- Generalize the Collision Scheme with Volume editing for drilling
- *Parallelization* of the algorithm for new Multicore and GPU systems
- *Sensation Preserving* optimization
Open Issues - Parallelization

- Parallel computing has a lot of history and only recently it is spreading.
- Multicore CPUs with 2-4 are now on the market and more parallelism is coming in the future.
- GPUs are much more parallel but with different constraint.

- How we can parallelize the Collision Detection algorithms?
- Depending on the complexity of the Voxel model we can assign different parts of the models to different CPUs or parallelize the pure computation.

- Pointshell can be parallelized grouping points and assigning them to different CPUs.
Open Issues - Parallelization

- The Collision Detection between Bounding Volumes generates is performed using a Collision Tree
- In the simplest case the Collision Tree is obtained with a recursive descend
- For parallelization we transform the recursion into parallel executions, one for each level

![Diagram]

Root (1)

Collision path

Every possible combination of BV at different levels

Color represents a different processor
Open Issue - Sensation Preserving

- Sensation Preserving is a concept introduced by Otaduy in 2003 for limiting the descent in the Collision Tree when the detail of the contact is not necessary.
- It is equivalent to the concept of Level of Detail in Computer Graphics.
Open Issue - Sensation Preserving

Sub-Mesh Surface Deviation

Sub-Mesh Lost Volume phi

max

divide

max

Weighted Surface Deviation for Collision

Contact Area

Contact Area

Sub-Mesh Surface Deviation

Sub-Mesh Lost Volume phi

A

B

descend

threshold

accept

How we can map this approach to Octree Voxel Volumes?
Acknowledgments

• Support was provided by the US National Institute of Health under grant LM07295, BioX 2DMA178 and by the European Community with the ENACTIVE NoE and SKILLS IP projects

• Prof. Ken Salisbury and Prof. MD. Sabine Girod for supporting my research

• Prof. Massimo Bergamasco for the inspiration
Thank you