

CS 277 - Experimental Haptics

Lecture 5

“Proxy-based rendering, implicit surfaces”



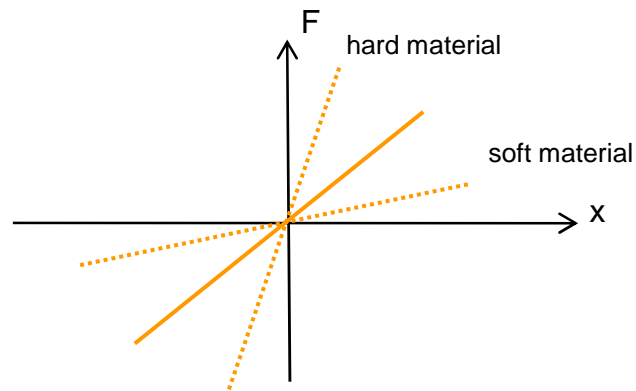
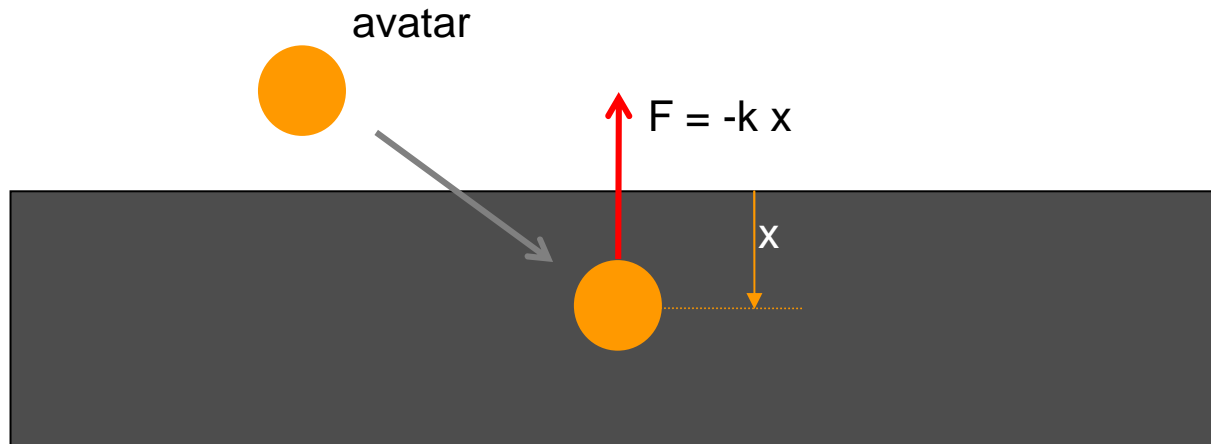
Outline

- Limitations when using Potential Fields
- God object algorithm
- Finger proxy algorithm
- Implicit surfaces
- Demonstrations

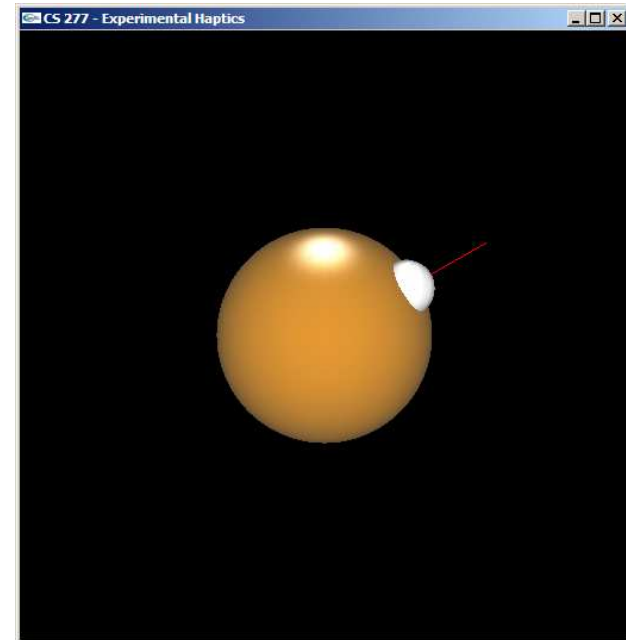
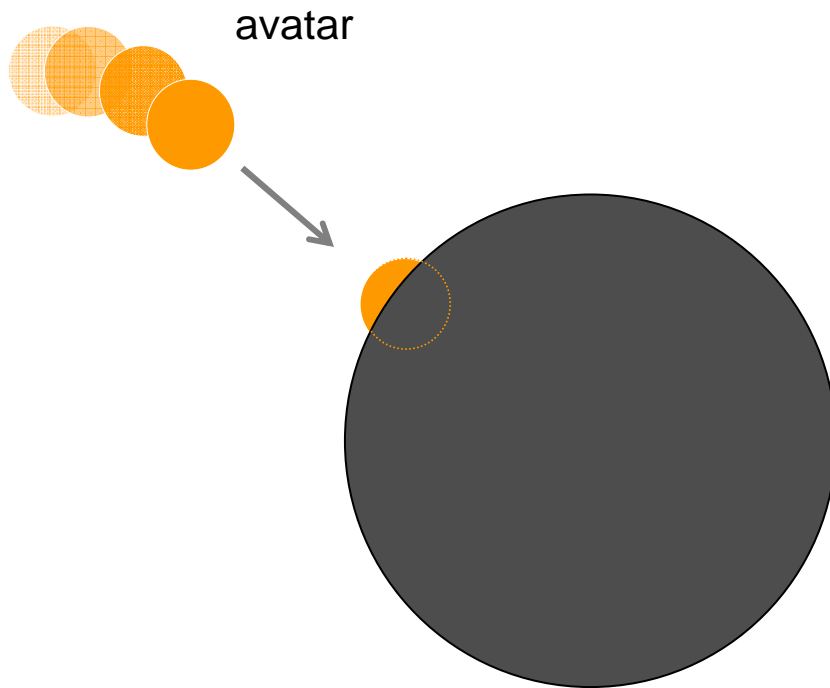
Outline

- Limitations when using Potential Fields
- God object algorithm
- Finger proxy algorithm
- Implicit surfaces
- Demonstrations

Potential Fields

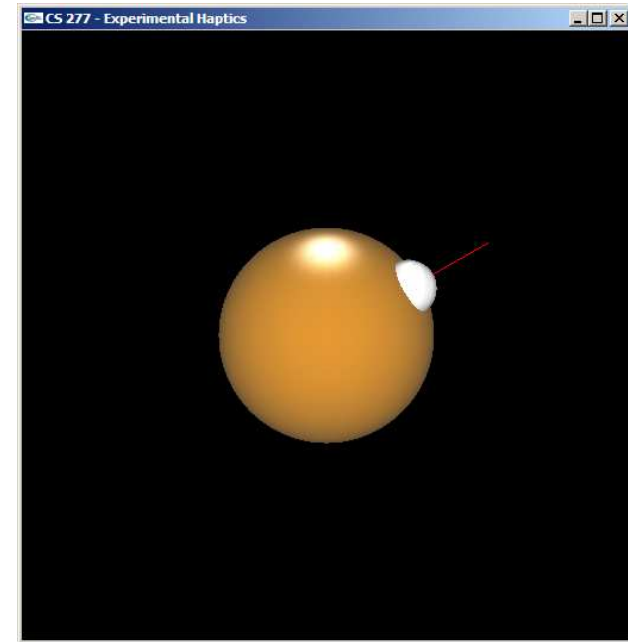
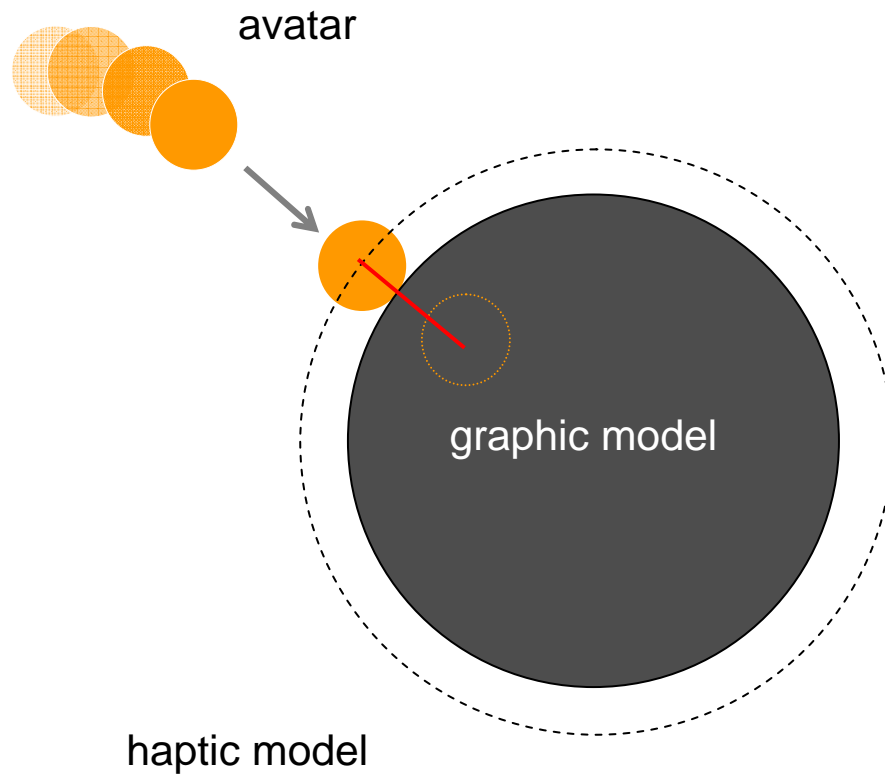


Potential Fields

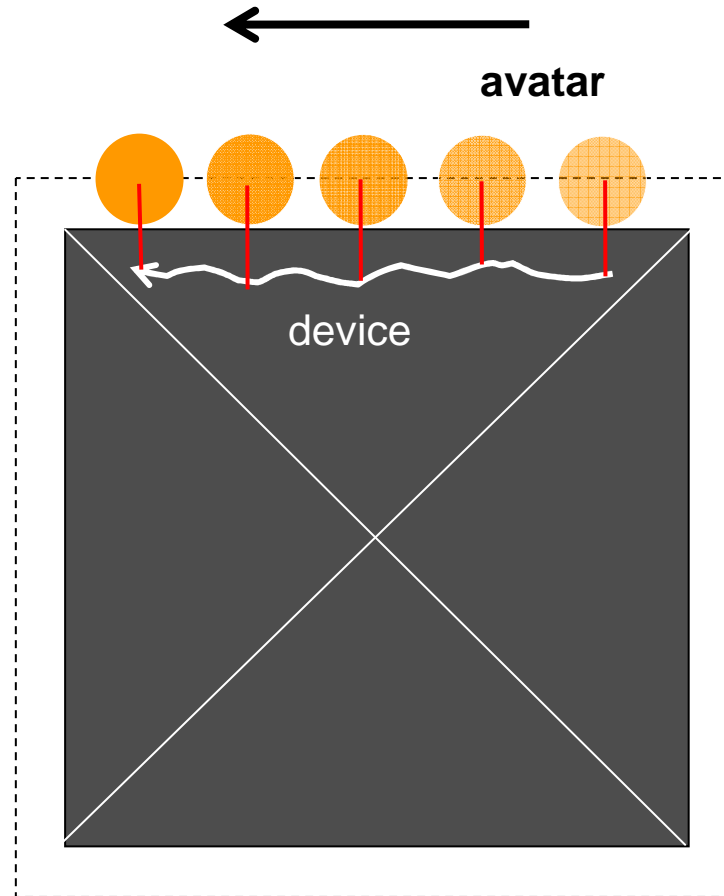


How to avoid sinking the avatar ?

Potential Fields

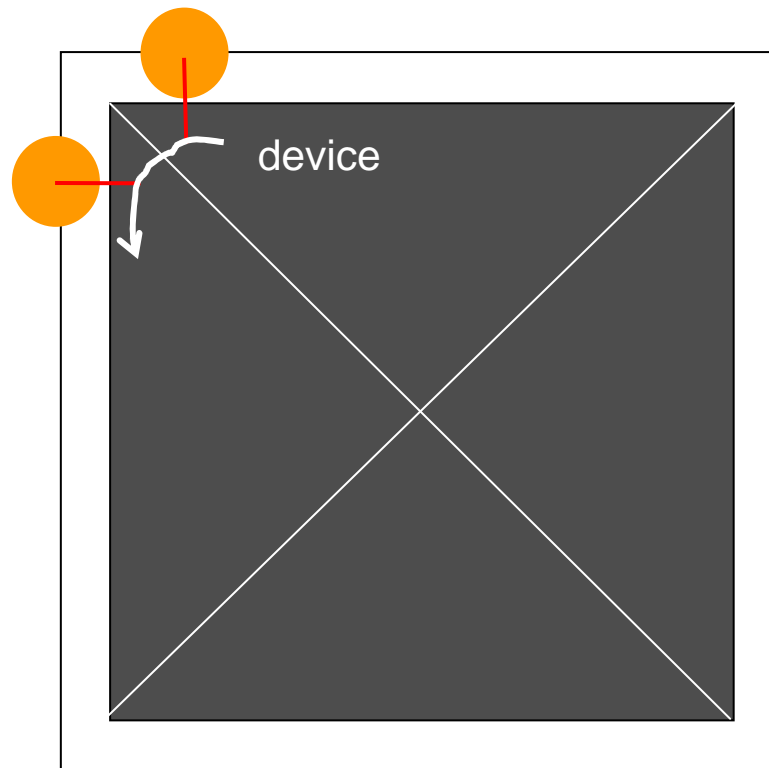


Potential Fields

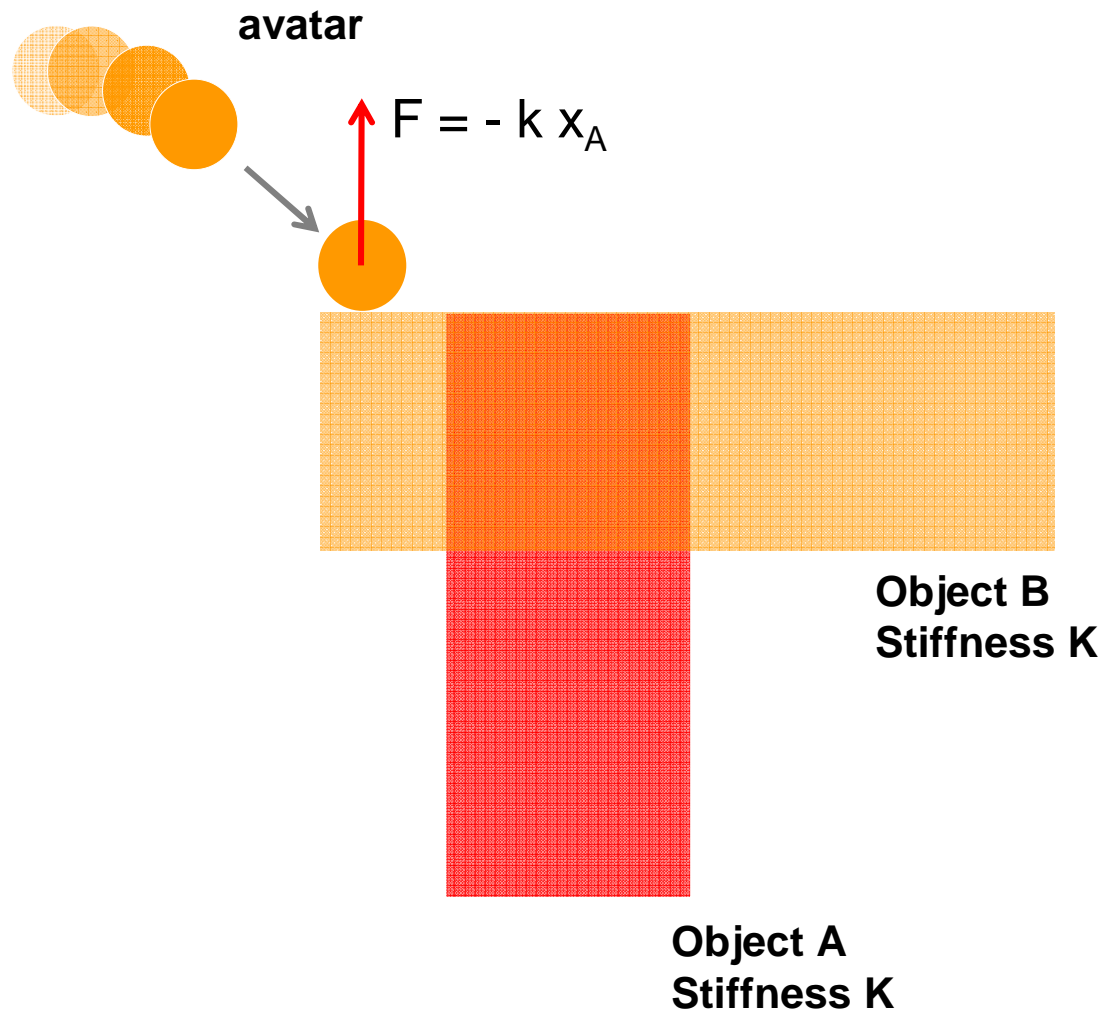


Potential Fields

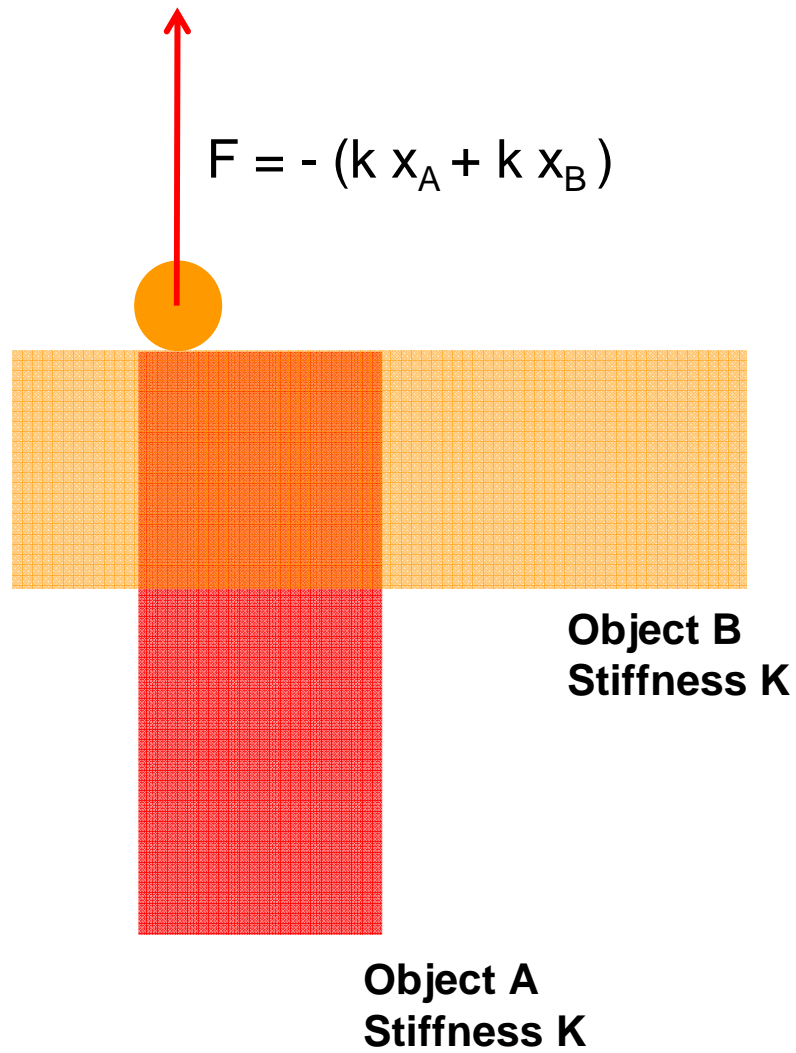
avatar “jump” !



