Fast Discretized Geometric Algorithms for Union and Envelope Computations

Dinesh Manocha
UNC-Chapel Hill
http://gamma.cs.unc.edu/

Collaborators

- Mark Foskey
- Young Kim
- Shankar Krishnan
- Ming C. Lin
- Avneesh Sud
- Gokul Varadhan

Arrangement Problems

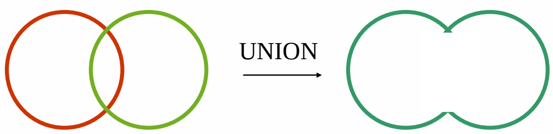
- **Arrangements**
  - Decomposition of space into connected open cells
  - Fundamental problem in computational geometry and related areas
- **Underlying structure in many geometric applications**
  - Swept Volumes
  - Minkowski Sums
  - CSG or Boolean operations
  - Many more…….

Basic Computational Pipeline

- Enumerate a set S of primitives that contribute to the final surface
- Compute the arrangement A(S) by performing intersection and trimming computations
- Traverse the arrangement and extract a substructure δA(S)

Example: CSG Union Operation

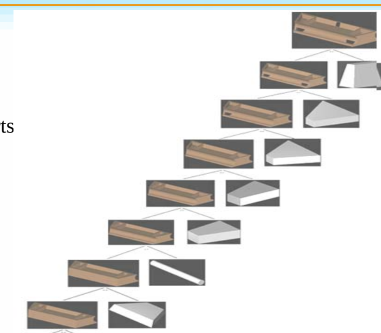
Boundary = outer envelope in the arrangement of the primitives



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

CSG Operations

Design of complex parts



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Boundary Evaluation of Complex CSG Models

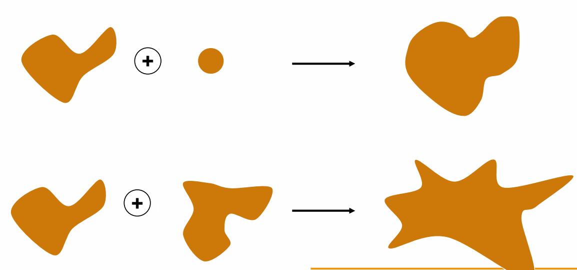
Bradley Fighting Vehicle
1200+ solids
8,000+ CSG operations

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Minkowski Sum

$A \oplus B = \{ a+b \mid a \in A, b \in B \}$

OFFSET



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

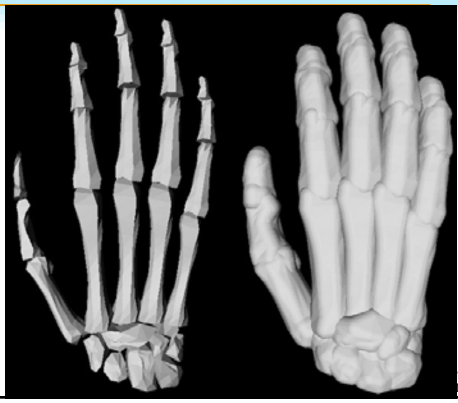
Minkowski Sums: Motivation

- Configuration space computation
- Offsets
- Morphing
- Packing and layout
- Friction model

Offset Computation

Offset: Minkowski sum with a sphere

Input: 2982 triangles



Minkowski Computation

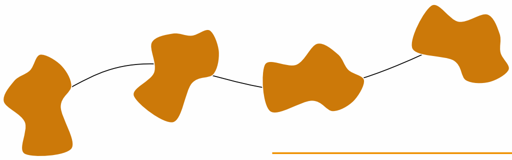
- Decompose A and B into convex pieces
- Compute pairwise convex Minkowski sums
- Compute their union

Issues:

- –High combinatorial complexity = $O(n^6)$
- –Exact computation almost impractical

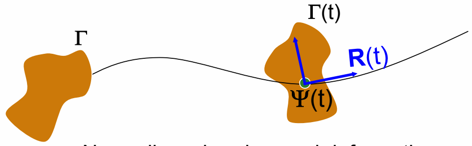
Swept Volume (SV)

- Volume generated by sweeping an object in space along a trajectory
- **Goal:** Compute a boundary representation of SV



Sweep Equation

- $\Gamma(t) = \Psi(t) + R(t)\,\Gamma,\ \ 0 \le t \le 1$
  - Γ **:** Generator (polyhedron)
  - Ψ**(t)** : Smooth vector in R^3 (sweeping path)
  - R**(t) :** Local orientation
- Swept Volume of $\Gamma := \cup\Gamma(t)$

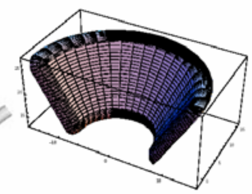


• No scaling, shearing, and deformation

Swept Volume: Applications

Numerically Controlled Machine Verification

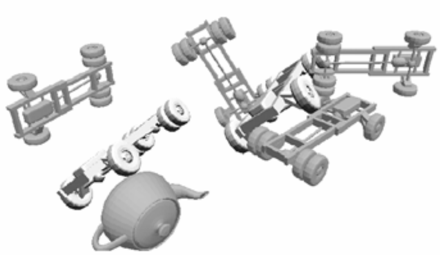
Tool and workpiece



Material removal

Swept Volume: Applications

Collision detection between discrete instances

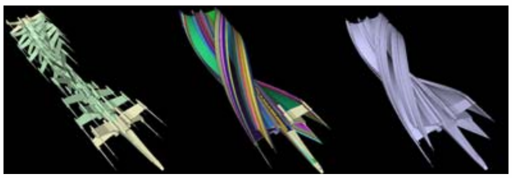


Swept Volume Computation

- Enumerate ruled and developable surfaces
- Boundary of SV = outer envelope of the arrangement

Swept Volume Computation

- X-Wing Model
  - 2496 triangles
  - 3931 ruled and developable surfaces
  - Intersection curves of degree as high as nine



Sweep Trajectory　Arrangement　Boundary of SV

Computation of Swept Volumes

- Generate ruled and developable surfaces
- Compute their arrangement
- Traverse the arrangement and extract the *outermost* boundary (outer envelope computation)

