Exploiting Perception in High-Fidelity Virtual Environments

Introduction to Haptic Rendering

Ming C. Lin

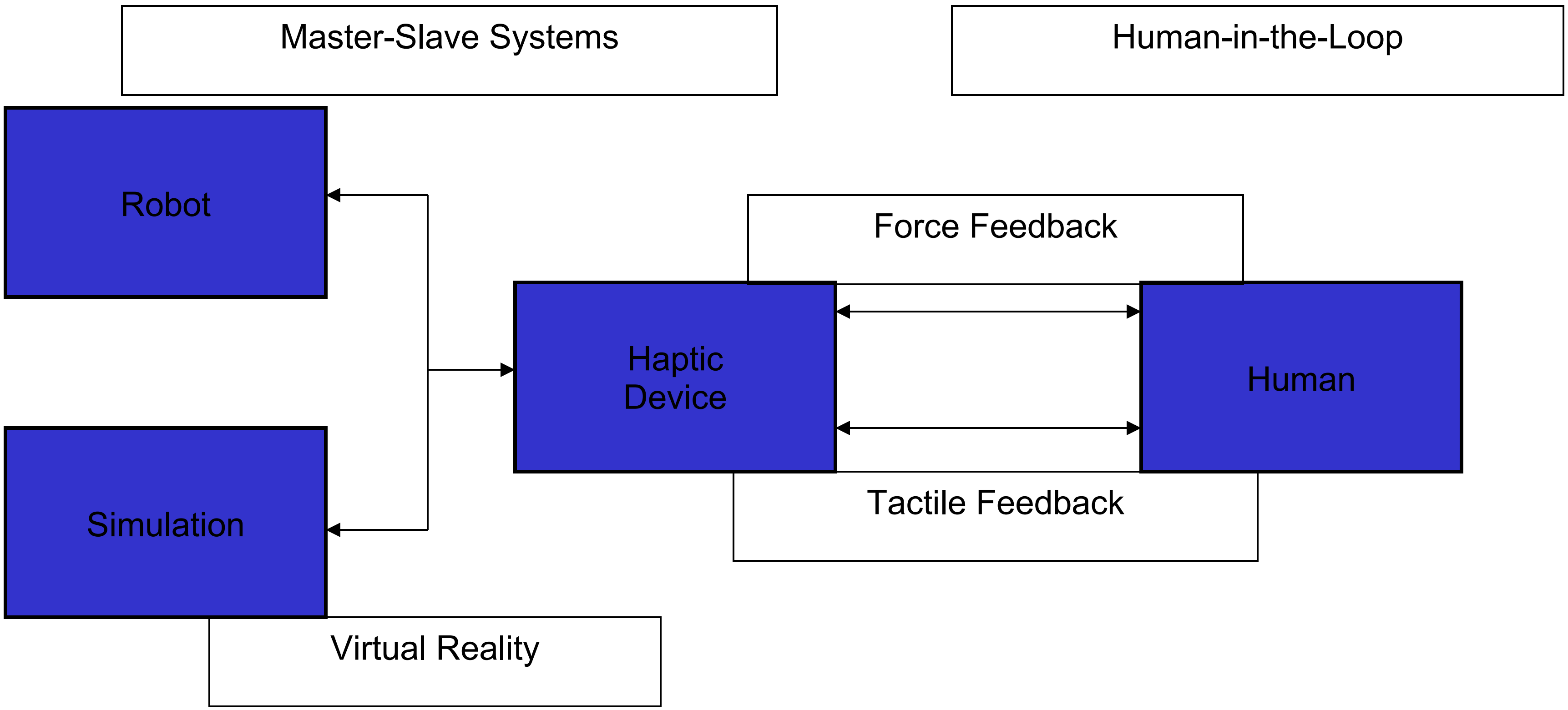
Department of Computer Science

University of Maryland at College Park

http://www.cs.umd.edu/~lin

lin@umd.edu

What Is Haptic Rendering?