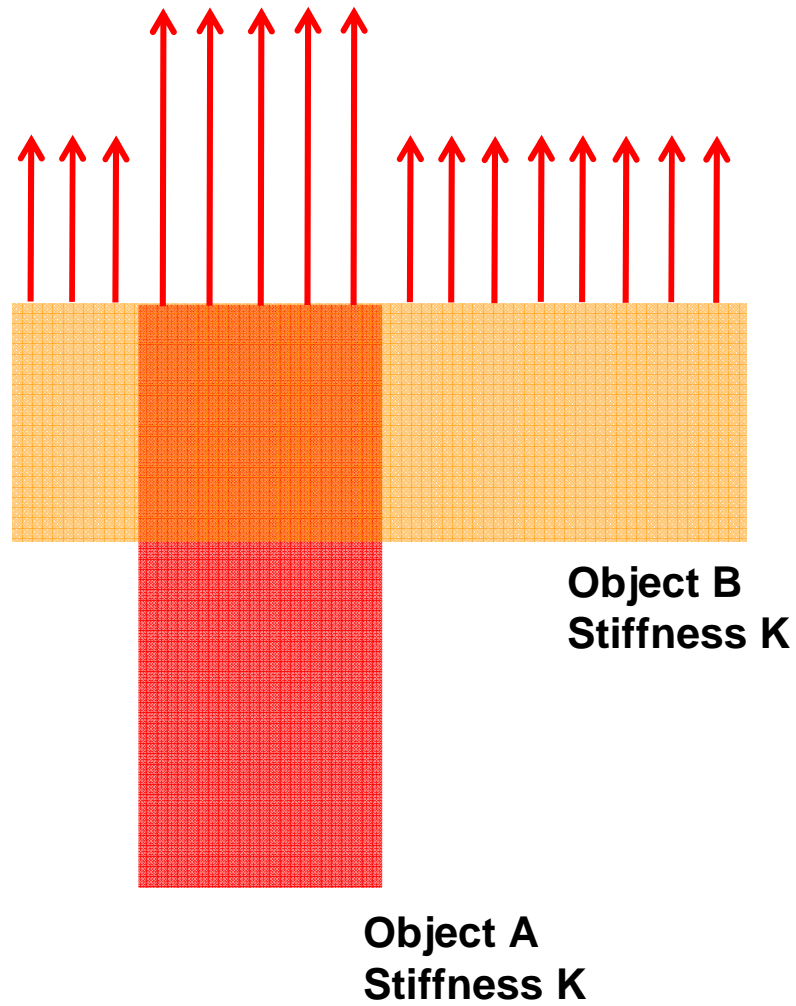
Stiffness Variation



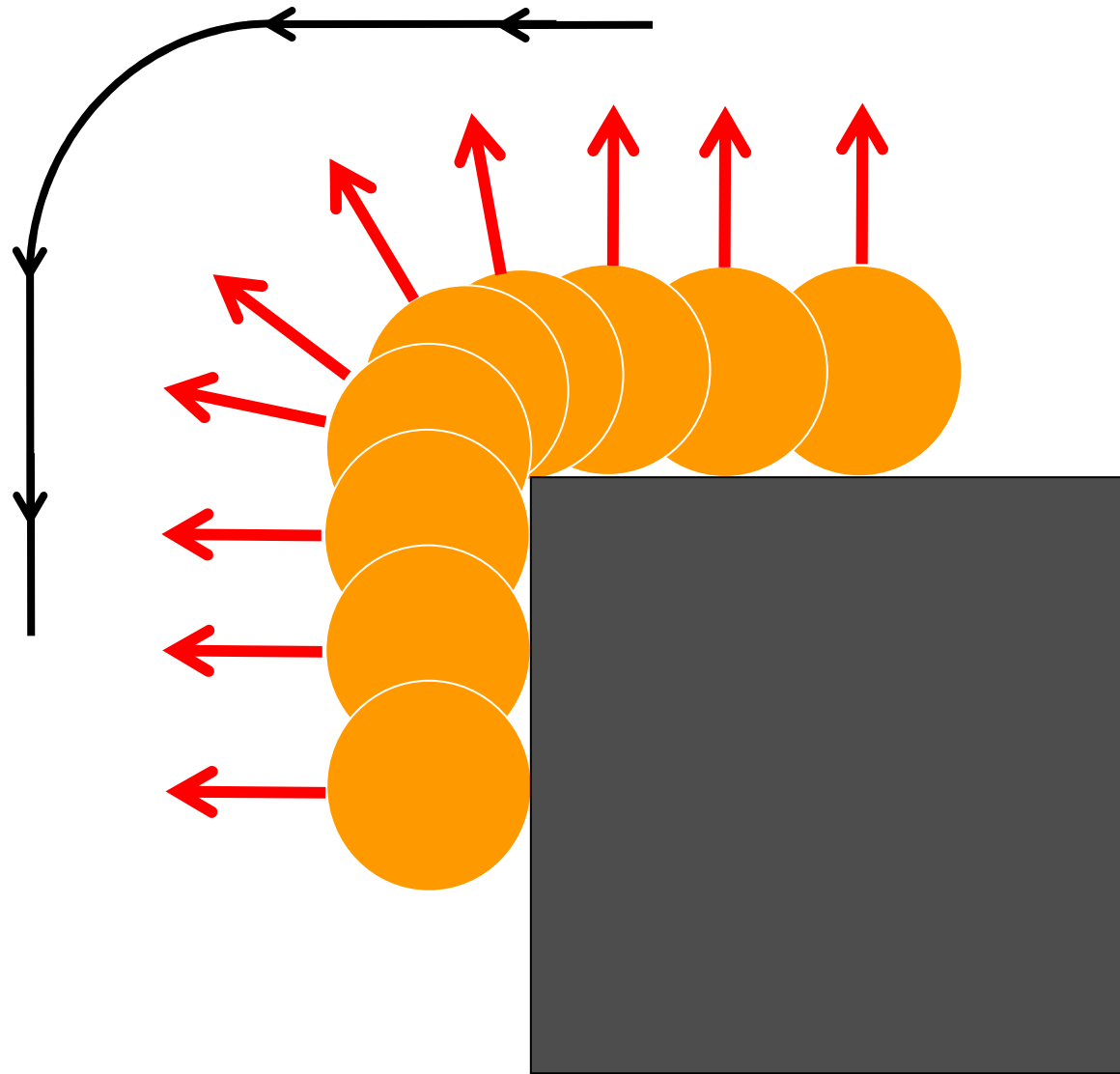
Stiffness Variation



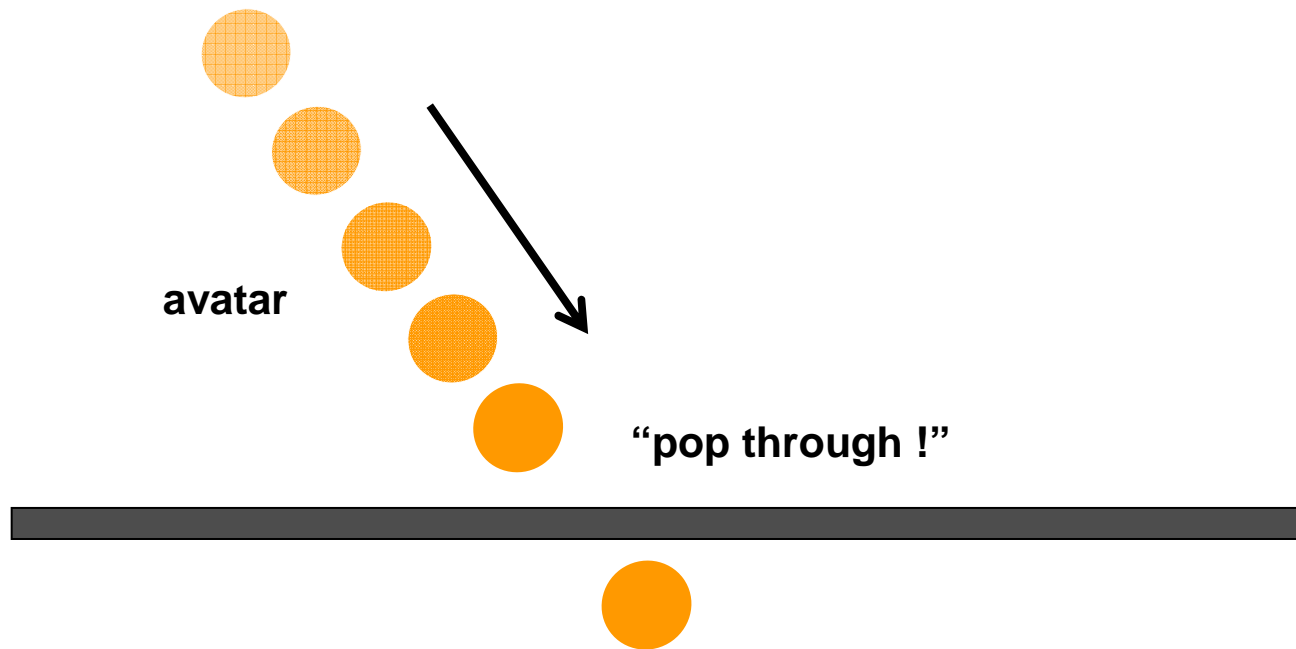
Stiffness Variation



Potential Fields



Potential Fields



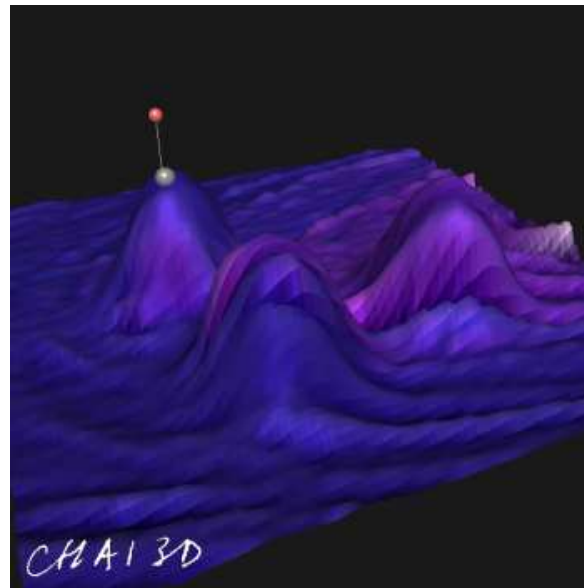
Outline

- Limitations when using Potential Fields
- God object algorithm
- Finger proxy algorithm
- Implicit surfaces
- Demonstrations

Haptic Rendering of Triangle Based Objects

“A Constraint-based God-object Method For Haptic Display”

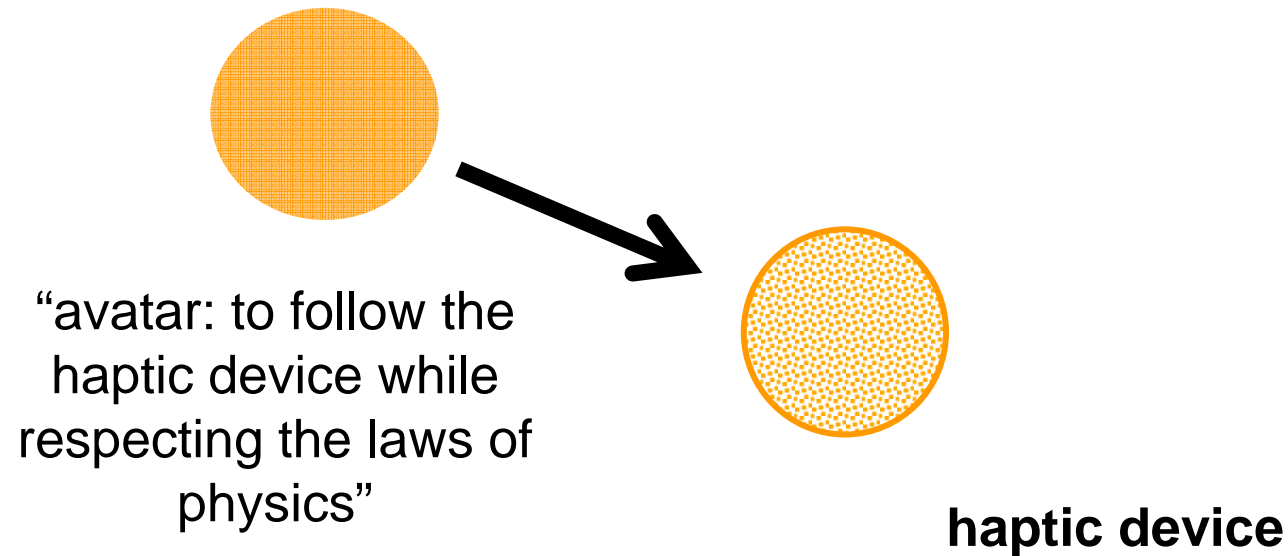
C.G.Zilles and J.K.Salisbury



Haptic Device and Avatar

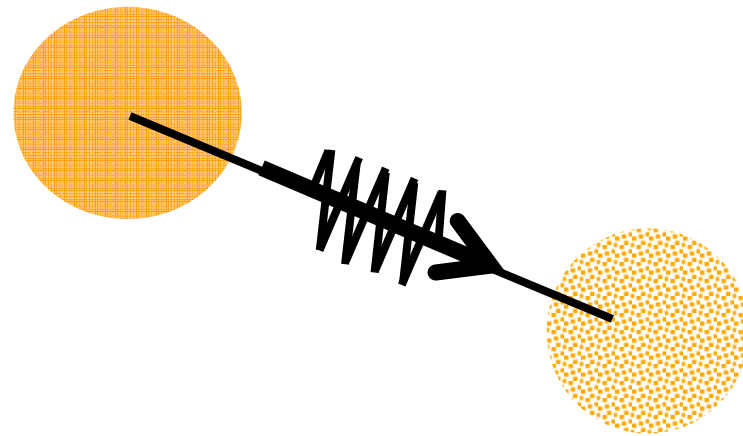
avatar

(tool, cursor, virtual finger, god object, proxy)



Connecting the Device to the Avatar

avatar

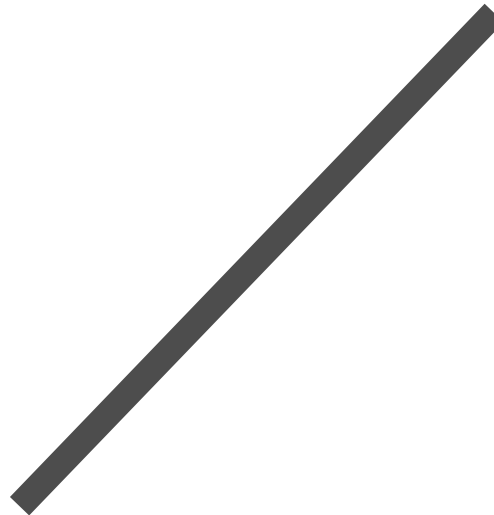
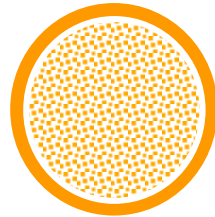


haptic device

God Object Algorithm

avatar / haptic device

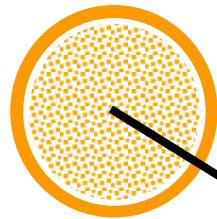
time: t



constraint

God Object Algorithm

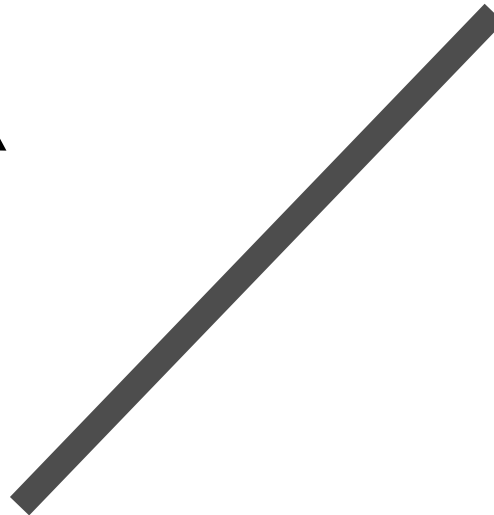
avatar
time: t



check for any collision
between the segment and the
VR environment

haptic device
time: $t + dt$

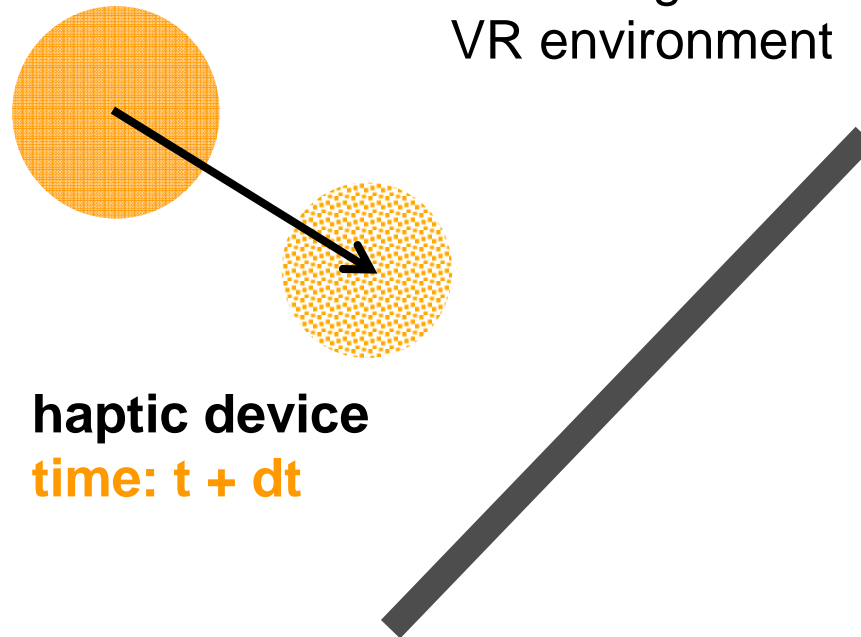
constraint



God Object Algorithm

avatar
time: t

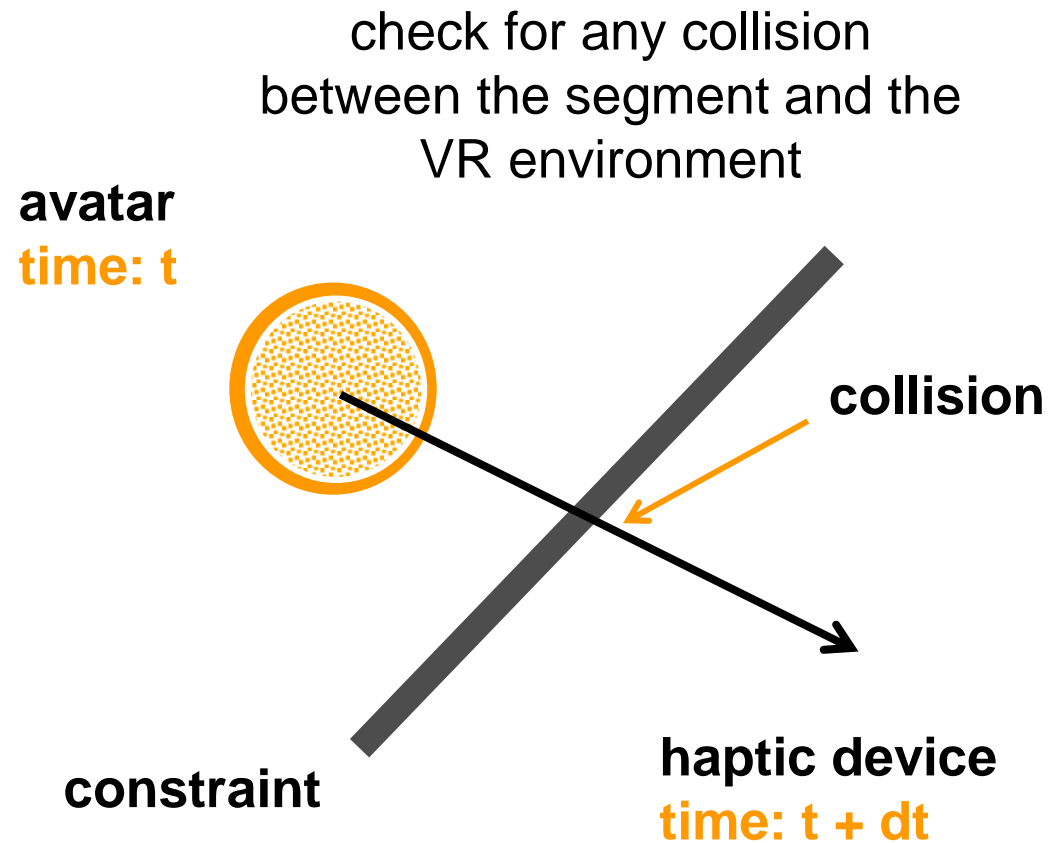
check for any collision
between the segment and the
VR environment