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Complexity of Arrangements

- High computational and combinatorial complexity
  - Super-quadratic in number of surfaces
- Accuracy and robustness problems
- No good practical implementations are available

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Approximation Pipeline

- Enumerate surface primitives
- Compute distance fields on a voxel grid
- Perform filtering operations on distance fields
- Use improved reconstruction algorithms

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Approximation Pipeline

- Enumerate surface primitives
- Compute distance fields on a voxel grid
- Perform filtering operations on distance fields
- Use improved reconstruction algorithms
  - Max-norm computations for reliable voxelization
  - Recover all connected components
  - Faithfully reconstruct sharp features

Organization

- Fast distance field computation
- Max-norm based voxelization
- Boundary reconstruction
- Analysis
- Applications
  - Boundary evaluation
  - Swept volume computation
  - Medial axis computation
  - Minkowski sums

Organization

- Fast distance field computation
- Max-norm based voxelization
- Boundary reconstruction
- Analysis
- Applications
  - Boundary evaluation
  - Swept volume computation
  - Medial axis computation
  - Minkowski sums

Distance Fields

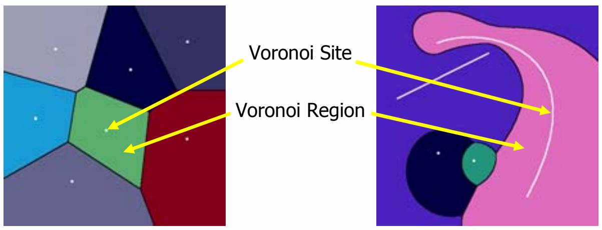
- **Distance Function**

  For a site a scalar function $f: R^n \rightarrow R$ representing the distance from a point $P \varepsilon R^n$ to the site
- **Distance Field**

  For a set of sites, the minima of all distance functions representing the distance from a point $P \varepsilon R^n$ to closest site

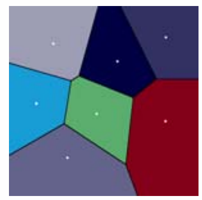
Voronoi Diagrams

Given a collection of geometric primitives, it is a subdivision of space into cells such that all points in a cell are *closer* to one primitive than to any other

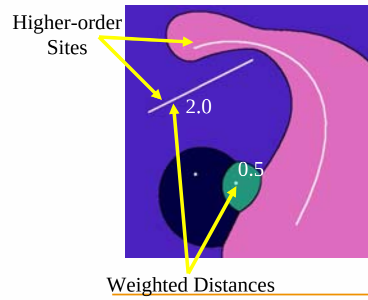


Ordinary Generalized

- Point sites
- Nearest Euclidean distance



- Higher-order site geometry
- Varying distance metrics



Voronoi Diagram & Distance Fields

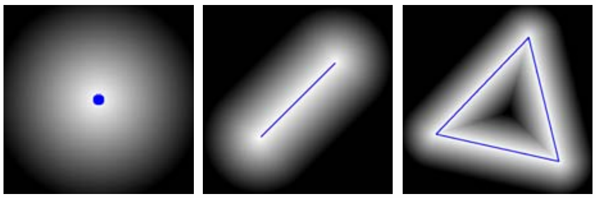
- Minimization diagram of distance functions generates a Voronoi Diagram
- ***Projection of lower envelope*** of distance functions

Distance Fields: Applications

Collision Detection
Surface Reconstruction
Robot Motion Planning
Non-Photorealistic Rendering
Surface Simplification
Mesh Generation
Shape Analysis

GPU Based Computation

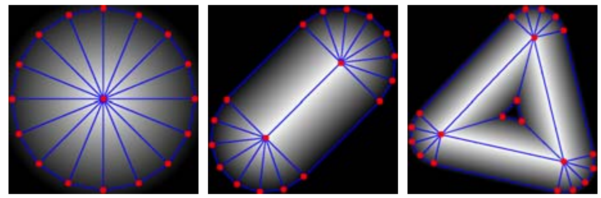
- **HAVOC2D, HAVOC3D [Hoff et al. 99,01]**
  - Evaluate distance at each pixel for all sites
  - Evaluate the distance function using graphics hardware



Point Line Triangle

Approximating the Distance Function

- Avoid per-pixel distance evaluation
- Point-sample the distance function
- Reconstruct by rendering polygonal mesh

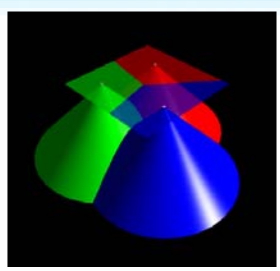


Point Line Triangle

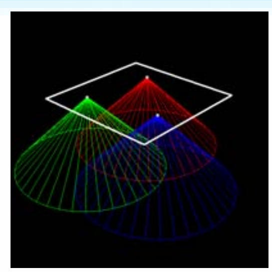
GPU Based Computation

- Triangular mesh approximation of distance functions
- Render distance meshes using graphics hardware

Meshing the Distance Function



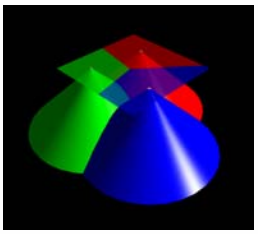
Shape of distance function for a 2D point is a cone



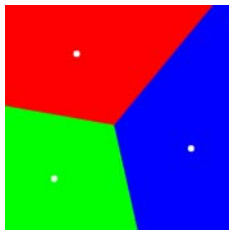
Need a bounded-error tessellation of the cone

Graphics Hardware Acceleration

- Rasterization to reconstruct distance values
- Depth test to perform minimum operator