Inter-disciplinary Research

Computer Science — Haptic Rendering

↓

Computation of the forces output by the device

Electrical Engineering — Control and actuators

Mechanical Engineering — Mechanical design

Control of Haptic Devices

- **Impedance Devices**

6-DOF Phantom

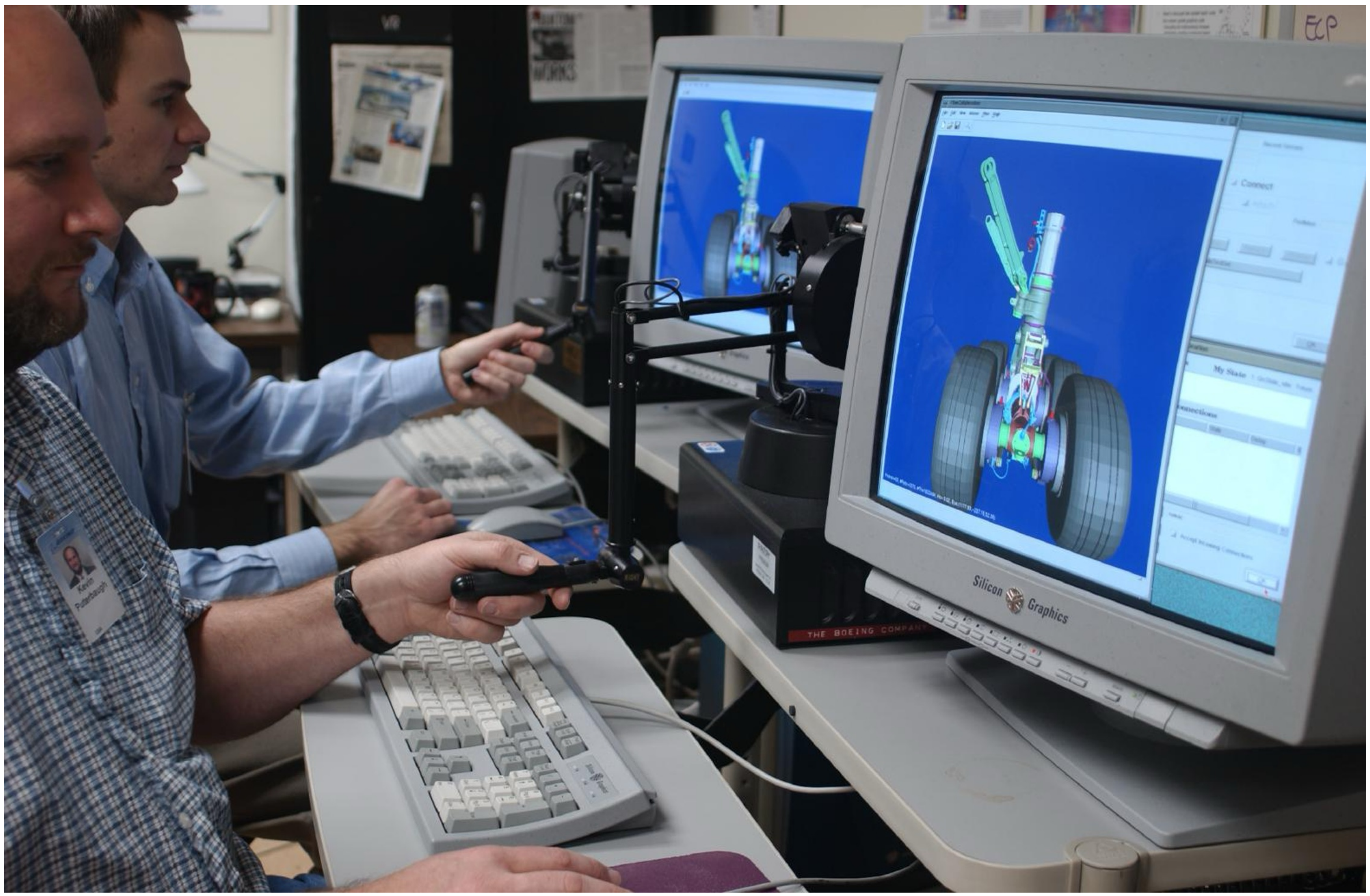
- Admittance devices

COBOTs

Engine Close-Up

Boeing VPS System

Collaborative Haptic Design Review



Other Examples

- A Haptic Hybrid Controller for Virtual Prototyping of Vehicle Mechanisms (Ford, BMW, etc)