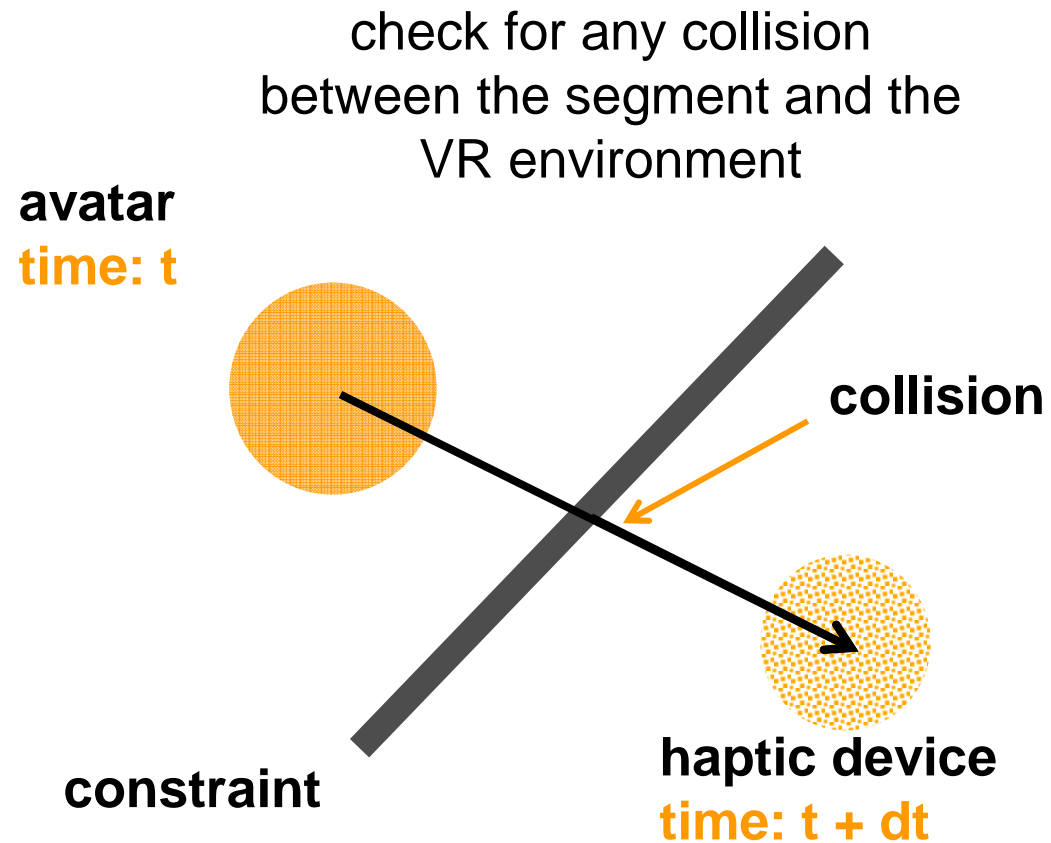
haptic device
time: $t + dt$

constraint

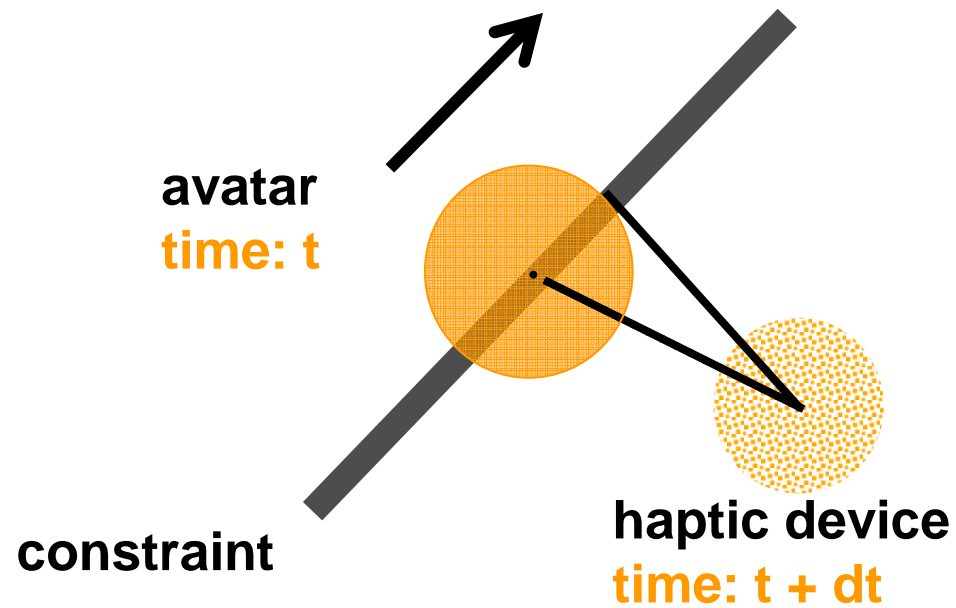
God Object Algorithm



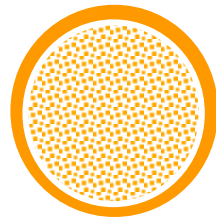
God Object Algorithm



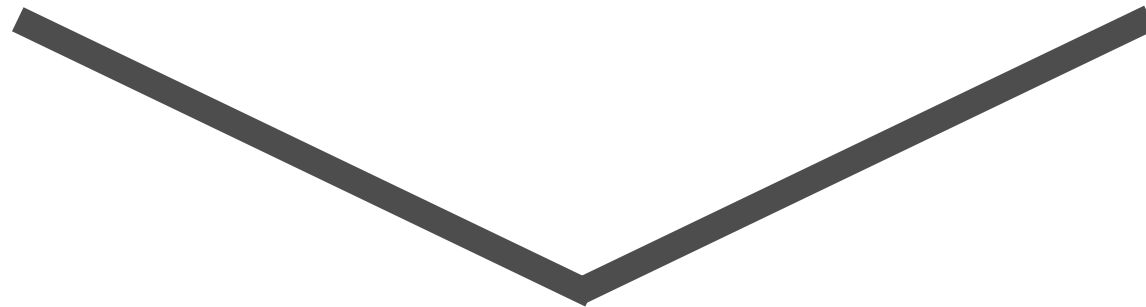
God Object Algorithm



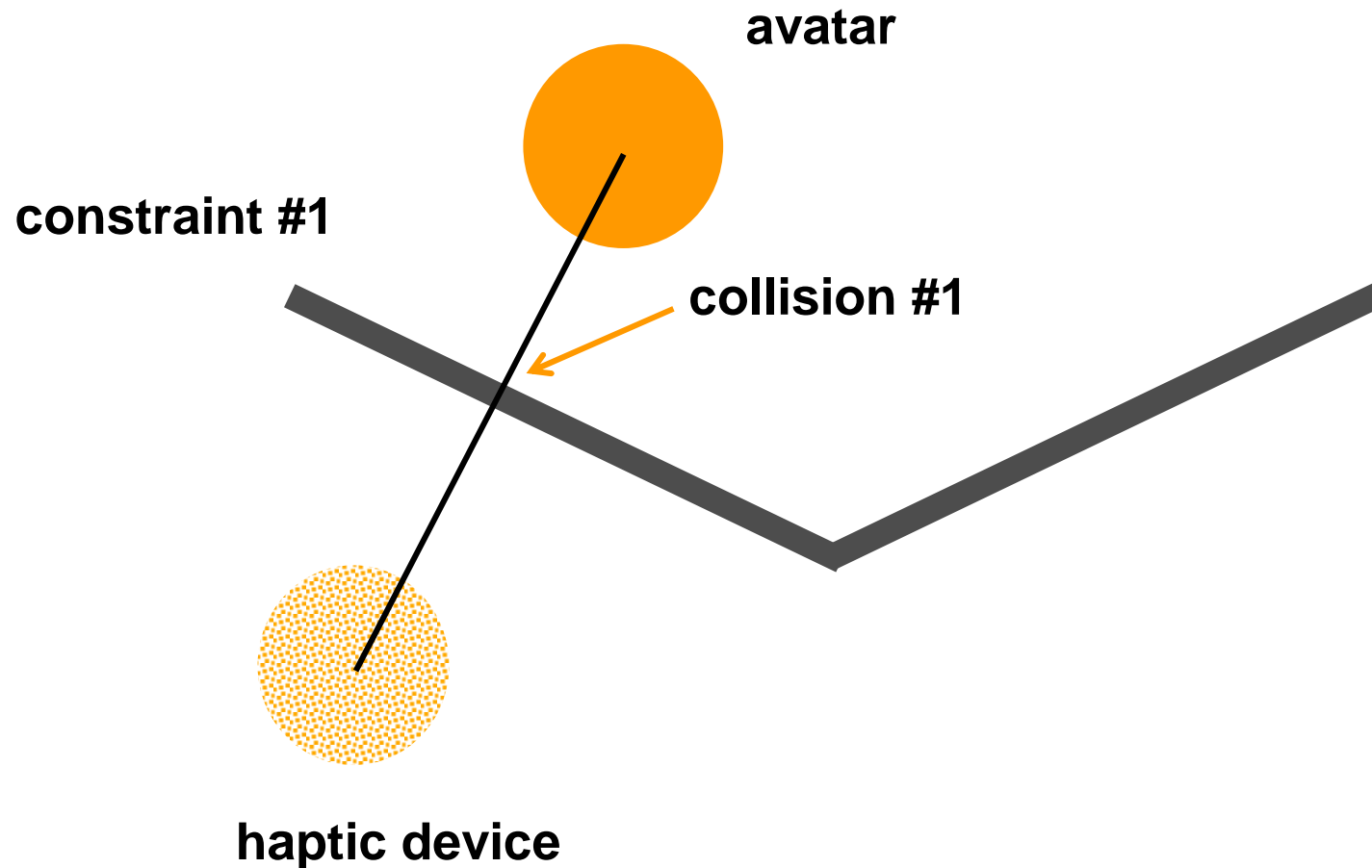
God Object Algorithm



avatar / haptic device

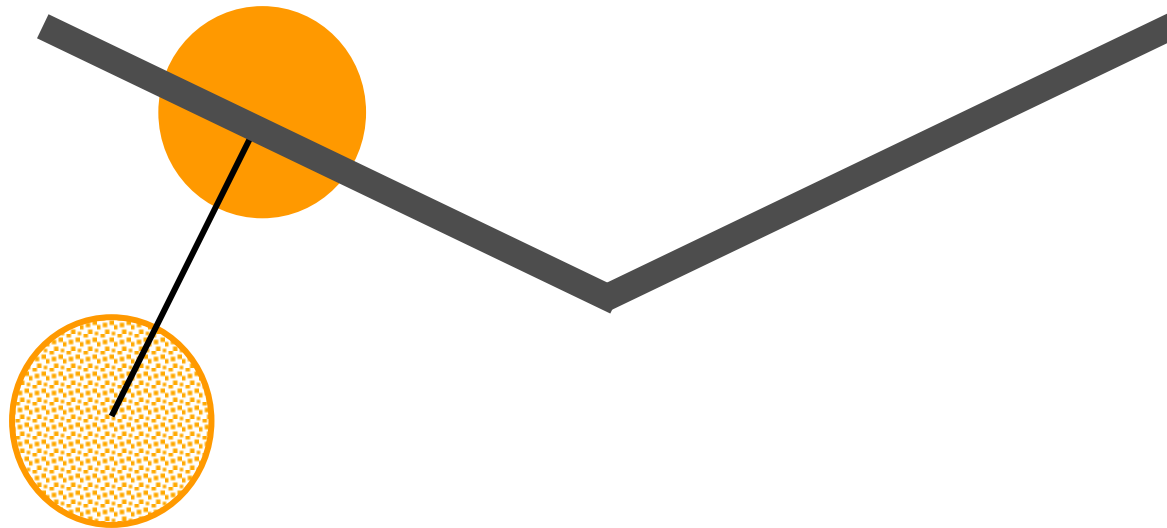


God Object Algorithm

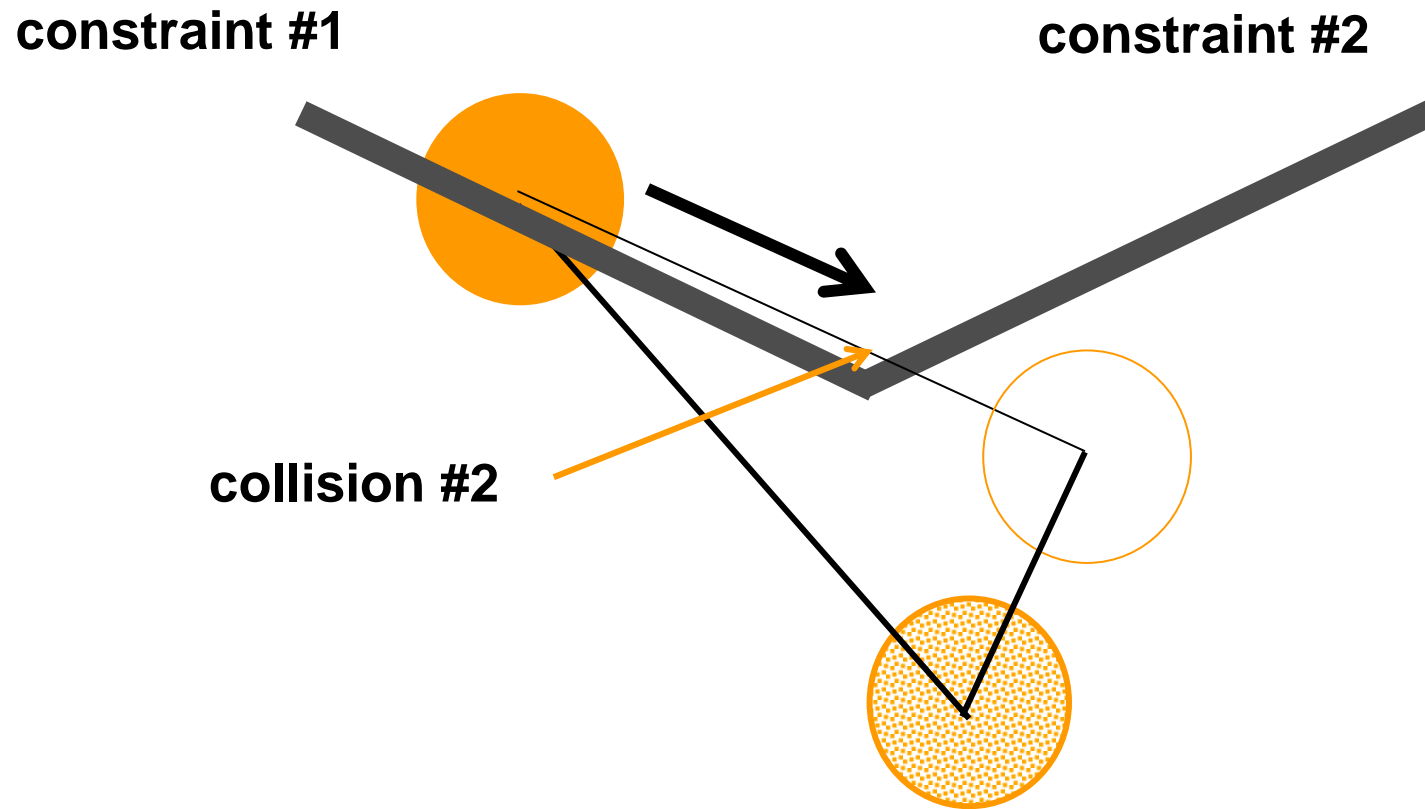


God Object Algorithm

constraint #1



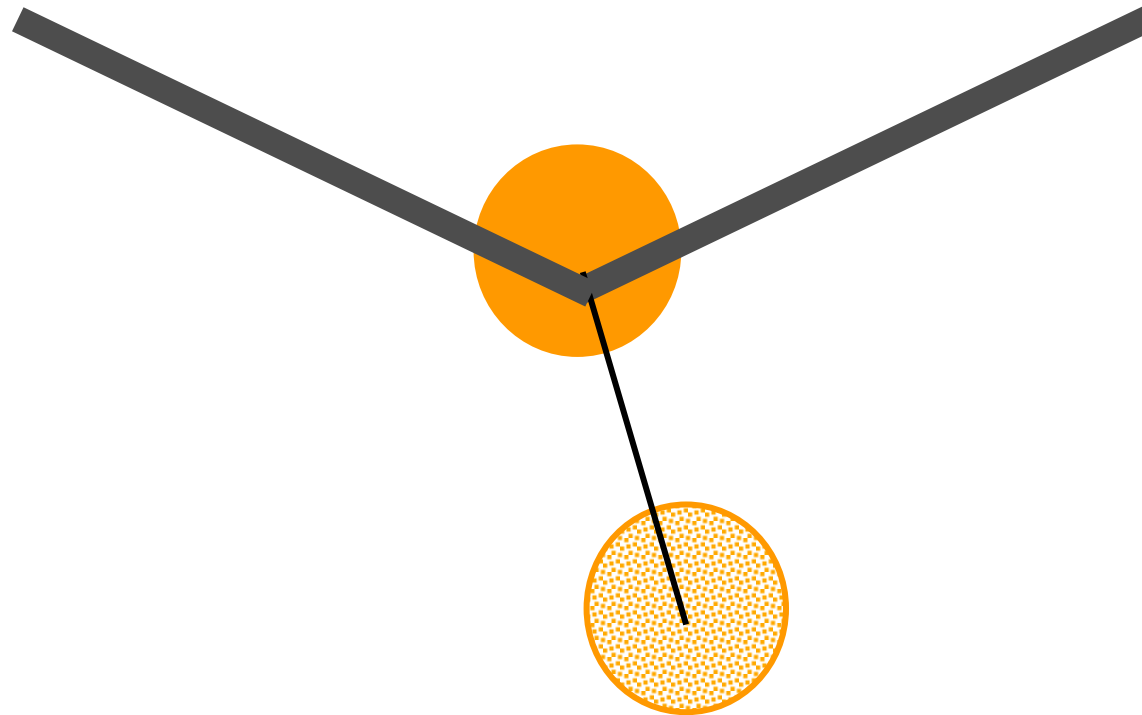
God Object Algorithm



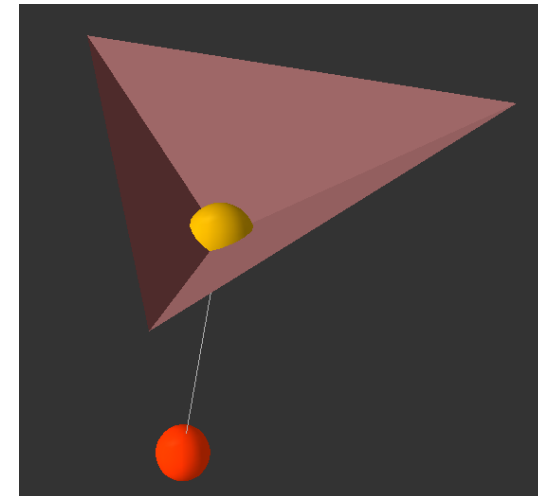
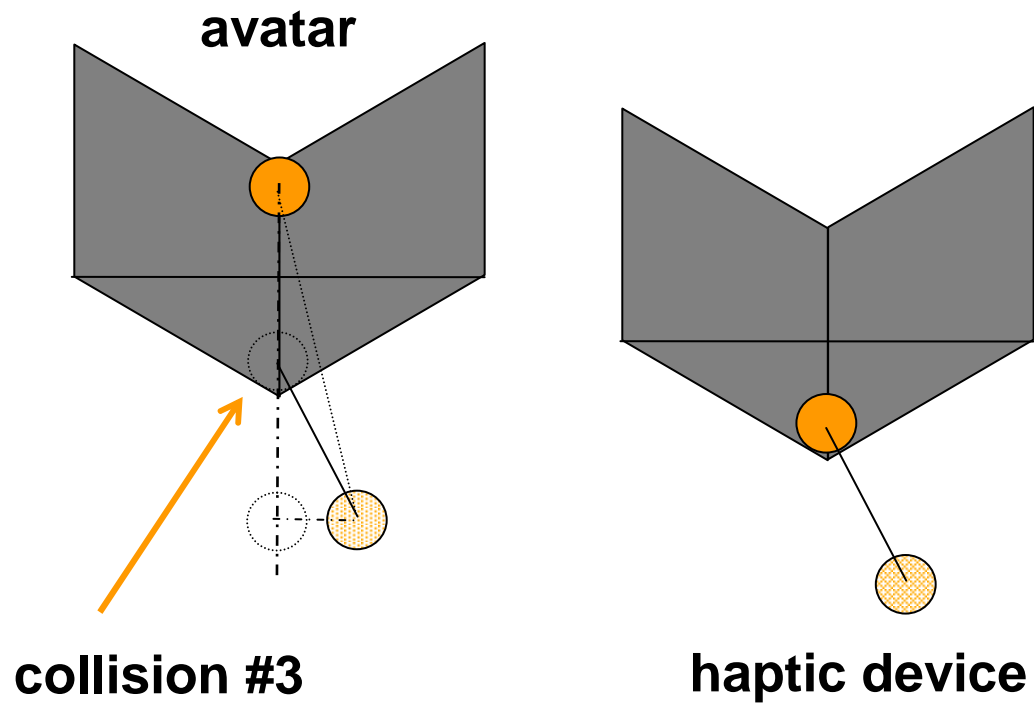
God Object Algorithm

constraint #1

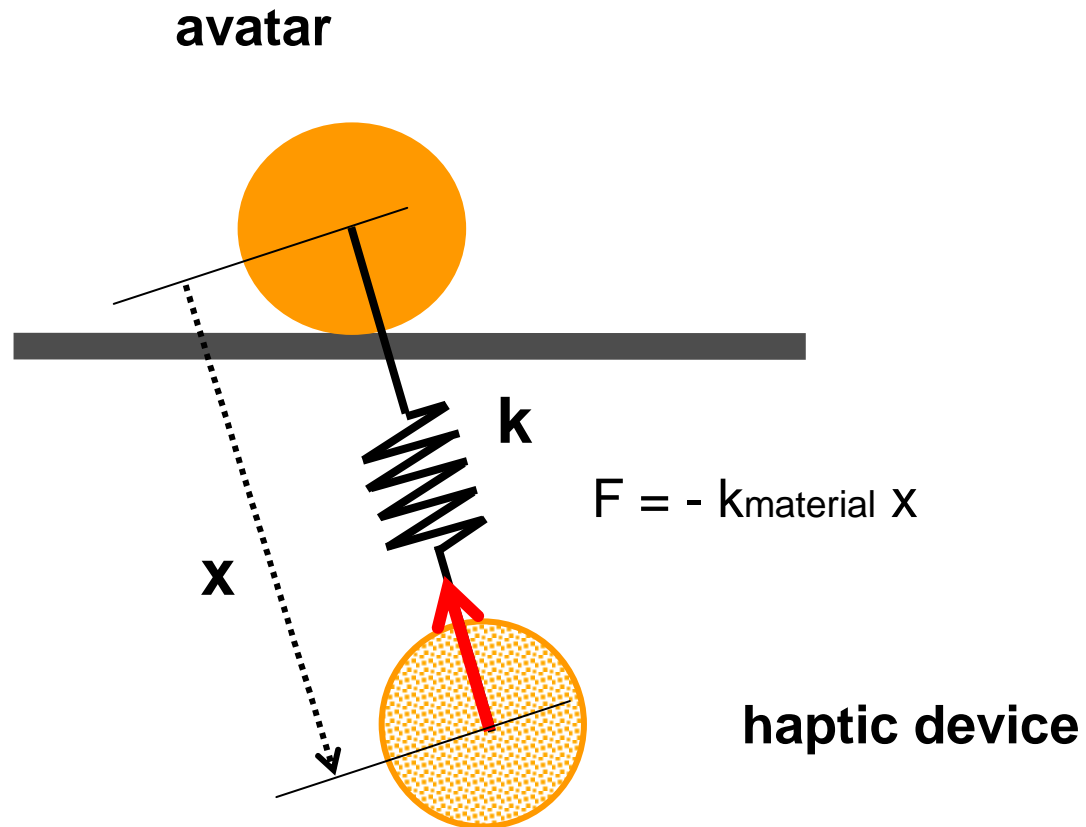
constraint #2



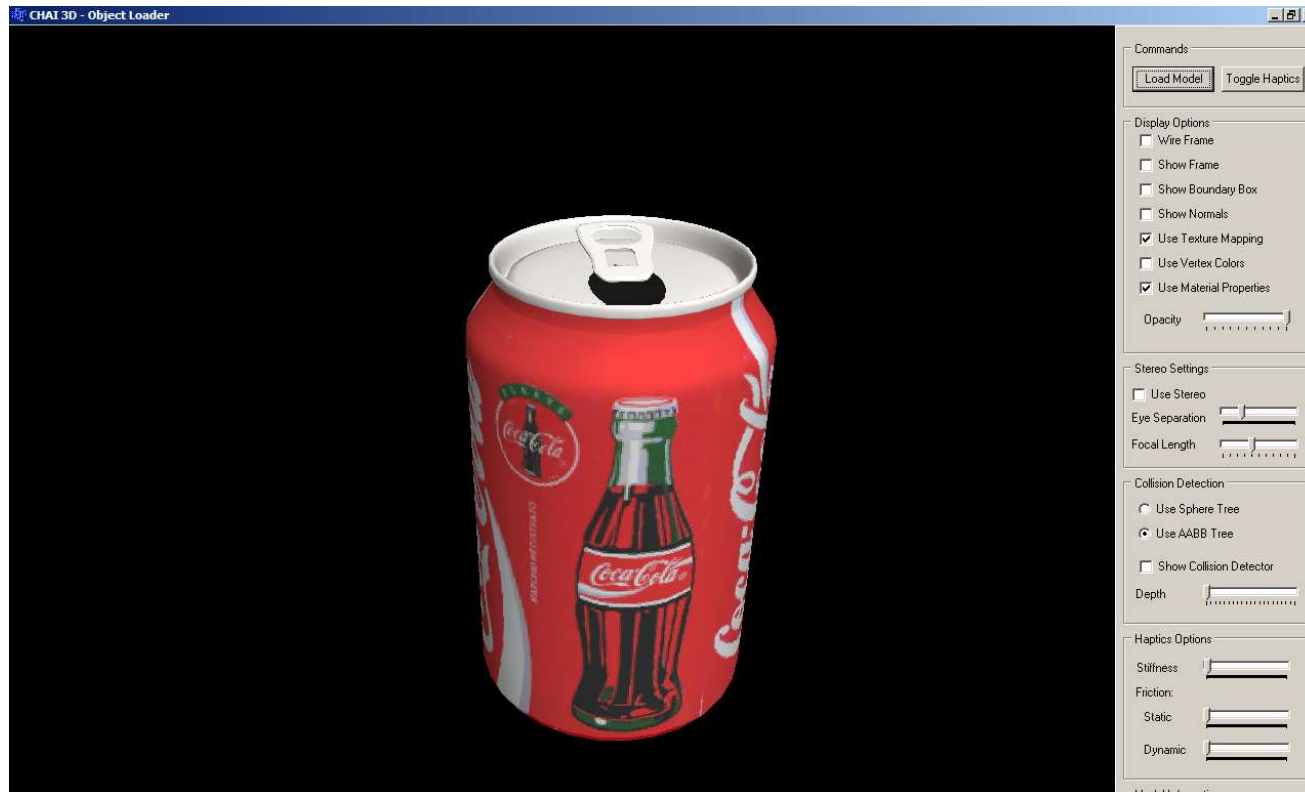
God Object Algorithm



Computing the Reaction Force



demonstration



Issues with the God object algorithm

- The avatar is modeled as a single point, and therefore half of the sphere (avatar) penetrates the object.
- Triangle meshes are rarely perfect and many hold ill conditioned triangles, or small gaps.
- It is not easy to know if the avatar is located inside the mesh. How can we define the notion of “inside” or “outside” ?
- The algorithm can be computationally expensive with objects composed of thousands of triangles.
- Higher complexity compared to the potential field approaches. (i.e a sphere object)