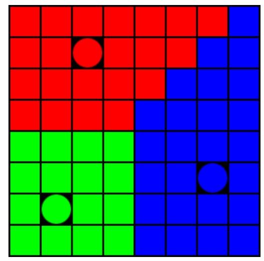
Perspective, 3/4 view



Parallel, top view

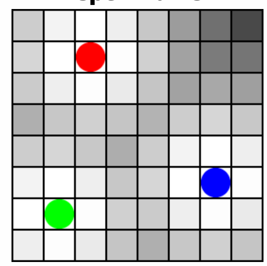
Results in the Frame Buffer

Color Buffer



Voronoi Regions

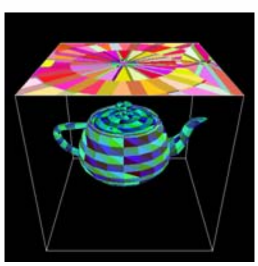
Depth Buffer



Distance Field

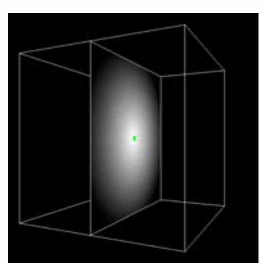
3D Voronoi Diagrams

- Graphics hardware can generate one 2D slice at a time
- Sweep along 3rd dimension (Z-axis) computing 1 slice at a time

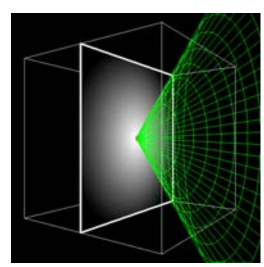


Distance Field of the Teapot Model

Shape of 3D Distance Functions

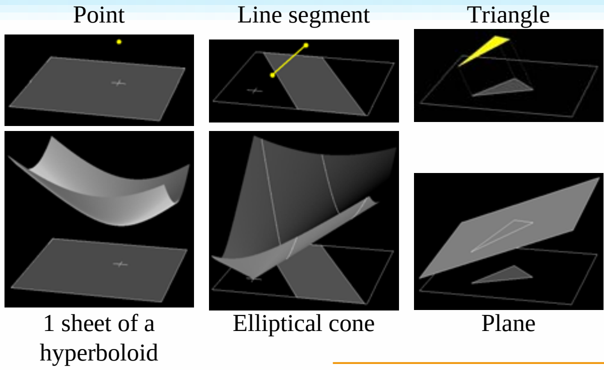


Slices of the distance function for a 3D point site



Distance meshes used to approximate slices

Shape of 3D Distance Functions



Bottlenecks in HAVOC3D

- **Rasterization:**
  - Distance mesh can fill entire slice
  - Complexity for *n* sites and *k* slices = O(*kn*)
  - Lot of Fill !
- **Readback:**
  - Stalls the graphics pipeline
  - Not suitable for interactive applications

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Improved Distance Field Computation (DiFi)

- Use graphics hardware
- Exploit spatial coherence between slices
- Use the programmable hardware to perform computations

[Sud and Manocha 2003]

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Improved Distance Field Computation (DiFi)

Reduce fill: Cull using estimated voronoi region bounds

- **Along Z:** Cull sites whose voronoi regions don't intersect with current slice
- **In XY plane:** Restrict fill per site using planar bounds of the voronoi region

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Voronoi Diagram Properties

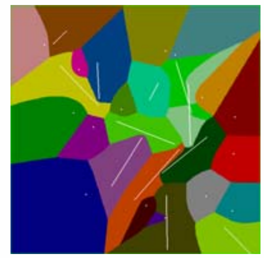
- Within a bounded region, all voronoi regions have a bounded volume

9 Sites, 2D

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Voronoi Diagram Properties

- Within a bounded region, all voronoi regions have a bounded volume
- As site density increases, average spatial bounds decrease



27 Sites, 2D

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Voronoi Diagram Properties

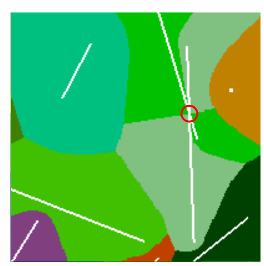
Voronoi regions are connected

- Valid for l_2 , l_{inf} etc. norms

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Voronoi Diagram Properties

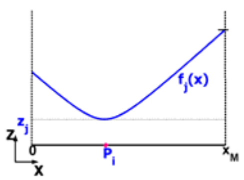
- **Voronoi regions are connected**
  - Valid for l_2 , l_{inf} norms
- **Special cases:** Overlapping features



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Voronoi Diagram Properties

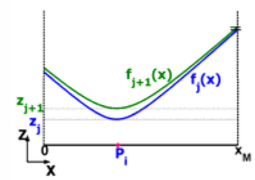
- High distance field coherence between adjacent slices
- Change in distance function between adjacent slices is bounded



Distance functions for a point site P_i to slice Z_j

Voronoi Diagram Properties

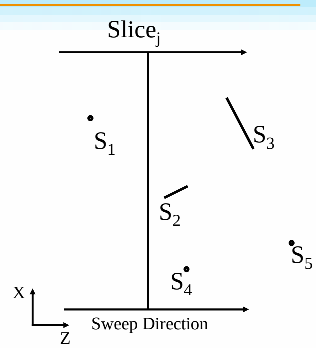
- High distance field coherence between adjacent slices
- Change in distance function between adjacent slices is bounded



Distance functions for a point site P_i to slice Z_{j+1}

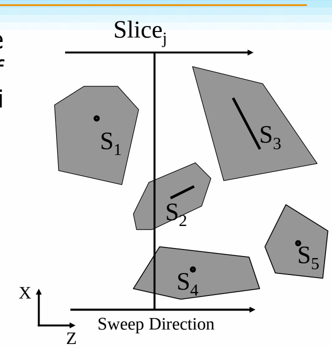
Site Culling: Classification

For each slice partition the set of sites



Site Culling: Classification

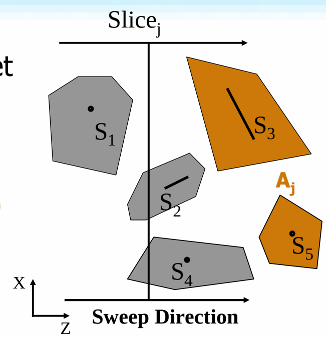
For each slice partition the set of sites using voronoi region bounds



Site Culling: Classification

For each slice partition the set of sites, using voronoi region bounds:

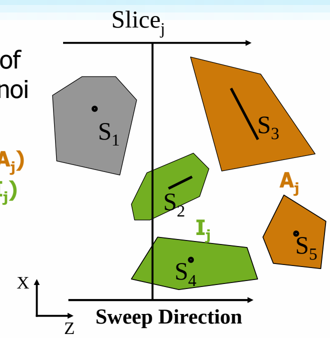
- Approaching (A_j)