- 3-DOF Cobot for Engineering Design

  (Northwestern University and Ford Automobile)

Immersion Medical Simulators

- **Endoscopy simulator** **-** Bronchoscopy and upper and lower gastrointestinal procedures on a single platform
- **Endovascular simulator** - Percutaneous coronary and peripheral interventions and cardiac rhythm management
- **Hysteroscopy simulator** - Skills assessment and myomectomy
- **Laparoscopy simulator** - Skills, cholecystectomy, sterilization, ectopic pregnancy, and myomectomy suturing
- **Vascular access simulator** - Adult, geriatric, and pediatric IV; PICC; phlebotomy; and skills assessment

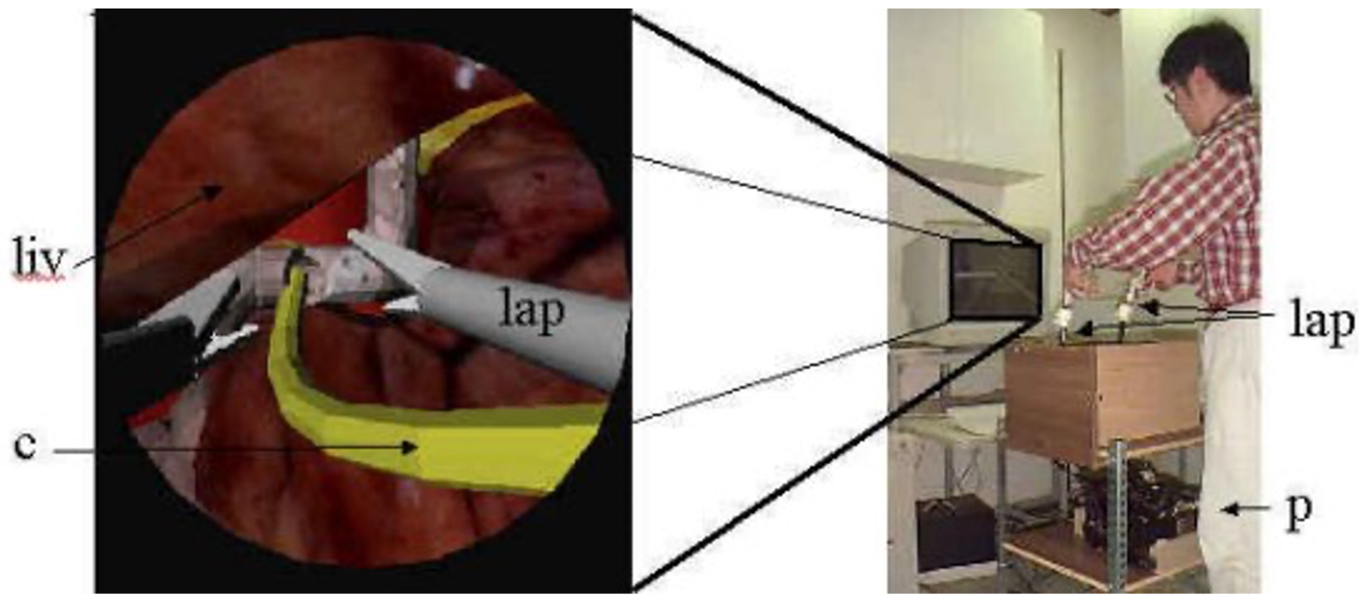
Virtual Endoscopic Surgery Training

VEST System *One* (VS*One*)
Technology

- 3 haptic (force-feedback) devices as mock-up endoscopic instruments
- 1 virtual endoscopic camera
- three new ***Basic Task Training (BTT)*** exercises - Find tubes/touch points/follow path