Haptic Rendering of Triangle Based Objects

“The Haptic Display of Complex Graphical Environments”

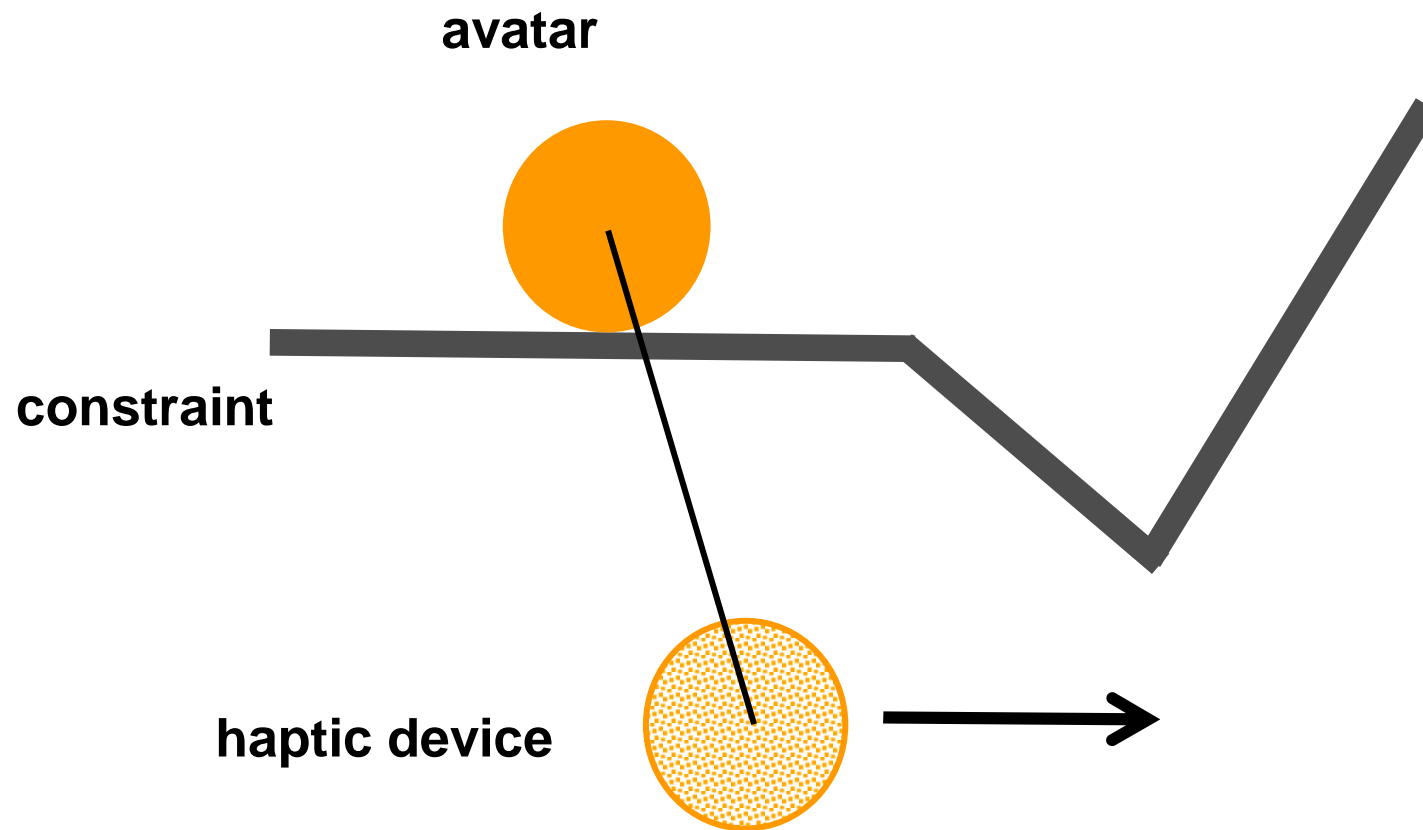
D. Ruspini, K. Kolarov and O. Khatib



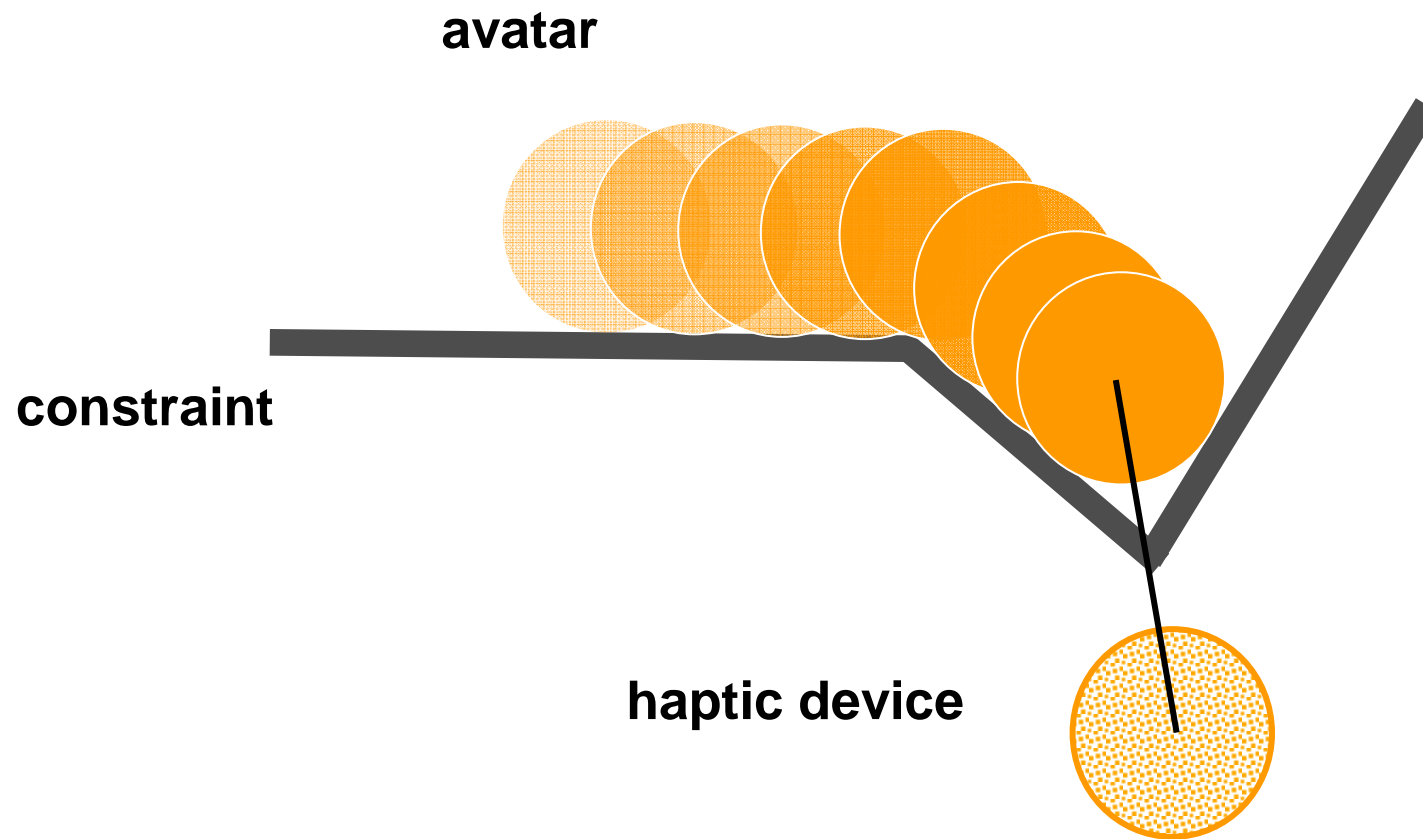
Outline

- Limitations when using Potential Fields
- God object algorithm
- **Finger proxy algorithm**
- Implicit surfaces
- Demonstrations