Site Culling: Classification

For each slice partition the set of sites, using voronoi region bounds:

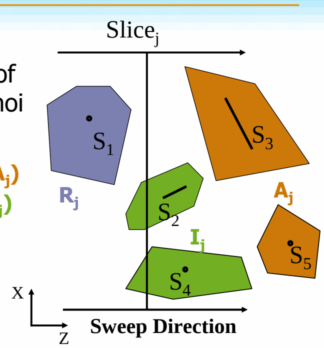
- Approaching (A_j)
- Intersecting (I_j)



Site Culling: Classification

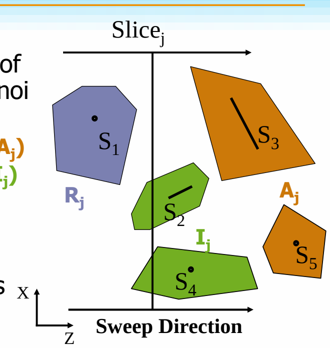
For each slice partition the set of sites, using voronoi region bounds:

- Approaching (A_j)
- Intersecting (I_j)
- Receding (R_j)



Site Culling: Classification

- For each slice partition the set of sites, using voronoi region bounds:
  - Approaching (A_j)
  - Intersecting (I_j)
  - Receding (R_j)
- Render distance functions for Intersecting sites only

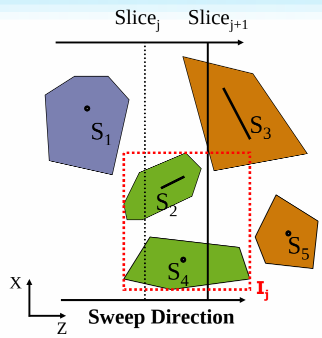


Coherence: Adjacent Slices

Updating $\mathbf{I_j}$

$\mathbf{I_{j+1} = I_j} \ldots$

Previously intersecting



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

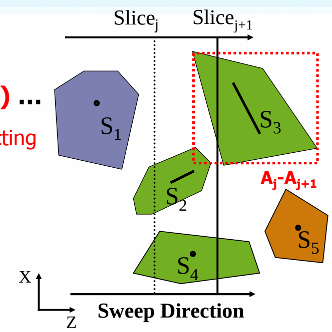
Coherence: Adjacent Slices

- **Updating $\mathbf{I_j}$**

$\mathbf{I_{j+1} = I_j}$

$\mathbf{+ (A_j - A_{j+1})} \ldots$

Approaching → Intersecting



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Coherence

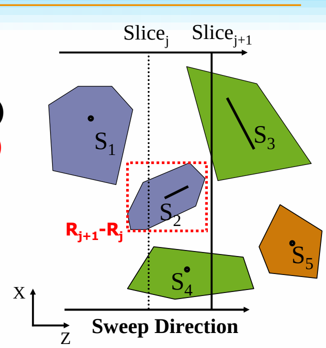
- **Updating $\mathbf{I_j}$**

$\mathbf{I_{j+1} = I_j}$

$\mathbf{+ (A_j - A_{j+1})}$

$\mathbf{- (R_{j+1} - R_j)}$

Intersecting → Receding



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

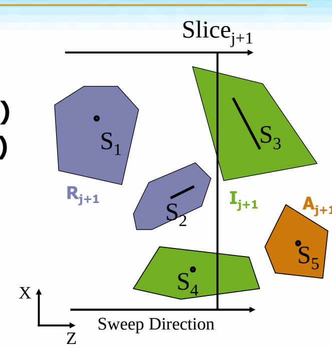
Coherence

- **Updating $\mathbf{I_j}$**

$\mathbf{I_{j+1} = I_j}$

$\mathbf{+ (A_j - A_{j+1})}$

$\mathbf{- (R_{j+1} - R_j)}$



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

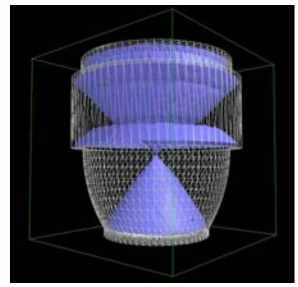
Estimate Potentially Intersecting Set (PIS)

- Computing exact intersection set = Exact voronoi computation
- **Conservative Solution**:
  - Use hardware based occlusion queries
  - Determine number of visible fragments
  - Computes *potentially intersecting* sites (PIS) $\hat{I}$

$$\hat{\mathbf{I}}_{\mathbf{j}} \supseteq \mathbf{I}_{\mathbf{j}}$$

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

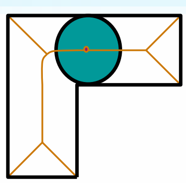
Application to Medial Axis Computation

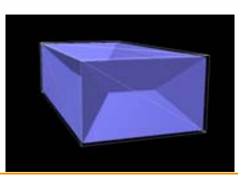


The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Blum Medial Axis

- Locus of centers of maximal contained balls
- Well-understood medial representation
- **Applications**
  - Shape analysis
  - Mesh generation
  - Motion planning
- Exact computation is hard





The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

θ-Simplified Medial Axis M_θ

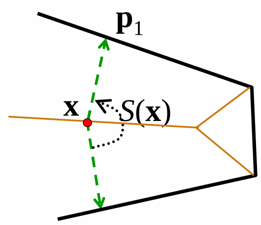
- A subset of the full medial axis *M*
- Relies on *separation angle* from points on the medial axis to the boundary
- More stable than Blum medial axis

[Foskey, Lin and Manocha 2002]

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Separation Angle

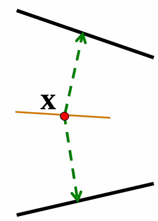
- Angle separating the vectors from x to nearest neighbors
- If more than 2 nearest neighbors, maximum angle is used



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Large Separation Angle

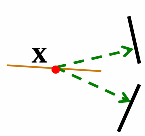
Point is roughly *between* its nearest neighbor points



