VR’05 Tutorial:
Real-Time Collision Detection for Dynamic Virtual Environments

Image-Space Techniques

Bruno Heidelberger

Computer Graphics Laboratory
ETH Zurich, Switzerland
Basic idea

- Exploit rasterization of object primitives as intersection test
- Benefit from graphics hardware acceleration
Early approaches

[Shinya, Forgue 1991]
Image-space collision detection for convex objects

[Myszkowski, Okunev, Kunii 1995]
Collision detection for concave objects with limited depth complexity

[Baciu, Wong 1997]
Hardware-assisted collision detection for convex objects
More approaches

[Lombardo, Cani, Neyret 1999]
Intersection of tool with deformable tissue
by rendering the interior of the tool

[Vassilev, Spanlang, Chrysanthou 2001]
Image-space collision detection
applied to cloth simulation

[Hoff, Zaferakis, Lin, Manocha 2001]
Proximity tests and later penetration
depth computation
Recent approaches

[Knott, Pai 2003]
Intersection of edges with surfaces

[Govindaraju, Redon, Lin, Manocha 2003]
Object and sub-object pruning based on occlusion queries

[Heidelberger, Teschner, Gross 2004]
Explicit intersection volume and self-collision detection based on LDIs
Layered depth images

- Compact, volumetric object representation
  [Shade et al. 1998]

![Diagram of Layered Depth Image]

- Layered Depth Image
- = entry point
- = exit point
Algorithm

<table>
<thead>
<tr>
<th>Stage 1</th>
<th>Stage 2</th>
<th>Stage 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume-of-interest</td>
<td>LDI generation</td>
<td>Collision query</td>
</tr>
</tbody>
</table>

- a) LDI intersection
- b) Vertex-in-volume
- c) Self-collision
**LDI generation**

- Object is rendered once for each LDI layer
- Two separate depth test per fragment are required:
  - Fragment has to be farther than the one in the previous layer
  - Fragment has to be the nearest of all remaining fragments

→ Shadow mapping functionality is a second depth test [Everitt 2001]
... or multiple depth textures and fragment shaders
Unordered LDI

- Fragments $z_1, \ldots, z_n$ are rendered in the same order in each rendering pass.
- Employ the stencil buffer to generate the $n$-th value in the $n$-th pass.

$\Rightarrow$ No depth test required, just stencil tests.
$\Rightarrow$ ~3x faster than ordered LDI approach!
Collision queries

Explicit intersection volume

Vertex-in-volume test
Self-collision query

- Check for incorrect ordering of front and back faces

→ Requires two passes (front-then back-faces) and sorting of depth values
Ongoing work

• Precision
  – Image resolution, depth precision and render direction introduce errors

• Collision information
  – Penetration depth and direction required for appropriate collision response

• GPU
  – Bottleneck: framebuffer readback
  – Integration with collision response, simulation, rendering etc.
Conclusion

• Image-space techniques exploit the rasterization of objects for collision and self-collision detection
• No pre-processing required
• Suitable for rigid and deformable objects
• Related image-based methods exist for collision culling, proximity tests and penetration depth computation
The End

Thank you!