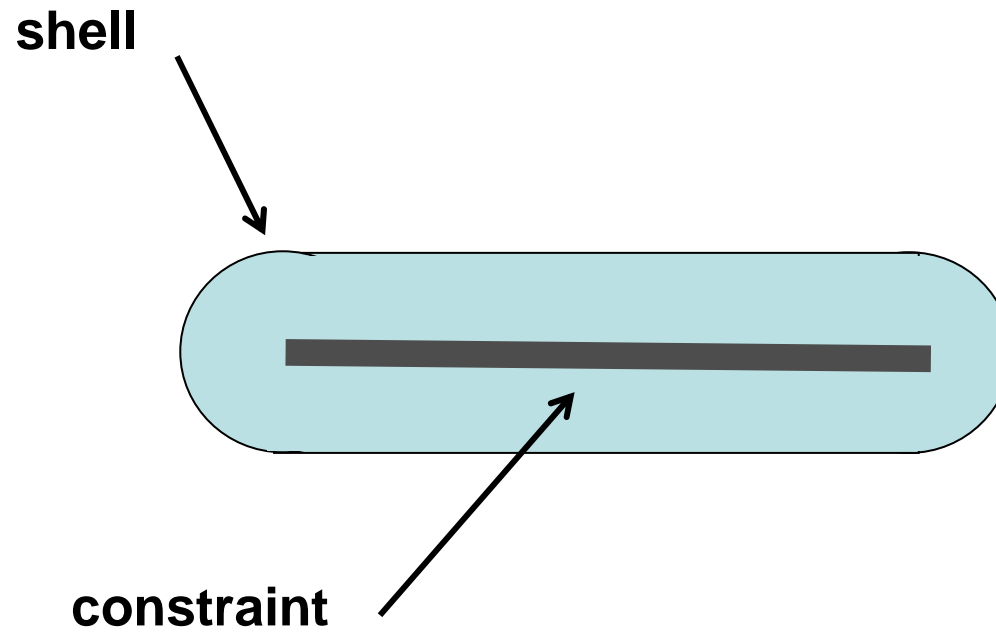
God Object Algorithm



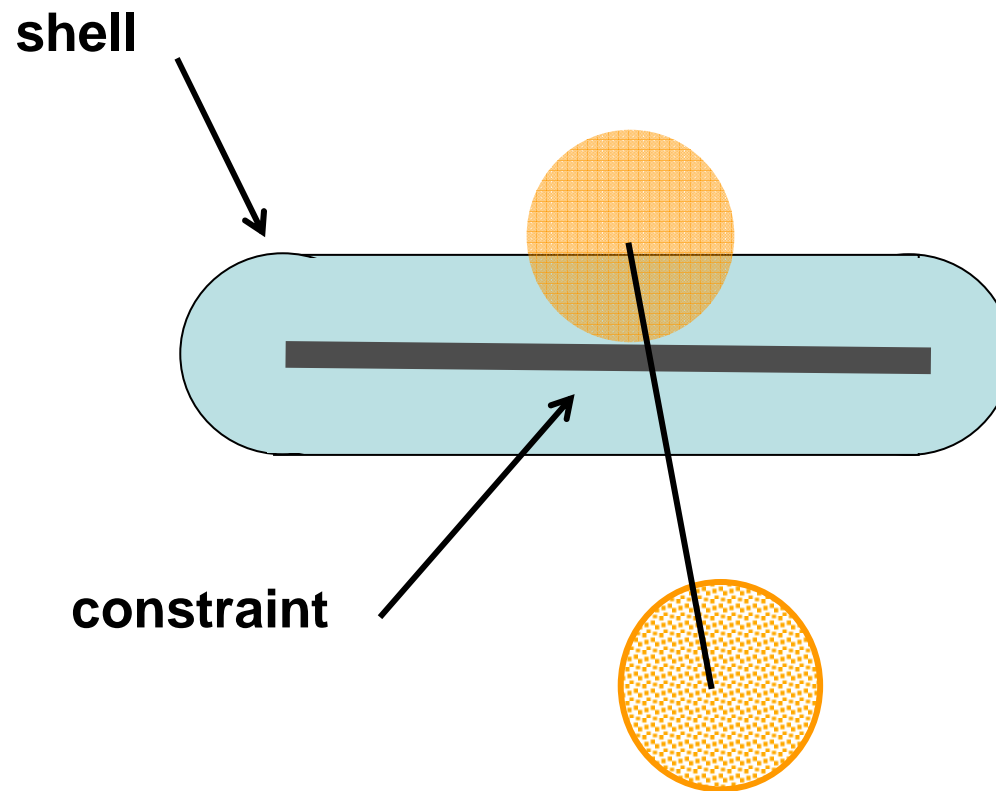
God Object Algorithm



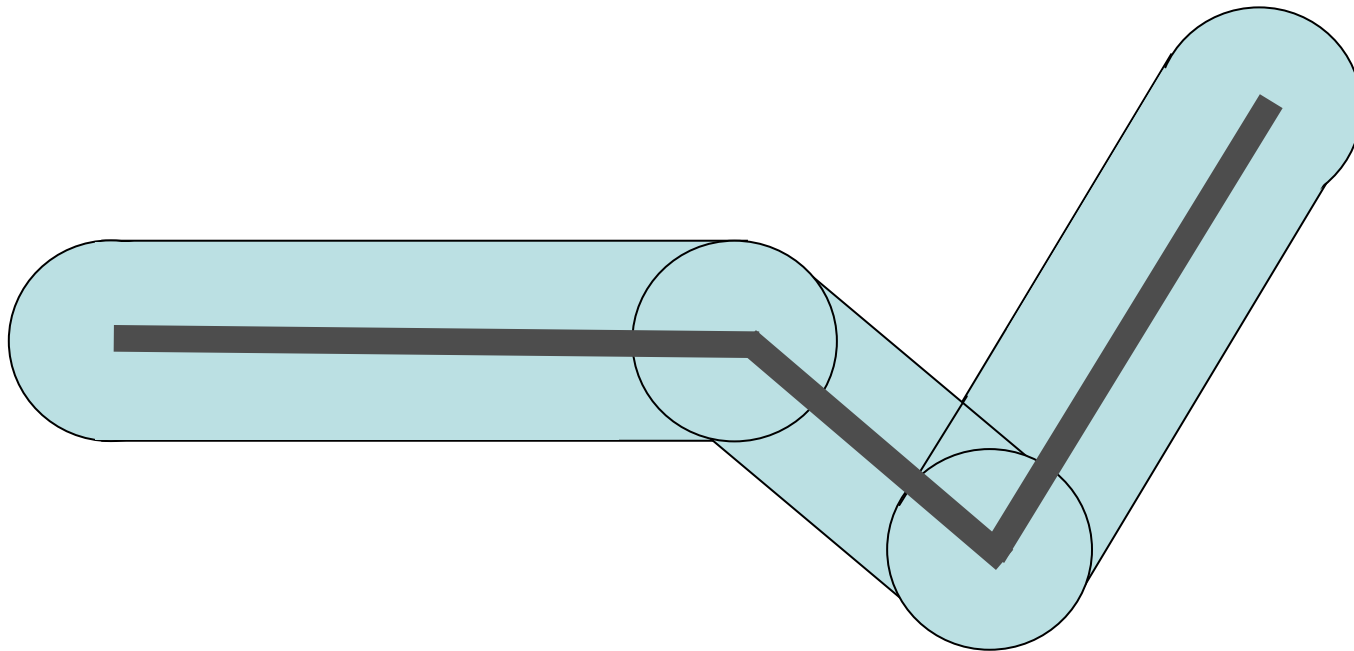
God Object Algorithm



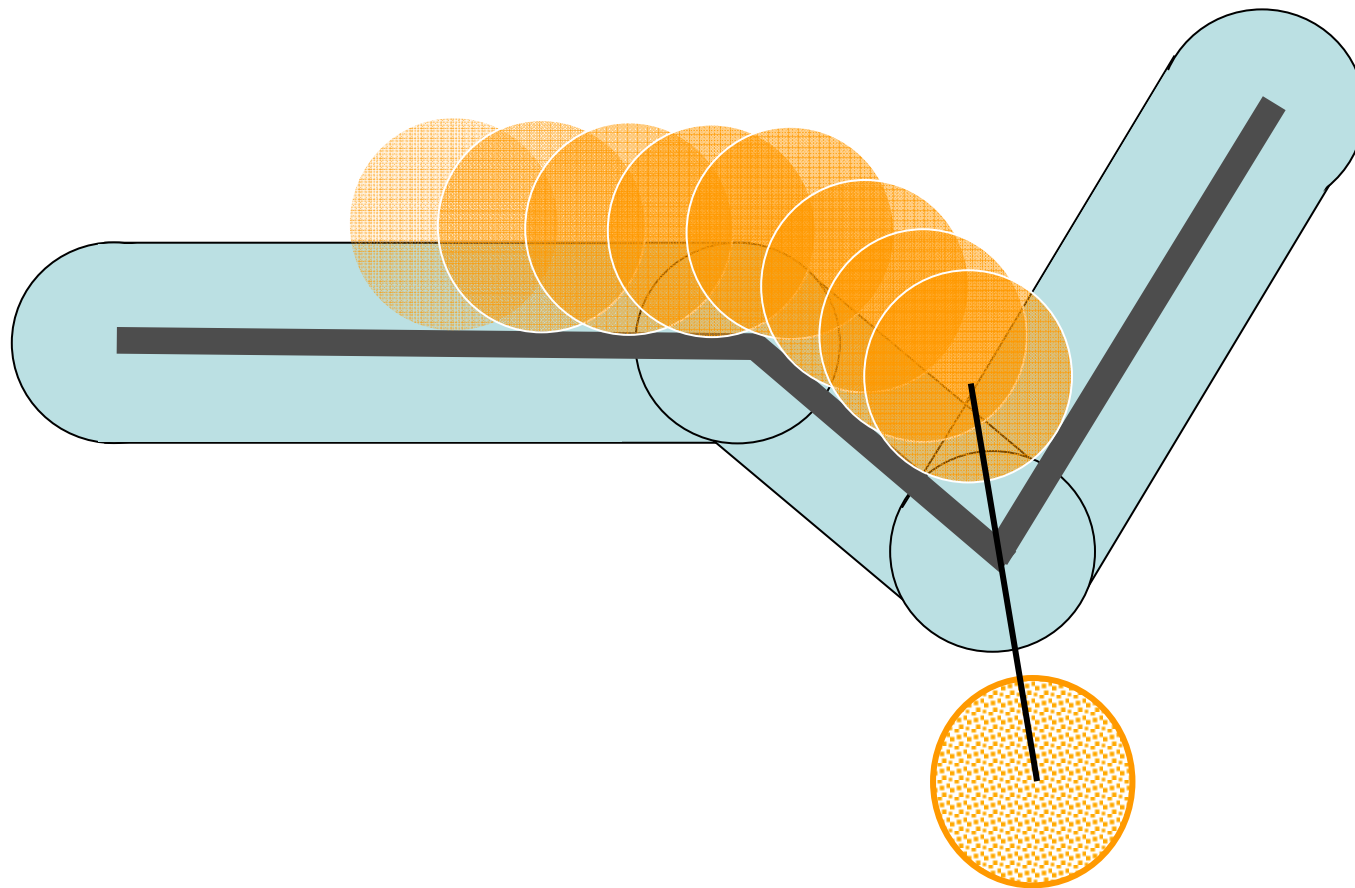
God Object Algorithm



God Object Algorithm

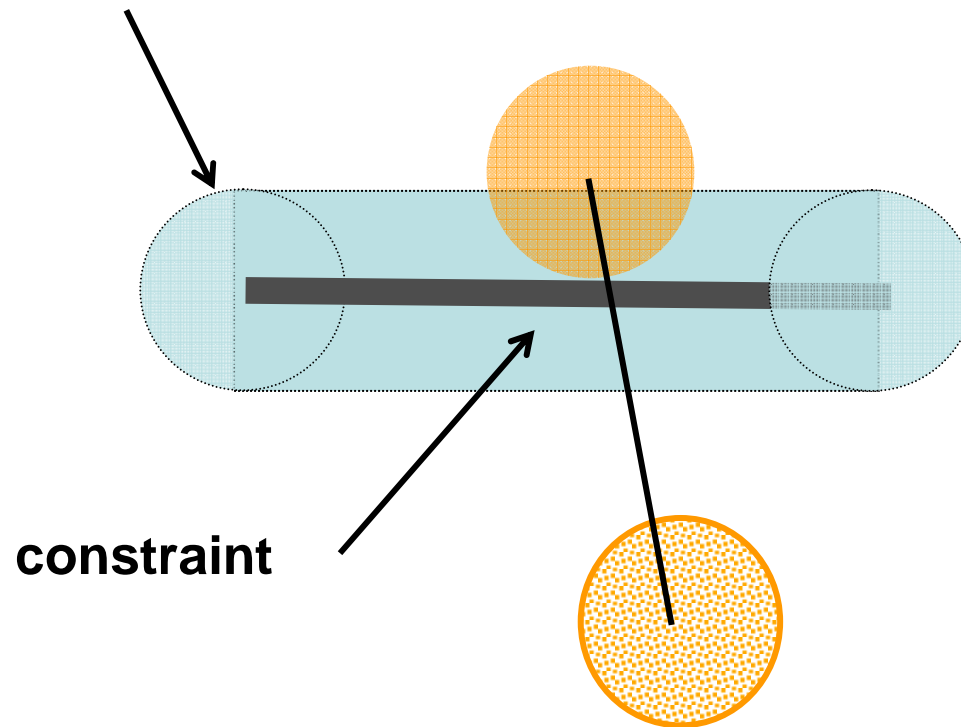


God Object Algorithm



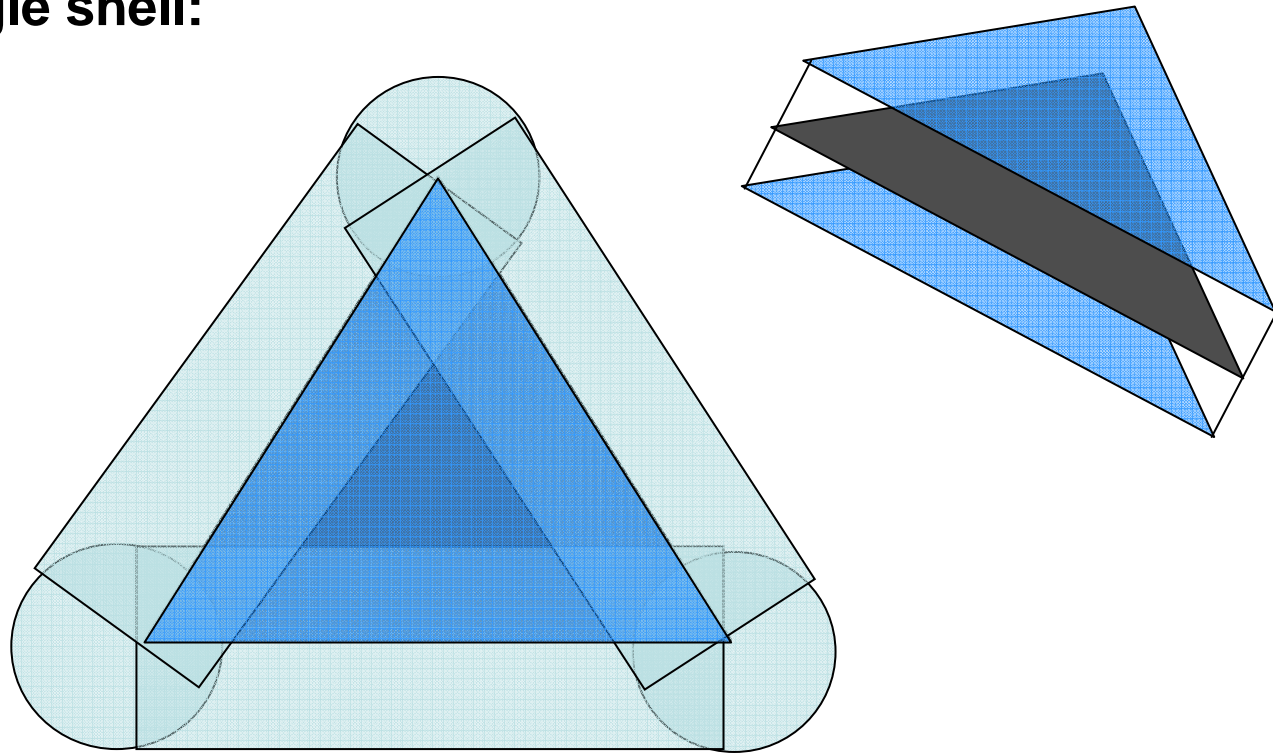
God Object Algorithm

shell: 1 box + 2 circles

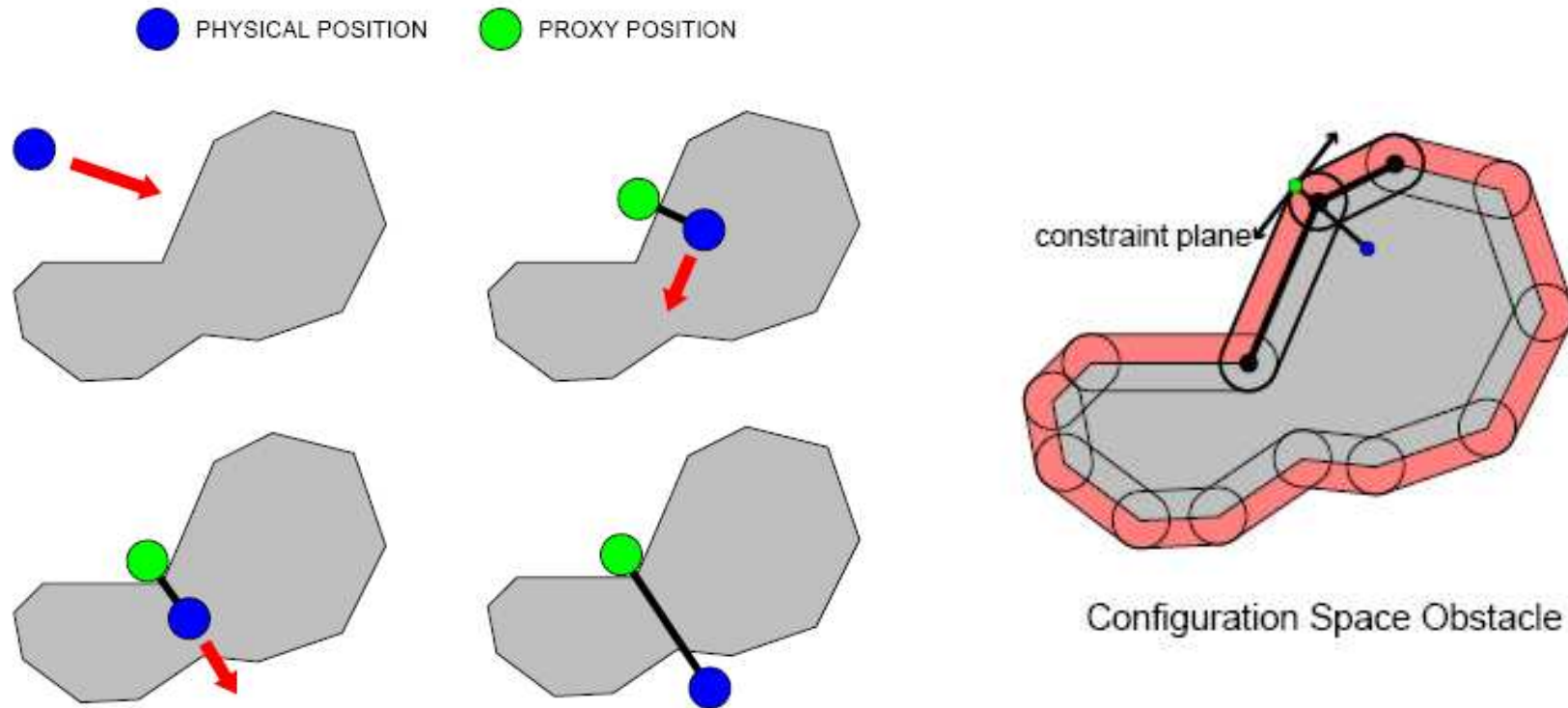


God Object Algorithm

triangle shell:



Finger – Proxy Algorithm



The Haptic Display of Complex Graphical Environments

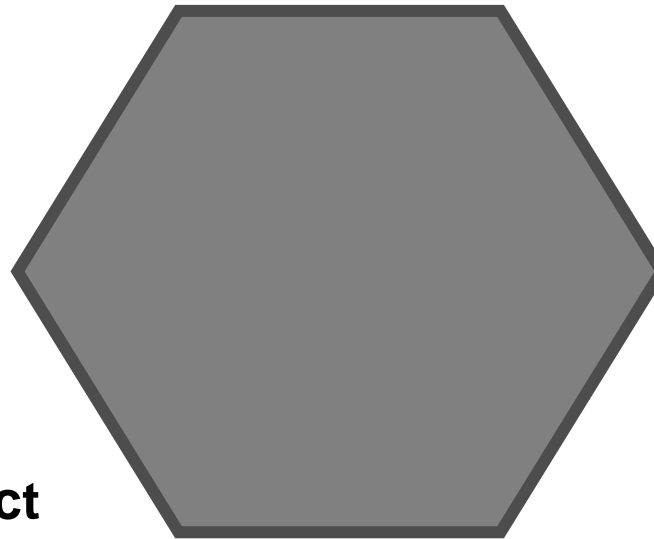
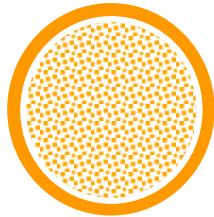
D. Ruspini, K. Kolarov and O. Khatib

CS277 - Experimental Haptics

CHAI3D

avatar / haptic device

time: t

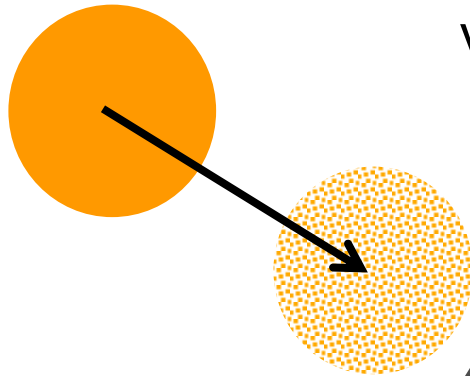


object

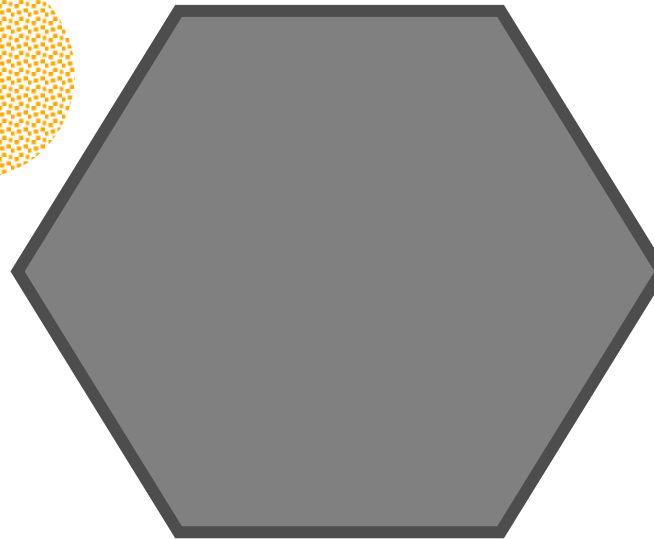
CHAI3D

avatar
time: t

check for any collision
between the segment and the
VR environment



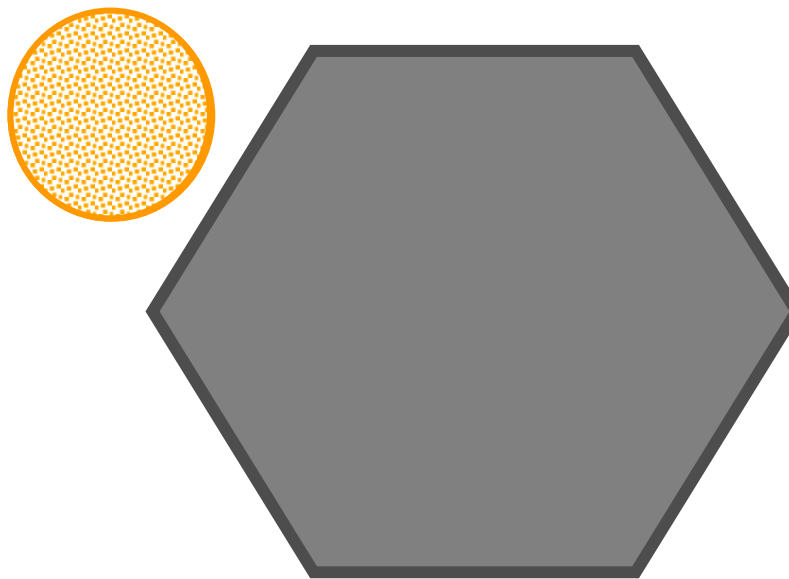
haptic device
time: $t + dt$



CHAI3D

avatar / haptic device

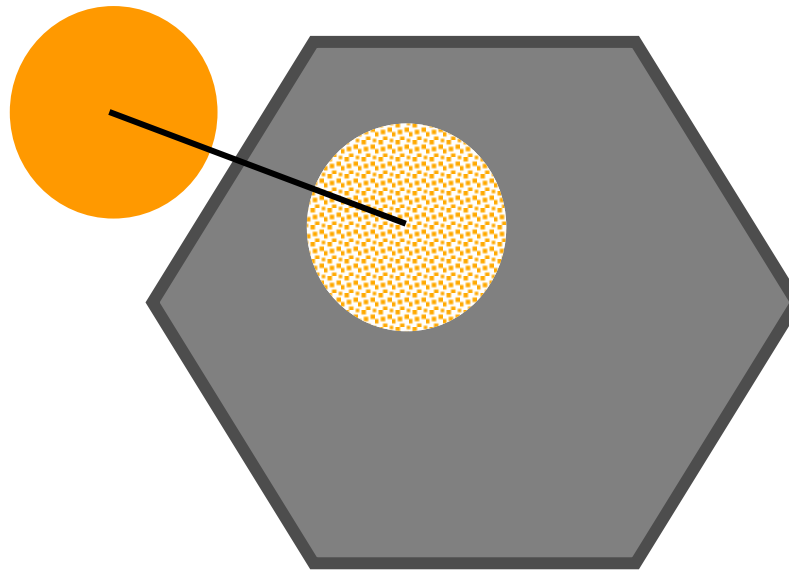
time: $t + dt$



CHAI3D

avatar
time: t

haptic device
time: $t + dt$

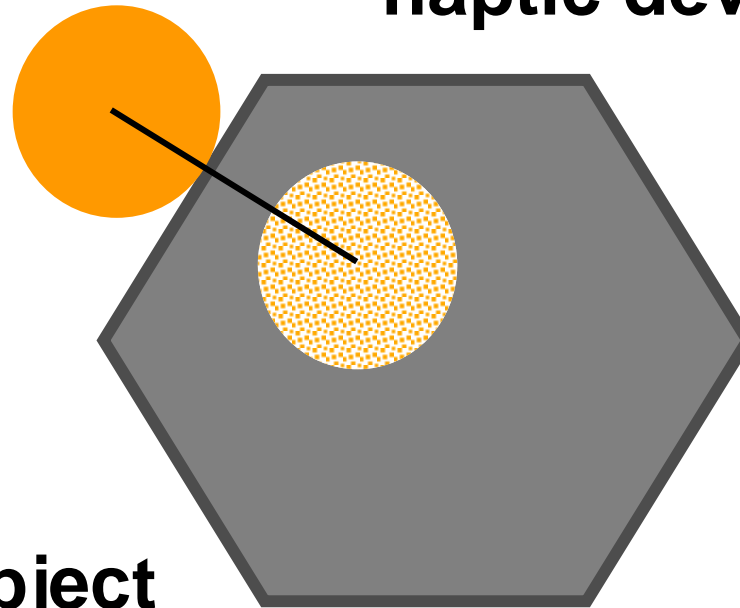


CHAI3D

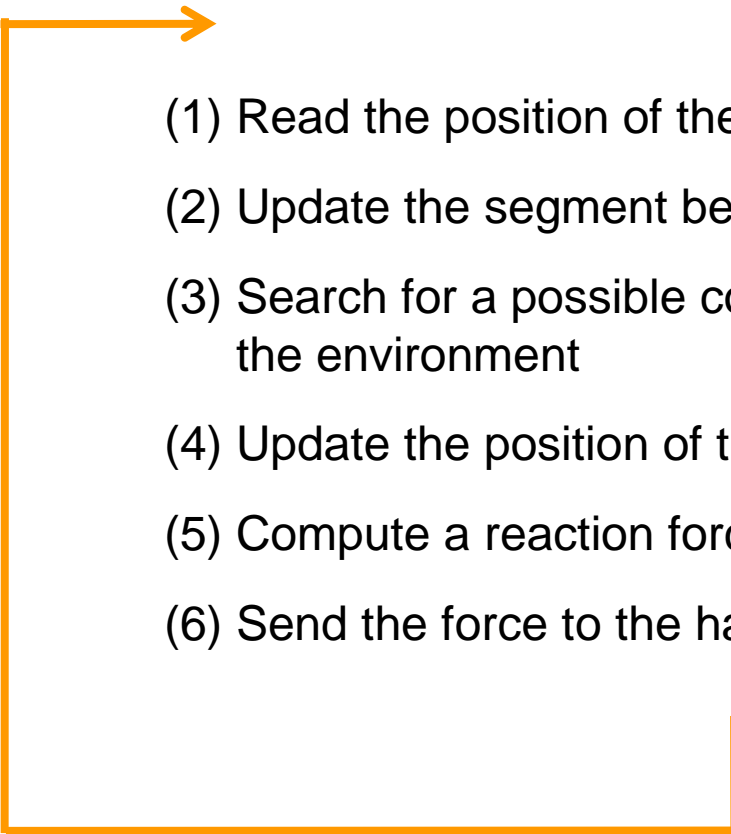
avatar

haptic device

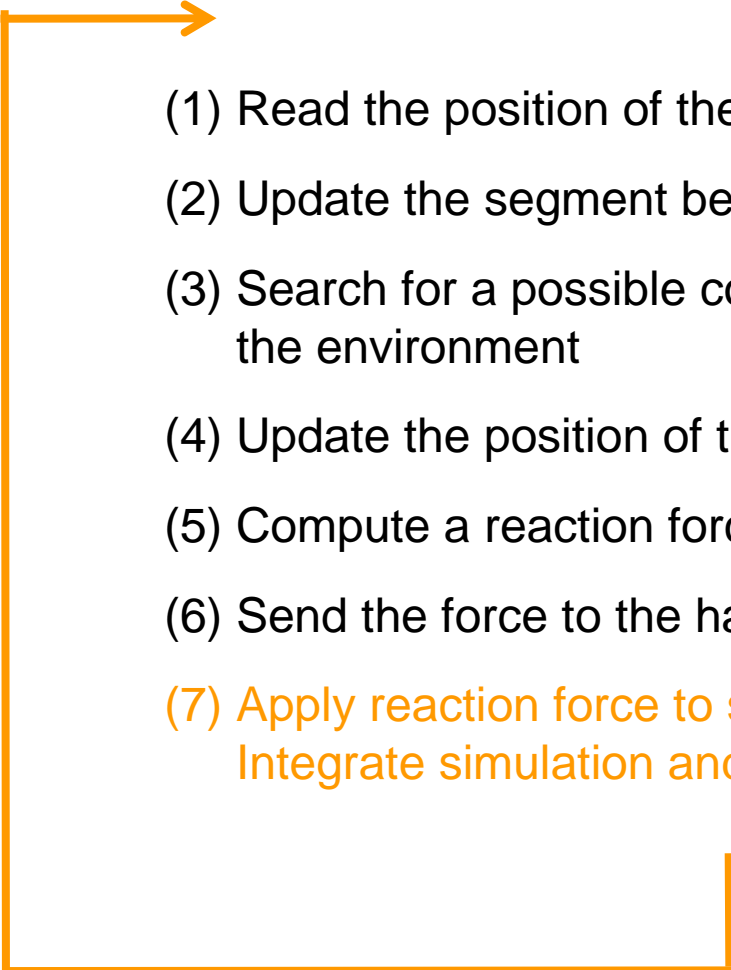
object



CHAI3D

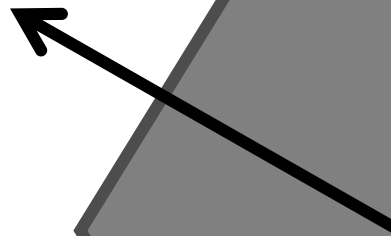
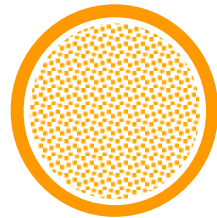
- 
- (1) Read the position of the haptic device
 - (2) Update the segment between the avatar and the device
 - (3) Search for a possible collision between the segment and the environment
 - (4) Update the position of the avatar
 - (5) Compute a reaction force
 - (6) Send the force to the haptic device

CHAI3D

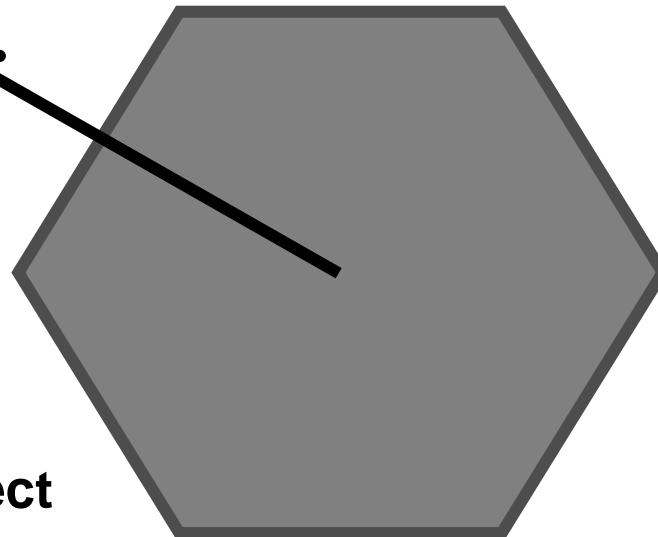
- 
- (1) Read the position of the haptic device
 - (2) Update the segment between the avatar and the device
 - (3) Search for a possible collision between the segment and the environment
 - (4) Update the position of the avatar
 - (5) Compute a reaction force
 - (6) Send the force to the haptic device
 - (7) Apply reaction force to simulated dynamic environment
Integrate simulation and compute new position of objects