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Small Separation Angle

Point is *off to one side* of its nearest neighbor points



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

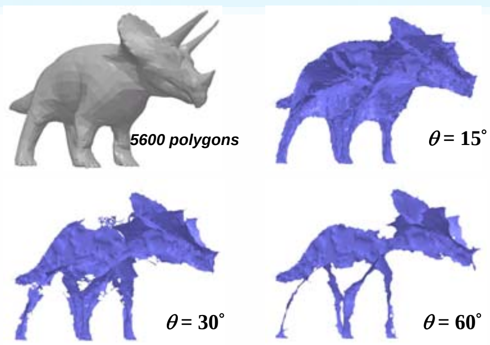
Simplified Medial Axis

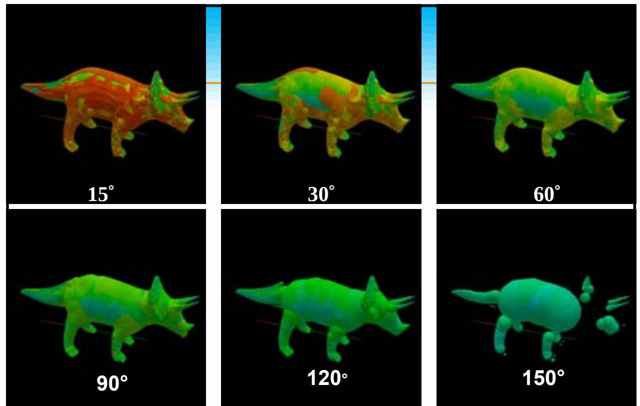
$$M_\theta = \{\mathbf{x} \in M \mid S(\mathbf{x}) > \theta\}$$

- Start with medial axis M
- Eliminate portions with $S(x) \le \theta$

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

3D Example: Triceratops



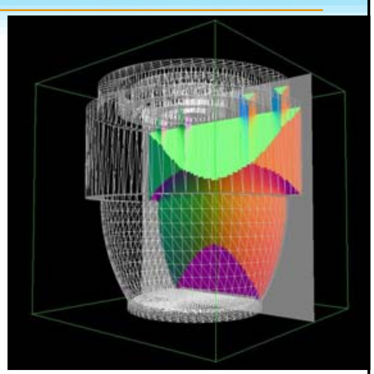


Shape Simplification using Simplified Medial Axis

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Direction Field

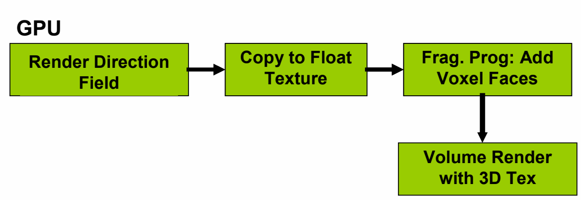
- Gradient of Distance Field
- Direction image rendered for each slice (constant z)
- Direction vectors encoded as RGB triples
- Length encoded in depth buffer



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Simplified MAT Computation using GPUs

Computation using **DiFi** [Sud et al. 2003]



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Simplified MAT Computation using Graphics Hardware

Real-time Capture from a Dell Laptop with NVIDIA GeForce4 To Go graphics card

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Simplified MAT Computation using Graphics Hardware

Real-time Capture from a Dell Laptop with NVIDIA GeForce4 To Go graphics card

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Direction Field Computation

- **4 -20 times speedup over HAVOC3D**

Model	Polys	Resolution	HAVOC (s)	DiFi (s)
Shell Charge	4460	128x126x126	31.69	3.38
Head	21764	79x106x128	52.47	13.60
Bunny	69451	128x126x100	212.71	36.21
Cassini	90879	94x128x96	1102.01	47.90

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Surface Reconstruction

- **2 - 75 times speedup**

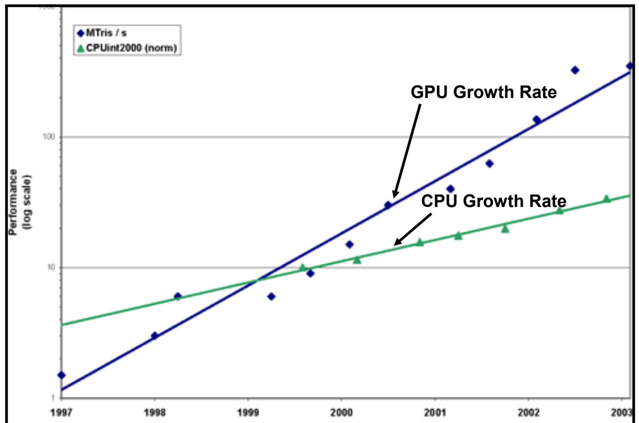
Model	Resolution	CPU (s)	GPU (s)
Shell Charge	128x126x126	3.50	0.14
Head	79x106x128	0.18	0.08
Bunny	128x126x100	0.68	0.13
Cassini	94x128x96	7.59	0.1

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Reconstruction: CPU vs. GPU

- **Depends on grid size**
- **2 - 75 times speedup via GPUs**

Model	Resolution	T(CPU) (s)	T(GPU) (s)
Shell Charge	128x126x126	3.50	0.14
Head	79x106x128	0.18	0.08
Bunny	128x126x100	0.68	0.13
Cassini	94x128x96	7.59	0.1



Benefits of using GPUs

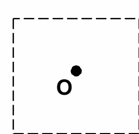
Organization

- Fast distance field computation
- Max-norm based voxelization
- Boundary reconstruction
- Analysis
- Applications
  - Boundary evaluation
  - Swept volume computation
  - Medial axis computation
  - Minkowski sums

Max-Norm (l_∞) Computation

- Max-Norm
  - Natural metric for axis-aligned voxels

$$\| \mathbf{p} \|_\infty = \max(|x|, |y|, |z|)$$



Iso-distance ball
$\| \mathbf{x} \|_\infty = c$
is a cube

Applications of Max-Norm Computation

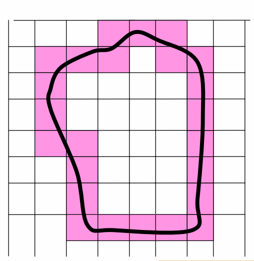
- Markov decision processes [Tsitsiklis et al. 96, Guestrin et al. 2001]
- Discrete objects in supercover model [Andres et al. 96]
- Image analysis [Lindquist 99]
- Volume graphics [Wang & Kaufman 94, Sramek & Kaufman 99]