CHAI3D

avatar / haptic device

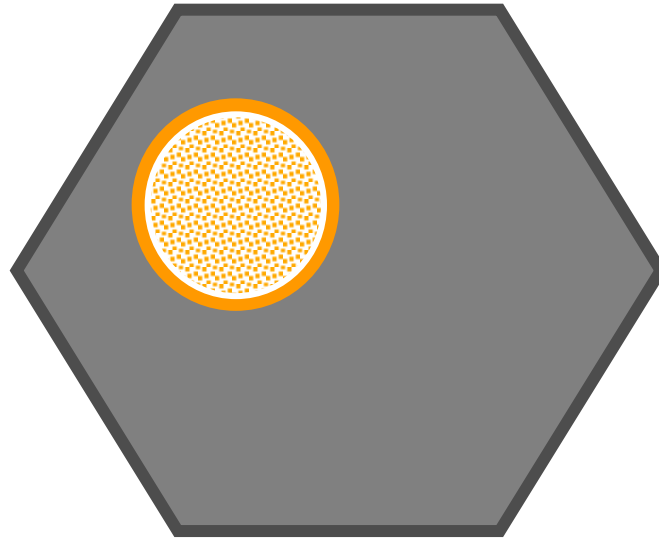


object



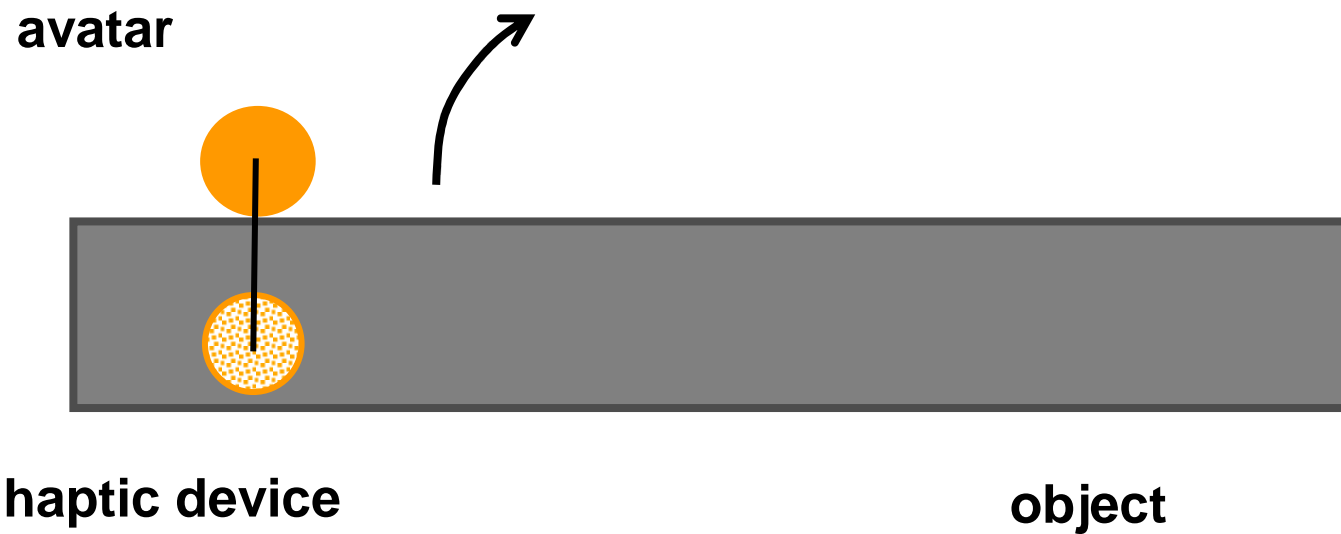
CHAI3D

avatar / haptic device

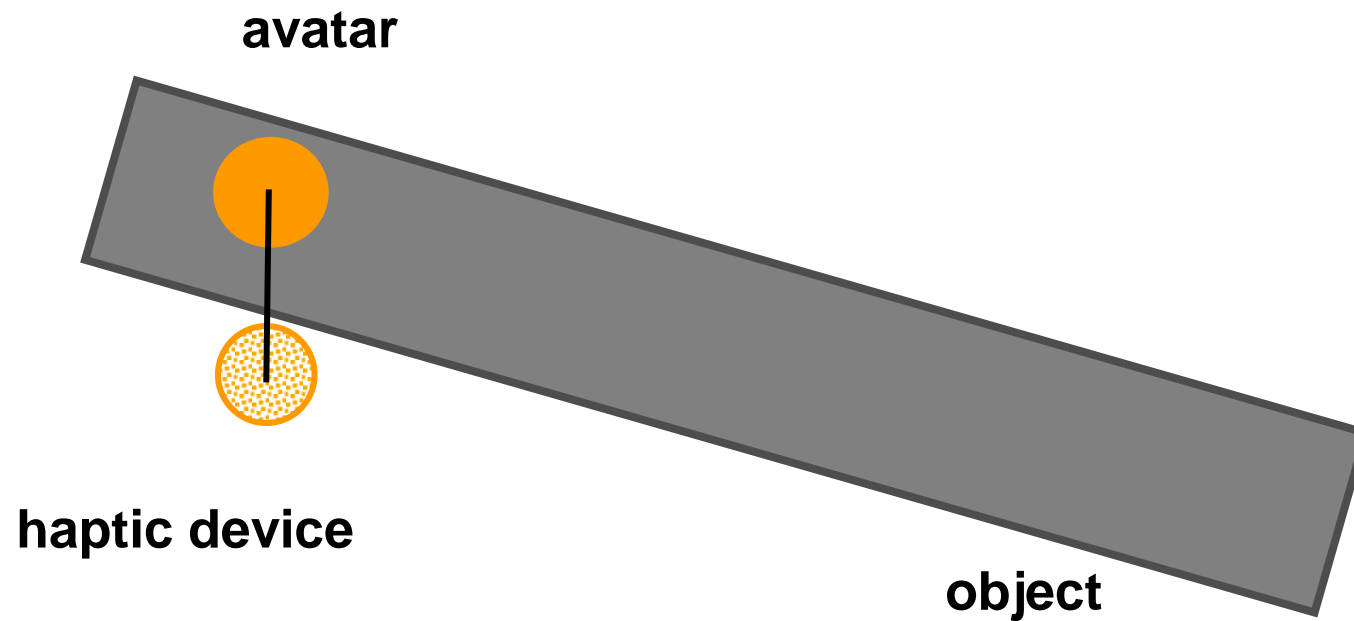


object

CHAI3D

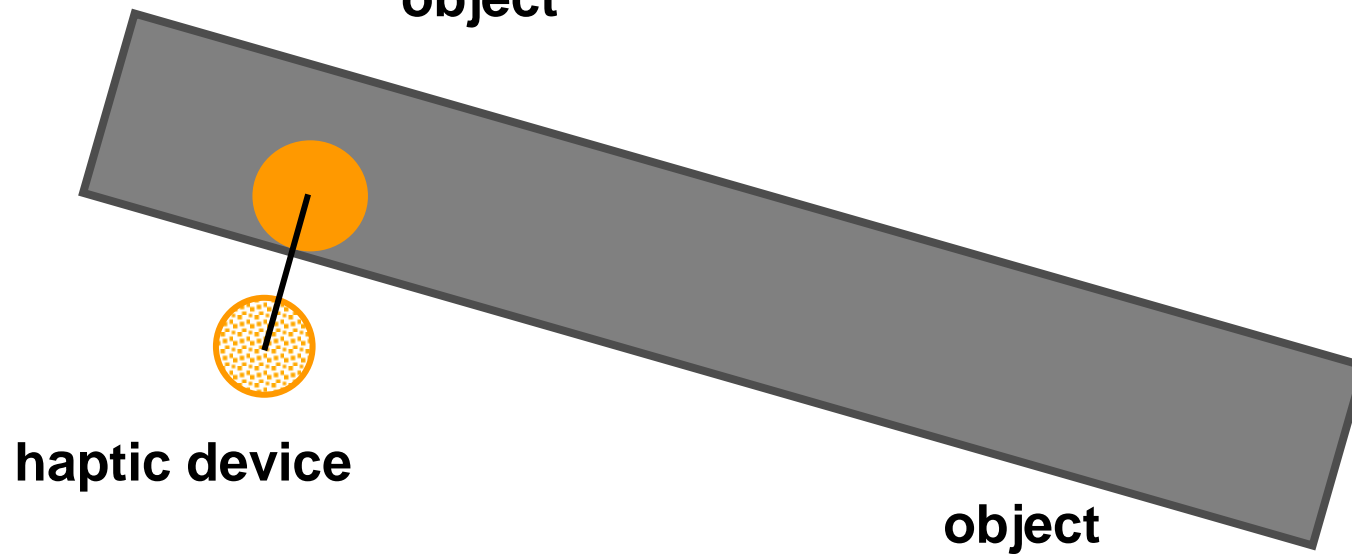


CHAI3D



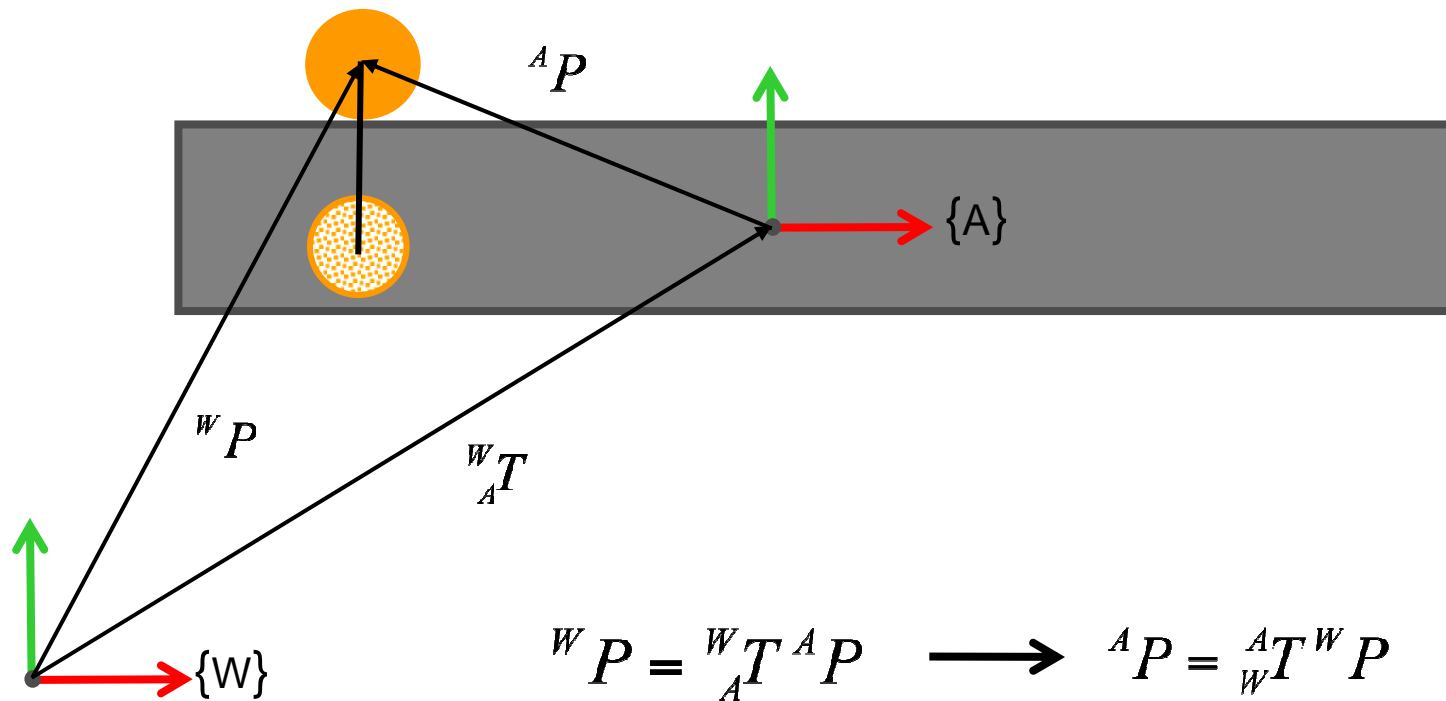
CHAI3D

avatar is now located inside the object

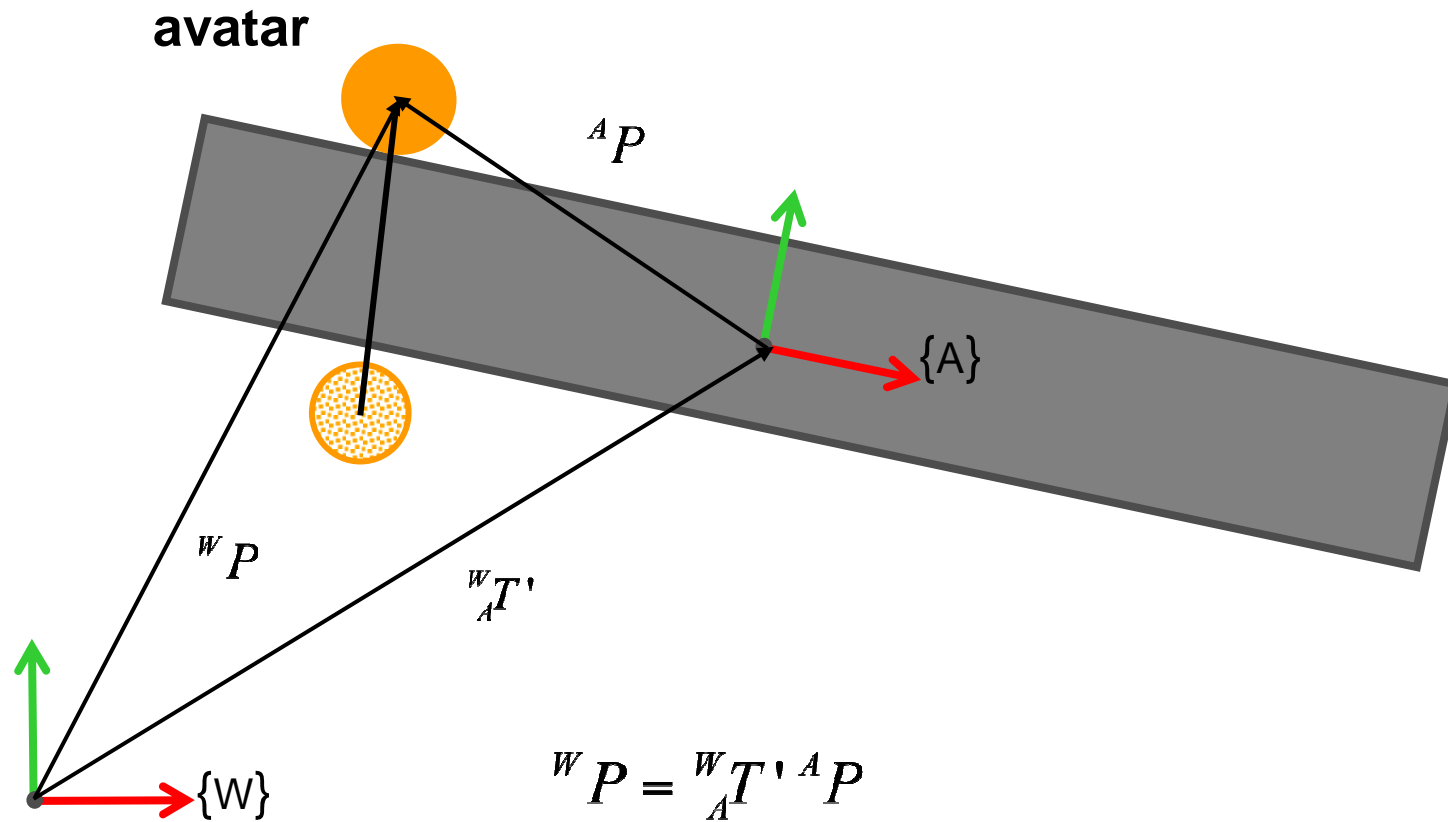


CHAI3D

avatar



CHAI3D



Outline

- Limitations when using Potential Fields
- God object algorithm
- Finger proxy algorithm
- **Implicit surfaces**
- Demonstrations

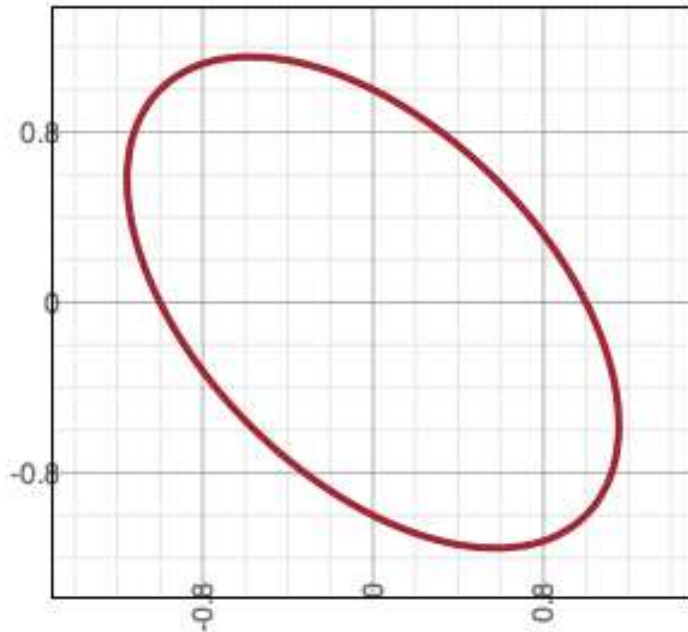
Rendering Implicit Surfaces

- A surface defined by an implicit equation:
 - $S(x, y, z) = 0$
- Can be rendered using the same proxy-based algorithm.

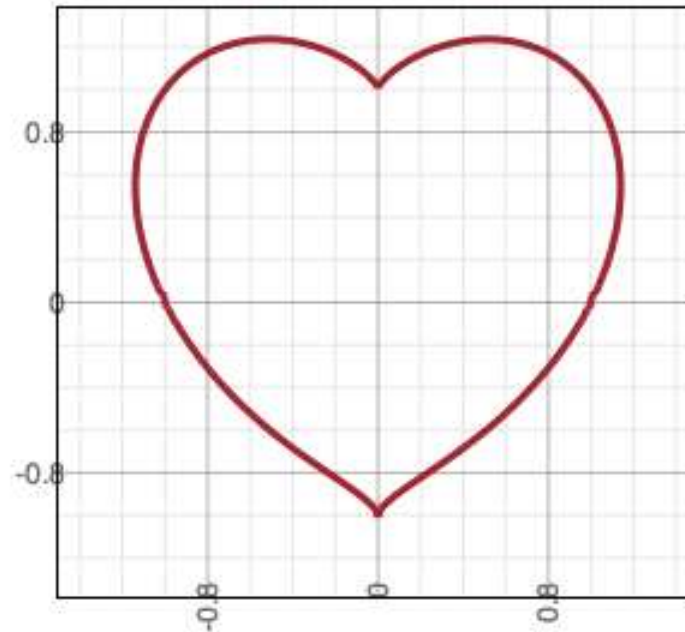


Review: Implicit Surfaces

- “Graph” the function, as you did in high school...