Goal

- Efficiently compute max-norm distance between a point and a wide class of geometric primitives
- Motivation
  - Voxelization

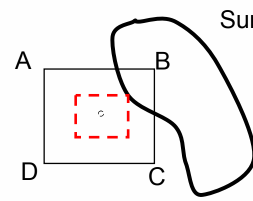
Voxelization

- Represent a scene by a discrete set of voxels



Voxelization

- Reduce to max-norm distance computation

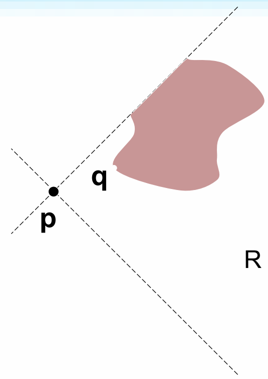


Surface intersects voxel ABCD if l_∞ iso-distance cube is smaller than the voxel

Outline

- l_∞ Distance Computation
  - Optimization Framework
  - Specialized Algorithms
- Complex Models
  - Bounding Volume Hierarchy
  - Graphics Hardware Approach

Optimization Framework



Minimize x

subject to

q lies on the primitive

q lies within R

Non-linear optimization

R – x+ dominating region

Convex Primitives

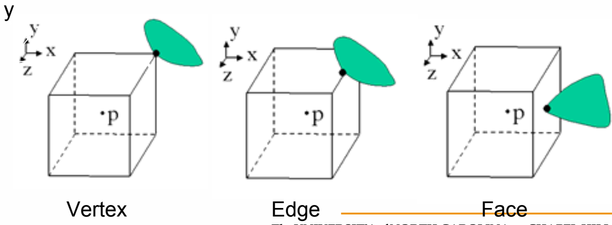
- Non-linear optimization reduces to convex optimization
- Simpler solution when the query point is inside the primitive

Outline

- l_∞ Distance Computation
  - Optimization Framework
  - Specialized Algorithms
    - Convex Primitives
    - Algebraic Primitives
    - Triangulated Models
- Complex Models

Algebraic Primitives

- Equation solving approach
- Applicable to convex and non-convex primitives
- Solve for the closest point, x



Equation Solving

- Solve above equations for each vertex, edge and face
- Solution set is finite in general
- Obtain a set X of feasible values for the closest point
- Calculate min { $||x-p||_\infty$ | x є X}

Equation Solving

- **Quadrics**
  - Quadratic Equation
- **Torus**
  - Symmetry
  - Degree 8 polynomial

Outline

- l_∞ Distance Computation
  - Optimization Framework
  - Specialized Algorithms
    - Convex Primitives
    - Algebraic Primitives
    - Triangulated Models
- Complex Models

Distance Computation for a Triangle

- Optimization framework applied to the special case of a triangle
- Split the triangle with respect to the partitioning triangles

Bounding Volume Hierarchy

- Large polyhedral model
- Naïve algorithm
  - Minimum over distance to each triangle
- Speed it up using a precomputed bounding volume hierarchy

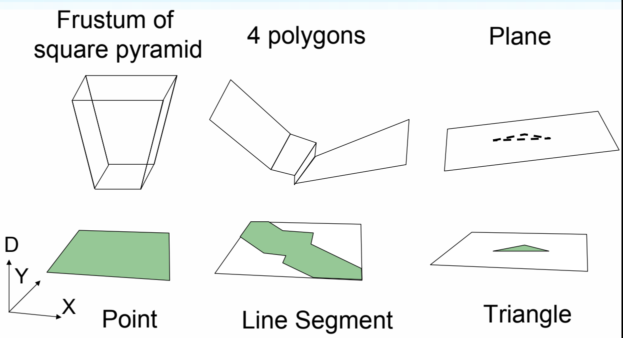
Outline

- l_∞ Distance Computation
  - Optimization Framework
  - Specialized Algorithms
    - Convex Primitives
    - Algebraic Primitives
    - Triangulated Models
- Complex Models
  - Bounding Volume Hierarchy
  - Graphics Hardware Approach

Graphics Hardware Approach

- Approach similar to [Hoff et al. 1999]
- Render distance function for each primitive
- Z-buffer holds the distance field

Linear Distance Functions for l_∞ Computations



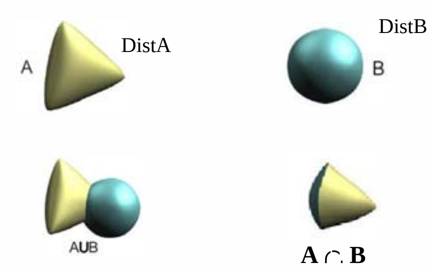
Organization

- Fast distance field computation
- Max-norm based voxelization
- **Boundary reconstruction**
- Analysis
- Applications
  - Boundary evaluation
  - Swept volume computation
  - Medial axis computation
  - Minkowski sums

Surface Reconstruction

- Objective – obtain a triangular mesh representation
- To extract the surface
  - Compute the zero-set { p | D(p) = 0 }

Boolean Operations



Min { DistA, Dist B} == 0

Max { DistA, Dist B} == 0

Isosurface Extraction

- Marching Cubes [Lorensen & Cline 87]
- Extended Marching Cubes [Kobbelt et al. 01]
- Dual Contouring [Ju et al. 02]
- Extended Dual Contouring [Varadhan et al. 03]

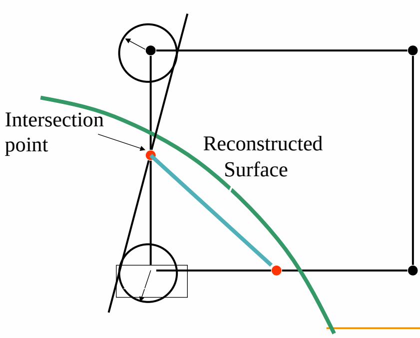
The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Marching Cubes

- Given the distance field grid,
  - Reconstruct the surface within each grid cell
- Once done with one cell (cube), march to the next

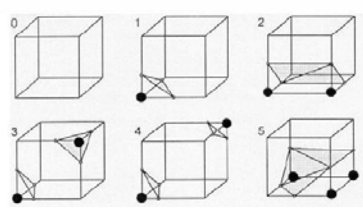
The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Marching Cubes