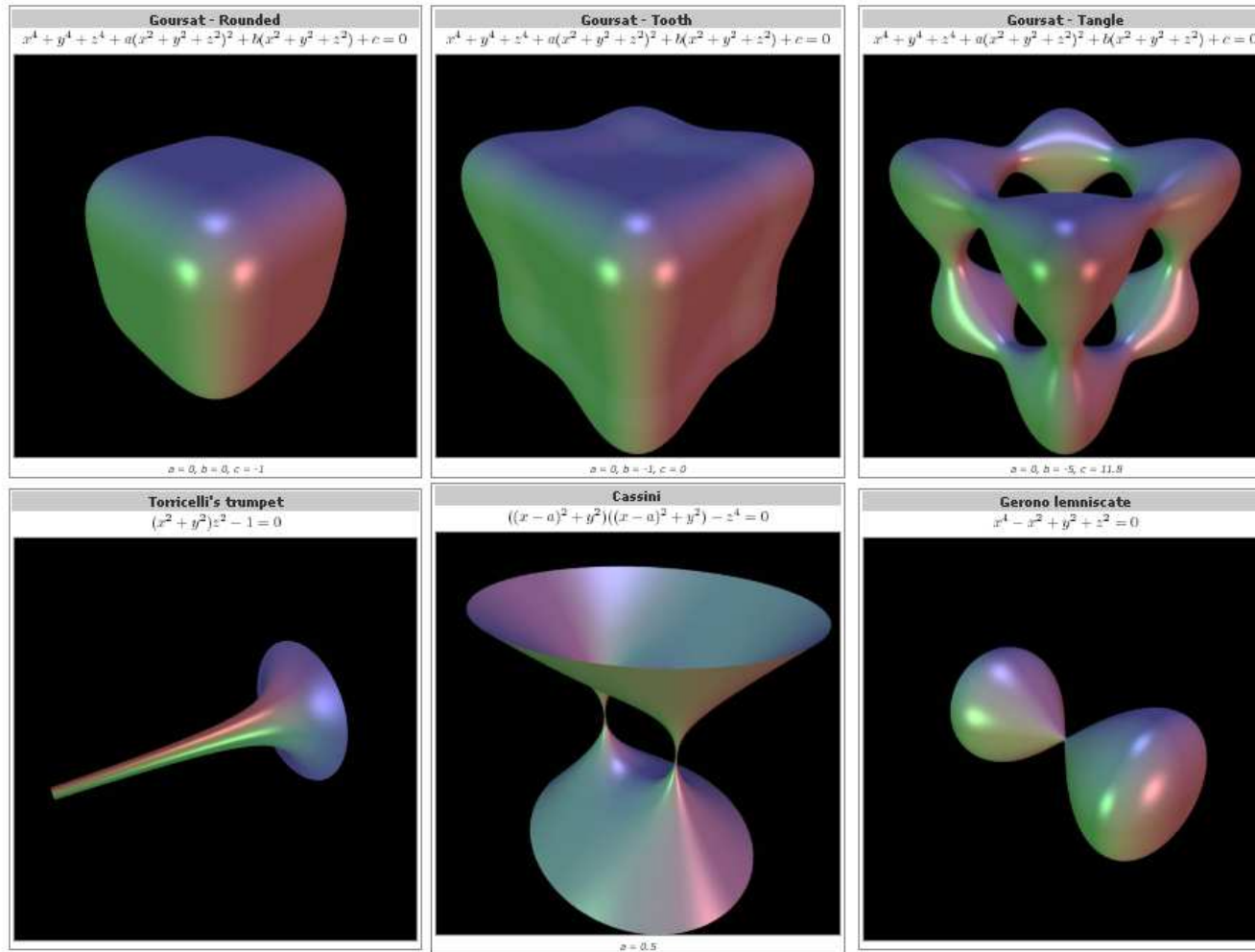
$$x^2 + xy + y^2 = 0$$



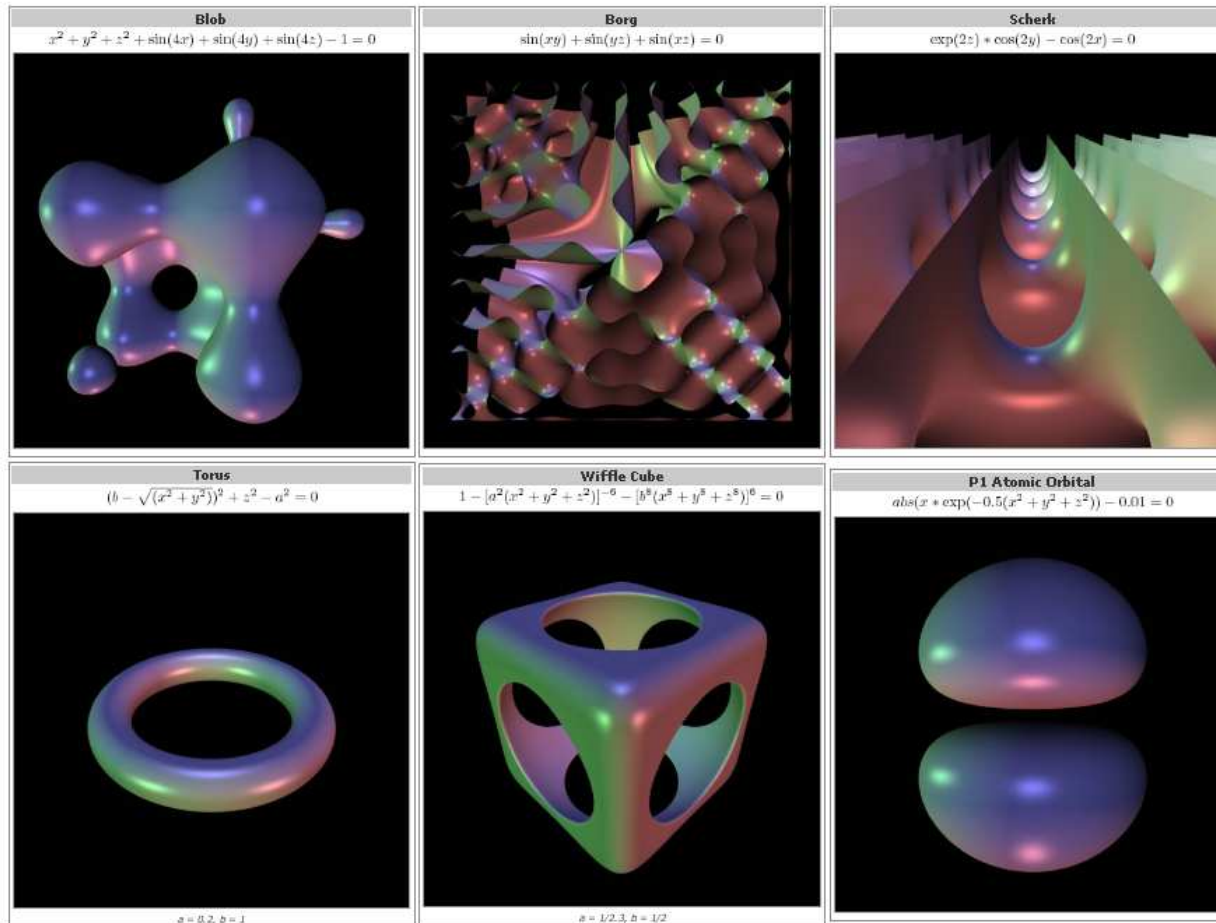
$$(x^2 + y^2 - 1)^3 - x^2y^3 = 0$$

From XRT Renderer: <http://xrt.wikidot.com/gallery:implicit>

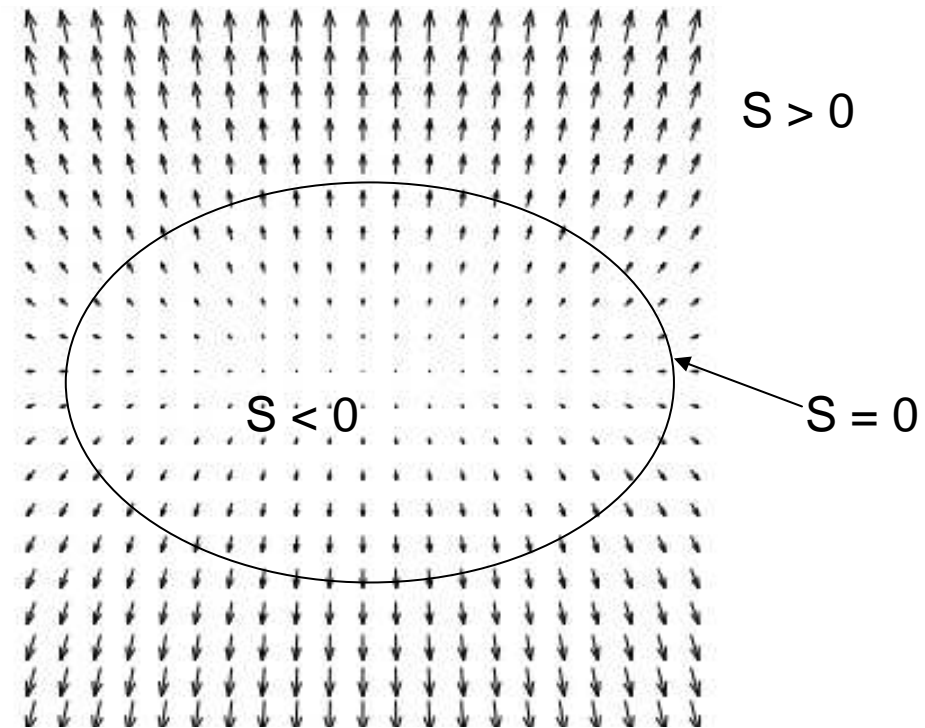
Quartic Surfaces

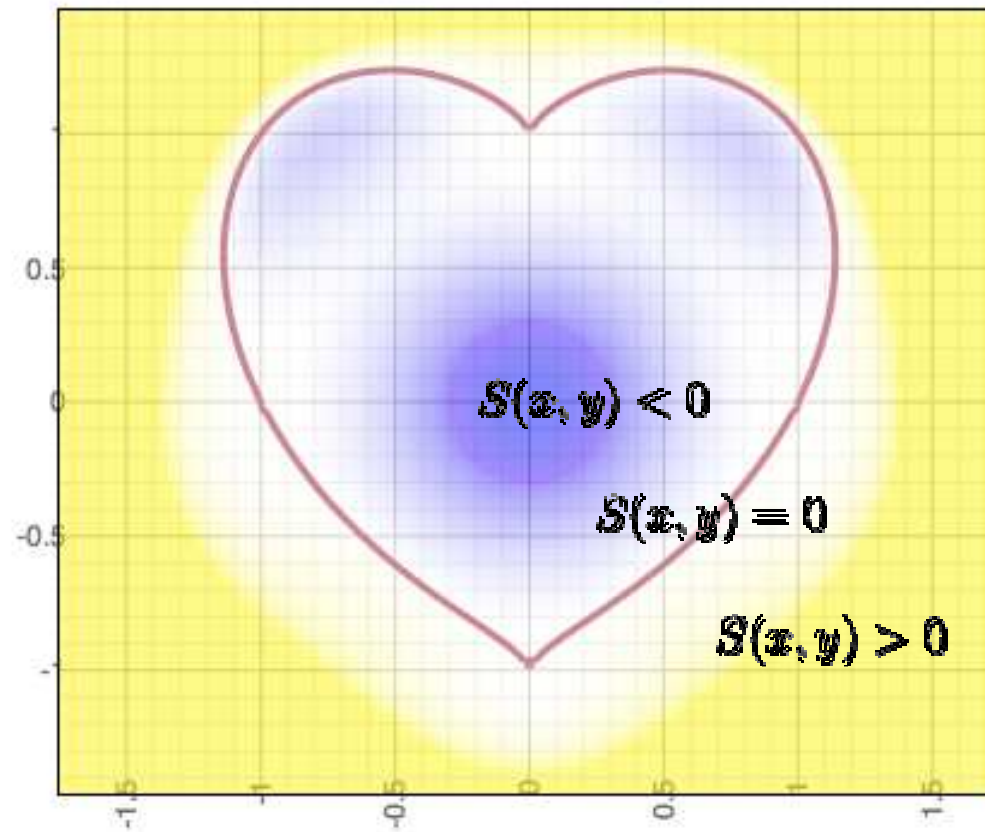


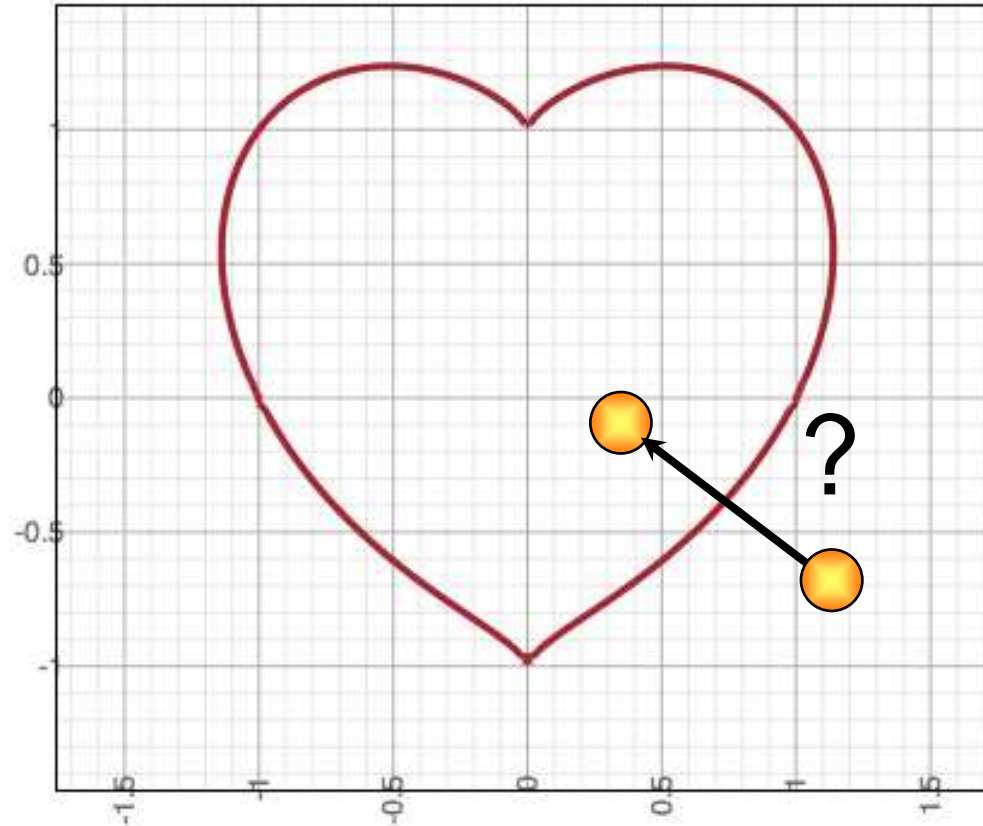
Non Algebraic Surfaces



Implicit surface and gradient map



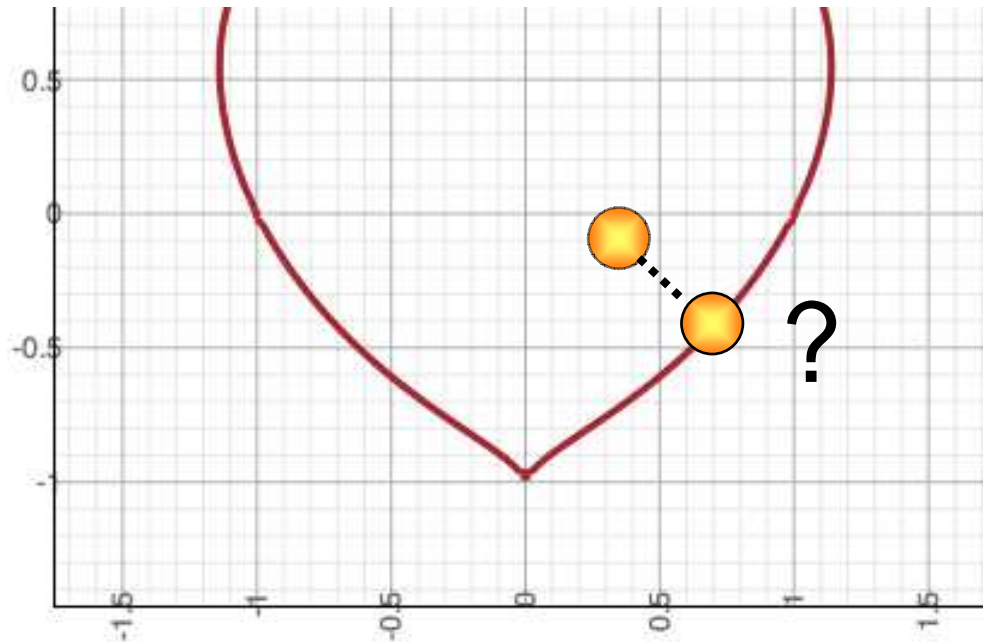




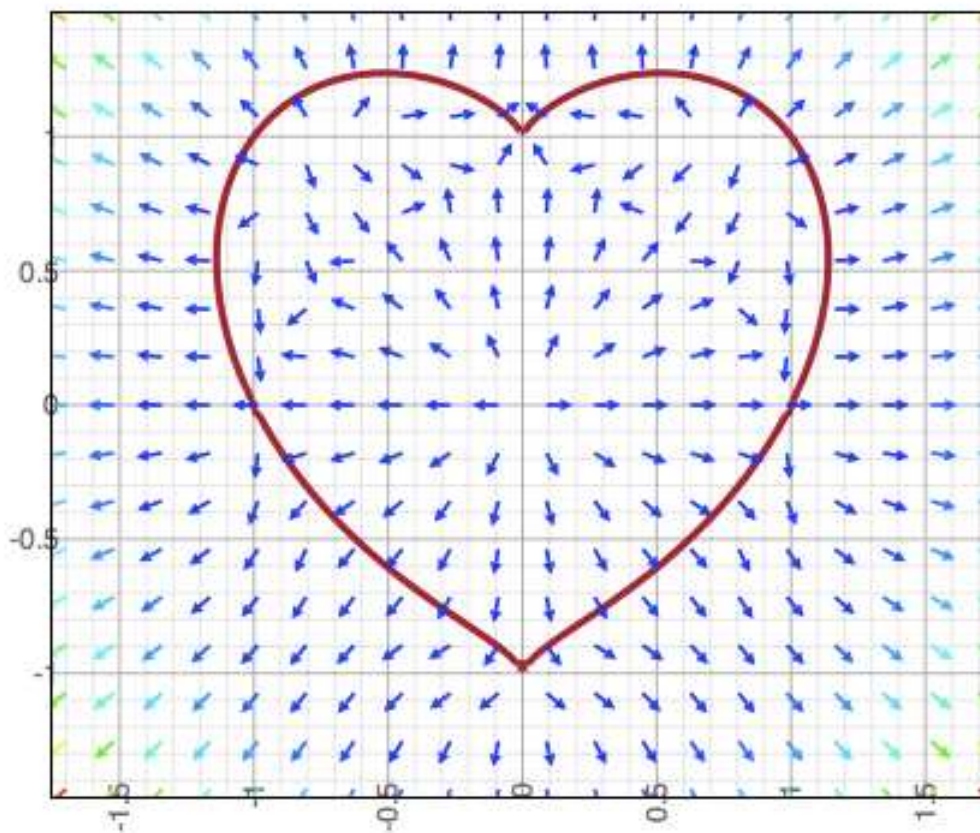
How do we detect the first contact with the object?

Step 2: Finding a Surface Point

- Once contact is made, we need to keep track of a point on the surface of the object.
- First, how do we find this point?

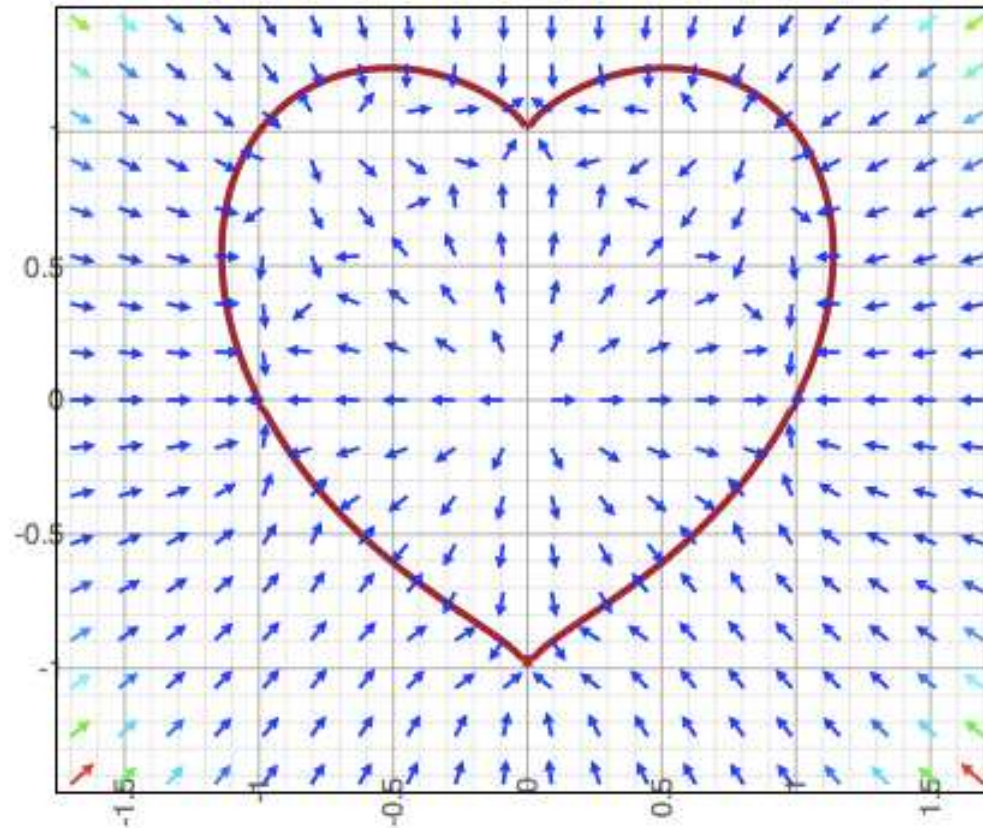


The Gradient



$$\nabla S(x, y) = \begin{pmatrix} \frac{\partial S}{\partial x} \\ \frac{\partial S}{\partial y} \end{pmatrix}$$

Direction to Surface

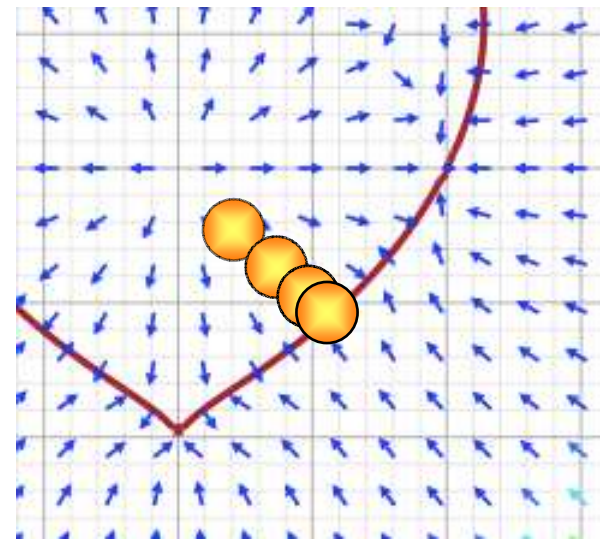


$$-S(x, y) \nabla S(x, y)$$

The “Seeding” Algorithm

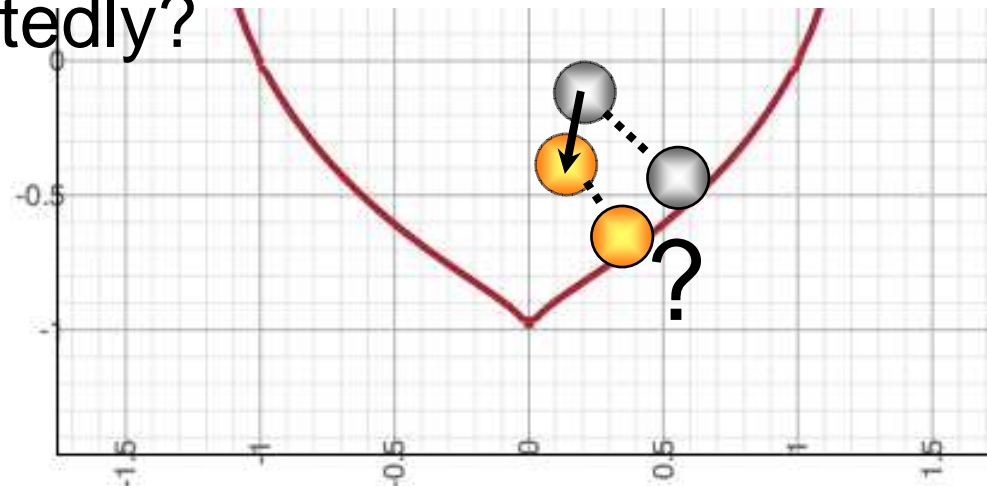
- Given a seed point, find the nearest point on the surface (within a certain tolerance)
- Exploit the condition that the seed is known to start close to the surface

```
p ← pseed  
do  
   $\delta \mathbf{p} \leftarrow -\frac{S(\mathbf{p})\nabla S(\mathbf{p})}{\nabla S(\mathbf{p}) \cdot \nabla S(\mathbf{p})}$   
  p ← p +  $\delta \mathbf{p}$   
until ( $\|\delta \mathbf{p}\| < \epsilon$ )
```



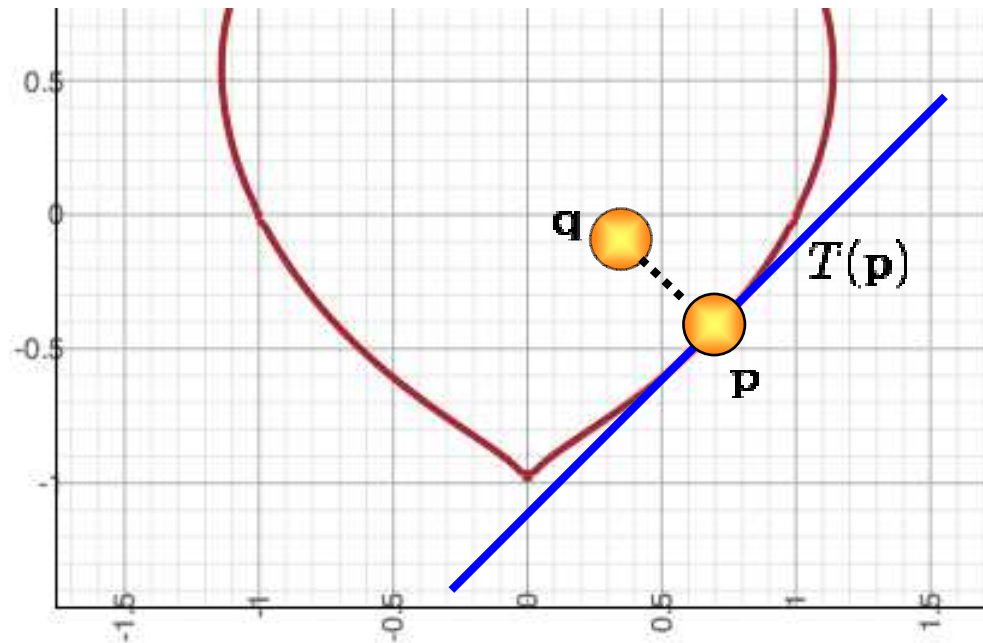
Step 3: Tracking the Surface Point

- As the device is moved around inside the object, we need to track of the movement of our point on the surface, subject to the surface constraints.
- What are these constraints?
- Can we use the seeding algorithm repeatedly?