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Marching Cubes



The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Marching Cubes

- Handle each cell independently
- Because intersection points along grid edges are consistent between adjacent cells
  - Reconstructed surface matches at cell boundaries and doesn’t leave holes

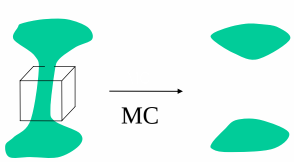
Our Approach

1. Generate distance field D for the union
2. Obtain an approximation by extracting an
   - Isosurface { p | D(p) = 0 }

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

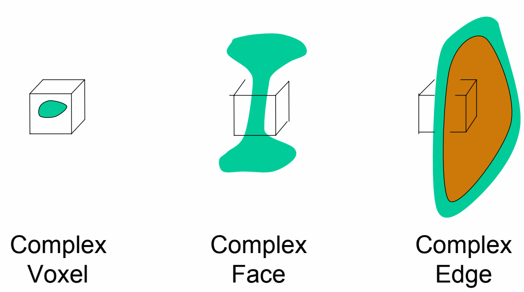
Issues

- Accuracy of the algorithm dependent on resolution of the underlying grid
  - Insufficient resolution can result in unwanted handles or disconnected components



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Complex Cells



The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Complex Cells

- How do you detect them?
  - Solution: Max-Norm Distance Computation

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Complex Cells

- Express voxel, face and edge intersection tests in terms of 3D, 2D and 1D max-norm distance respectively.
- A voxel, face, or edge is *complex* if it is intersecting but does not exhibit a sign change (i.e., a different in the outside/inside status)

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Complex cells

- Once detected, how do you handle them?
  - Subdivide them

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Issues

- Many cells in the grid do not contain a part of the final surface
  - Cull them away
- For each grid cell, first perform the voxel intersection test
- If the test fails, do not consider the voxel any further
- Makes the algorithm *output-sensitive*

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Issues

- Large number of primitives
- Each distance and outside/inside query defined in terms of all the primitives

Local Queries

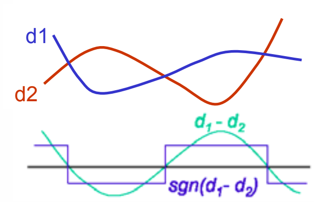
- Perform a local query within each cell by considering only the primitives intersecting the cell
  - Preserves correctness of the query
  - Drastically improves performance

Sharp Features

- Surface-surface intersection causes many sharp features on the boundary of the final surface
- When do two surfaces S1 and S2 intersect each other?
  - Track the bisector surface d1-d2, where d1, d2 are the distance functions for the two surfaces [Varadhan et al. 03]

Bisector Surface

- Bisector surface (d1-d2) contains the intersection curve
- It changes sign at intersection
  - Track sgn(d1-d2)



Grid Generation

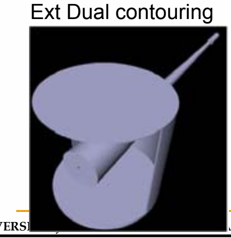
- Can reconstruct atmost one sharp feature per voxel
- Subdivide voxels with more than one sharp feature

Reconstruction algorithm

- Extended dual contouring algorithm [Varadhan et al. 03]
  - can reconstruct arbitrary thin features without creating handles

Dual contouring

Ext Dual contouring



Organization

- Fast distance field computation
- Max-norm based voxelization
- Boundary reconstruction
- **Analysis**
- Applications
  - Boundary evaluation
  - Swept volume computation
  - Medial axis computation
  - Minkowski sums

Bounds on Approximation

Let **S**: exact answer of the union or envelope computation
B(**S**): boundary of S

.

Our approximation algorithm takes as input $\varepsilon > 0$, and generates an approximation $A(\varepsilon)$

$B(A(\varepsilon))$: denote the boundary of the approximation

Bounds on Approximation

Theorem 1: Given any $\varepsilon > 0$, our algorithm computes an approximation $B(A(\varepsilon))$ such that

2-Hausdorff($B(A(\varepsilon))$, $B(S)$) $< \varepsilon$,

where 2-Hausdorff is the two sided Hausdorff distance

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Bounds on Approximation

Theorem 2:Given any $\varepsilon > 0$, our algorithm computes an approximation $A(\varepsilon)$ to the exact union or envelope **S** such that $A(\varepsilon)$ has the same number of connected components as **S**

Corollary: S is connected if and only if $A(\varepsilon)$ is connected

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

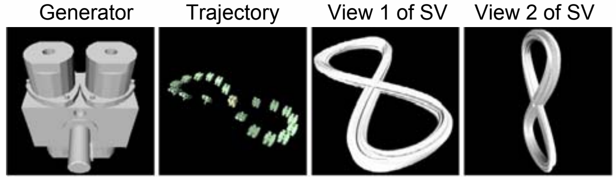
Organization

- Fast distance field computation
- Max-norm based voxelization
- Boundary reconstruction
- **Analysis**
- Applications
  - Boundary evaluation
  - Swept volume computation
  - Medial axis computation
  - Minkowski sums

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Swept Volume Computation

Generator | Trajectory | View 1 of SV | View 2 of SV



2,280 triangles | 1,152 surfaces | Time = 12 secs

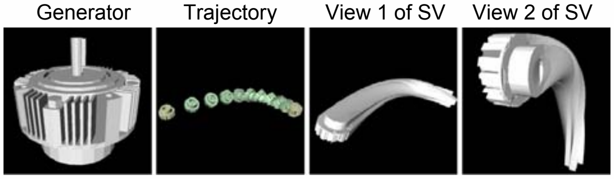
[Kim et al. 2003]
http://gamma.cs.unc.edu/SV

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Results: Swept Volume

Input Clutch Model

Generator | Trajectory | View 1 of SV | View 2 of SV



2,116 triangles

1,175 surfaces

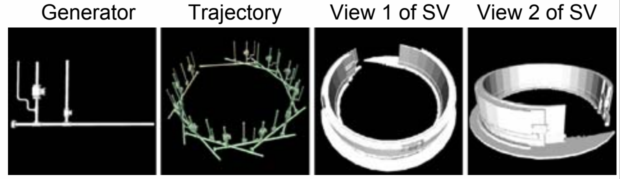
Time = 21 secs

[Kim et al. 2003]
http://gamma.cs.unc.edu/SV

Results

Pipe Model

Generator | Trajectory | View 1 of SV | View 2 of SV



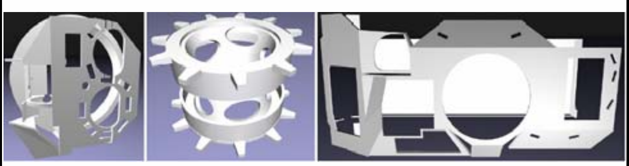
10,352 triangles

15,554 surfaces

Time = 67 secs

Boundary Evaluation of Complex CSG Models

30-40 solids defined using 2-7 Boolean operations
8-13 secs per solid



Turret | Drivewheel | Hull

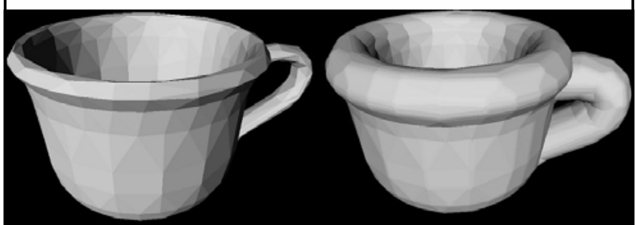
[Varadhan et al. 2003]
http://gamma.cs.unc.edu/recons

Boundary Evaluation of Complex CSG Models

Bradley Fighting Vehicle
1200 solids
8,000 CSG operations
Took 2 hours

[Varadhan et al. 2003]
http://gamma.cs.unc.edu/recons

Cup: Offset Computation

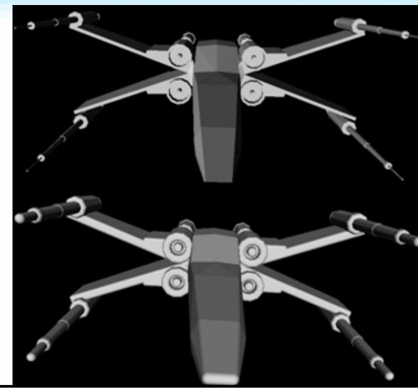


1000 triangles:
338 convex pieces

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

Xwing: Offset Computation

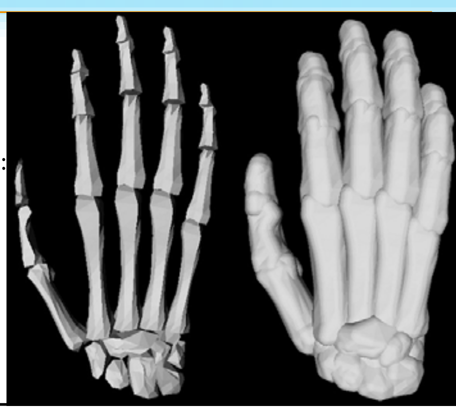
2496 triangles:
1294 convex
pieces



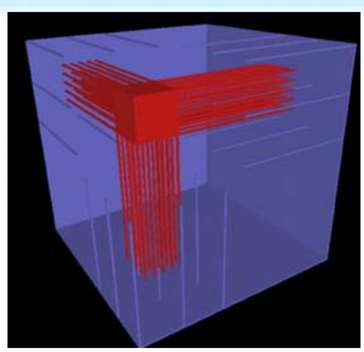
HILL

Hand: Offset Computation

2982 triangles:
910 convex
pieces



Minkowski Computation

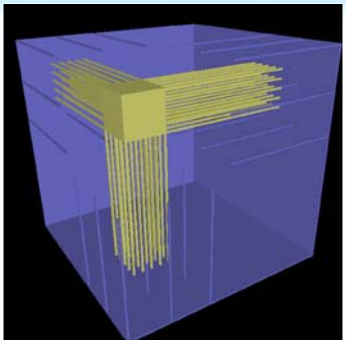


Non-convex polyhedra

Red polyhedra: 1134 polygons
Blue polyhedra: 444 polygons

The UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

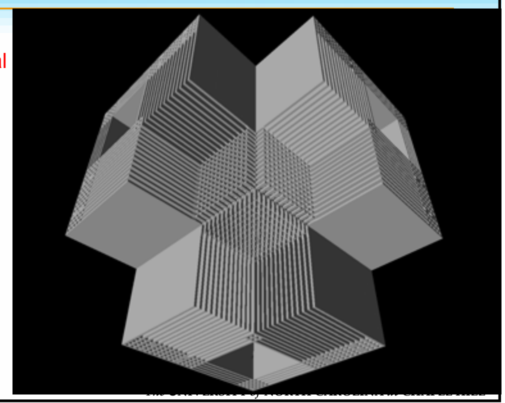
Minkowski Computation



Non-convex polyhedra
Yellow polyhedra: 1134 polygons
Blue polyhedra: 444 polygons

Worst Case Minkowski Sum

$O(n^6)$
Combinatorial complexity



Application to Continuous Collision Detection

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

References

- A. Sud and D. Manocha. "DiFi: Fast distance field computation using graphics hardware". *UNC-CH Computer Science Technical Report TR03-026,* 2003 http://gamma.cs.unc.edu/DiFi
- M. Foskey, M. Lin, and D. Manocha. "Efficient computation of a simplified medial axis". *Proc. of ACM Solid Modeling*, 2003.
- Y. Kim, G. Varadhan, M. Lin and D. Manocha. "Fast approximation of swept volumes of complex models". *Proc. of ACM Solid Modeling, 2003.*
- G. Varadhan, S. Krishnan, Y. Kim and D. Manocha. "Feature-based Subdivision and Reconstruction using Distance Field". *Proc. Of IEEE Visualization, 2003 (to appear).*

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

Conclusions

- Discretized geometric computations
- Union and envelope comptuations
- Fast distance field computation
- Max-norm computation algorithms
- Application to medial, swept volume, Minkowski and CSG computations
- Use of GPUs for geometric computations
- Benefits
  - Improved performance
  - Robust implementations

Acknowledgements

- Army Research Office
- Intel
- National Science Foundation
- Office of Naval Research

The End