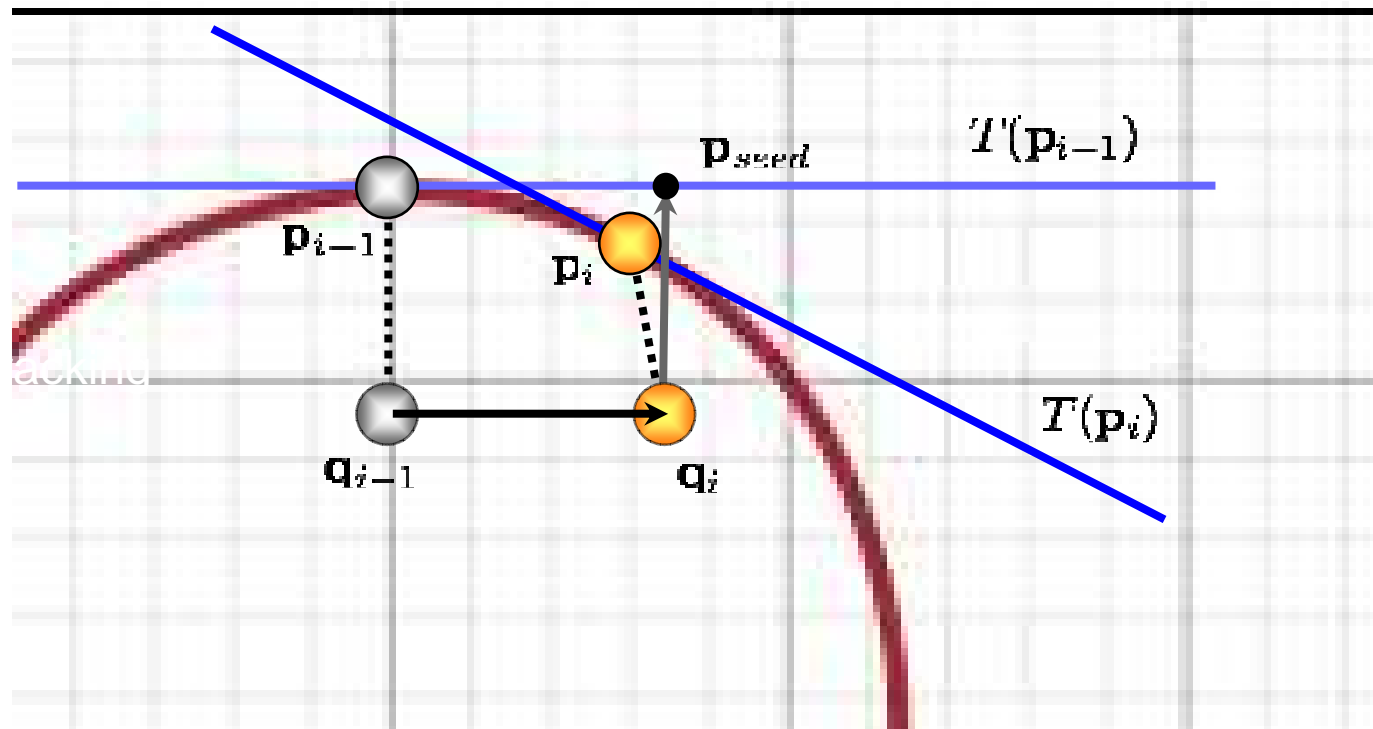


Constrained by a Plane

- We have a point on the surface...
- We have the surface normal (the gradient)...
- The answer is to use a tangent plane!

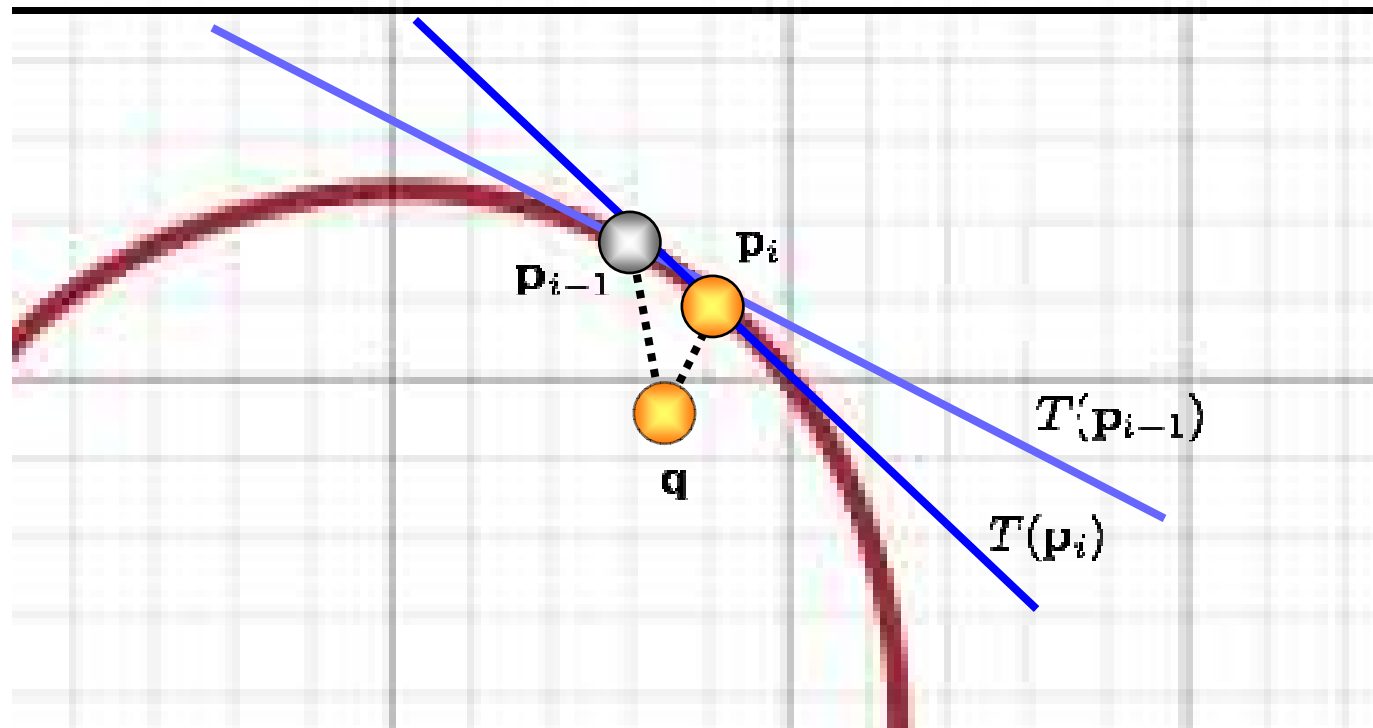


Surface Tracking



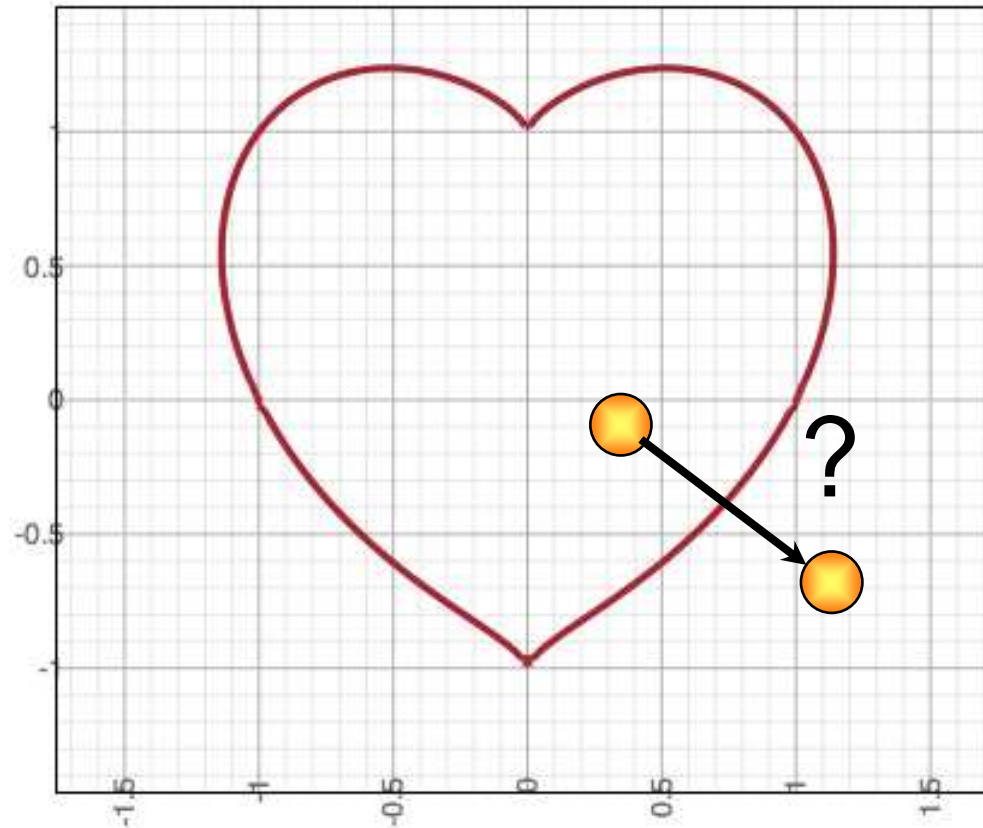
One time step: Is p the nearest surface point?

Surface Tracking



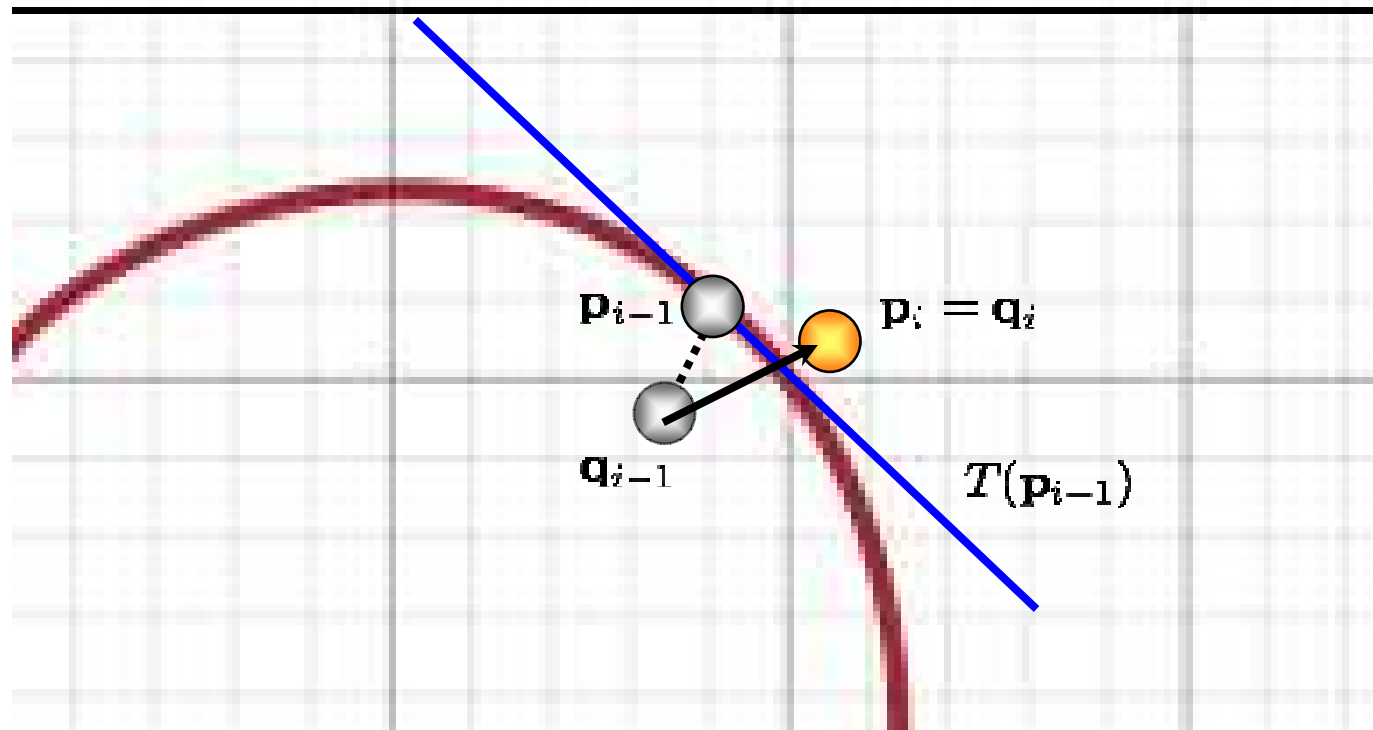
One more time step: A lot closer now!

Step 4: Breaking Contact



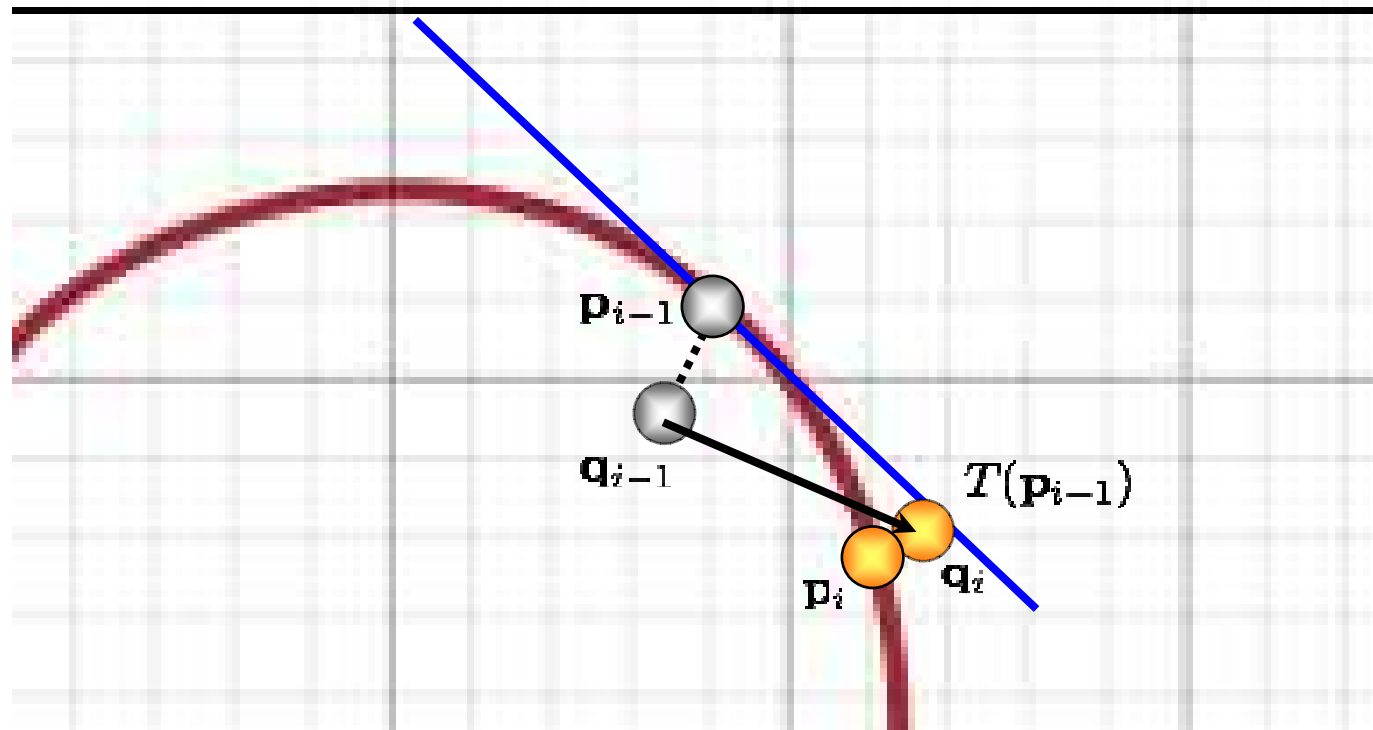
How do we know when to stop tracking?

Tangent plane to the rescue



Contact is broken when q moves to the outside of the constraining plane (same direction as normal).

Incorrect Break?



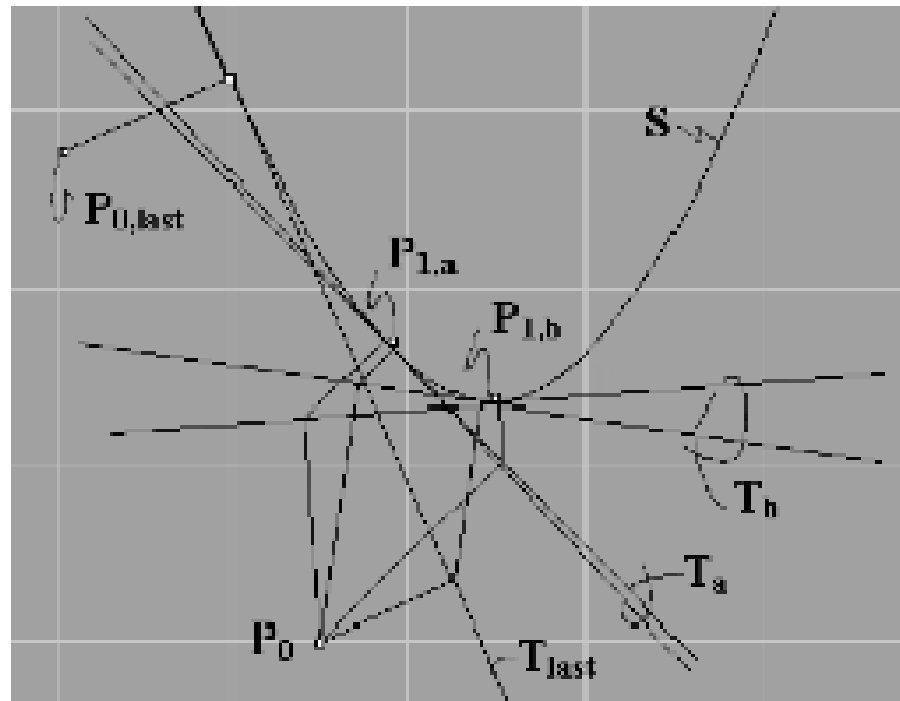
What happens here?

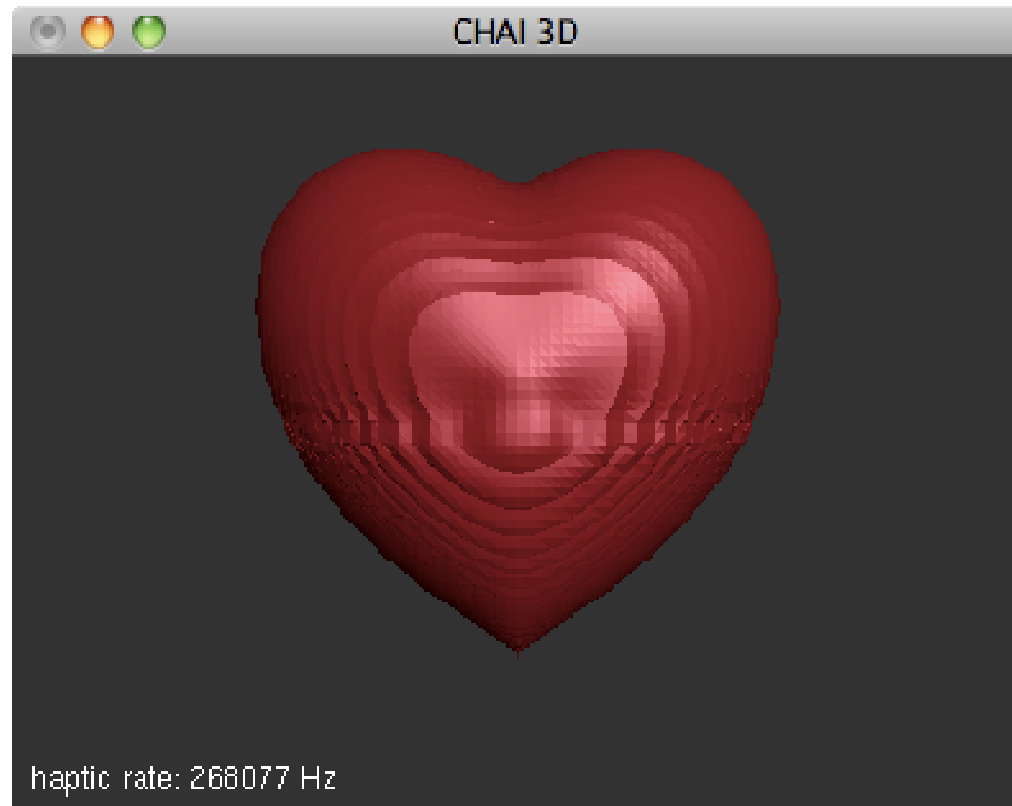
Summary

- The full implicit surface rendering algorithm:
 - Detect initial contact when $S(p) < 0$
 - Find surface point using initial point as seed
 - Update the surface point as the device moves by using the tangent plane as a constraint
 - Contact breaks when device is moved outside the constraining plane
- Repeat from start...

Potential Limitations

- Can this algorithm handle thin objects?
- A limit cycle?!





Additional References:

XRT Renderer

<http://xrt.wikidot.com/gallery:implicit>

A. Ricci, "A constructive geometry for computer graphics," The Computer Journal (1973) 16(2): 157-160 doi:10.1093/comjnl/16.2.157

<http://comjnl.oxfordjournals.org/content/16/2/157.full.pdf+html>

Zilles, Craig, "Haptic Rendering with the Tool-Handle Haptic Interface," MS and BS Thesis, Mechanical Engineering Dept, Massachusetts Institute of Technology, May 12, 1995.

<http://ai.stanford.edu/~jks/theses/zilles/1995-zilles-thesis.pdf>