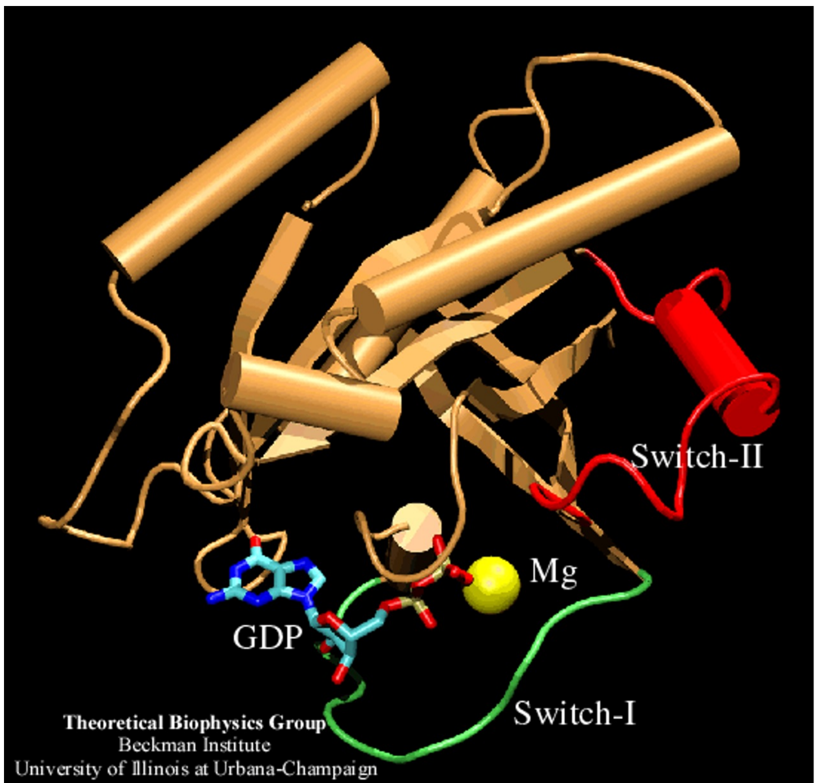
Laparoscopic Surgery

- MIT Touch Lab



Molecular Dynamics

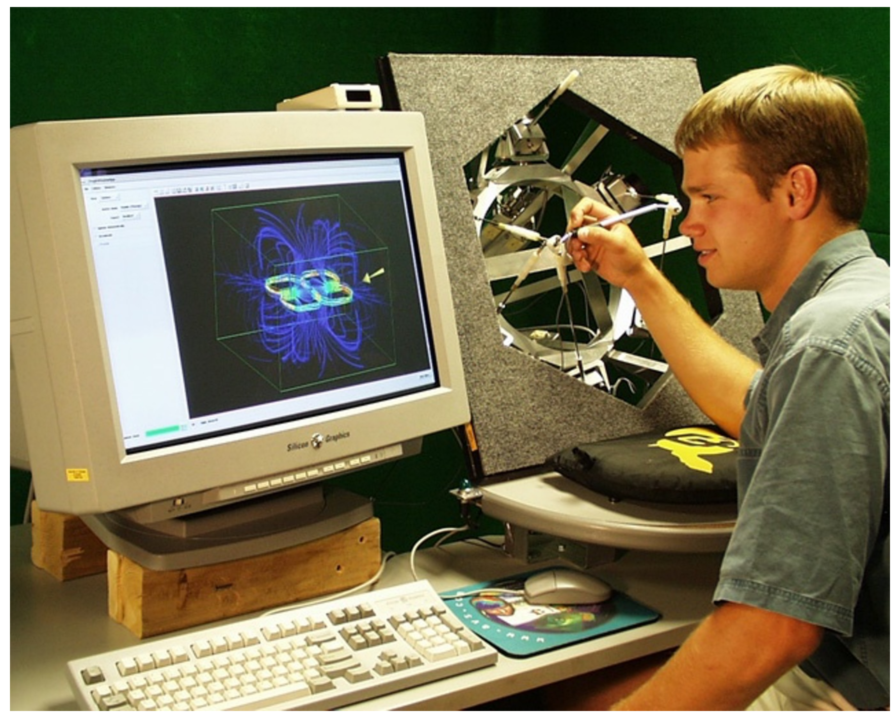
- VMD: Visual Molecular Dynamics



Humphrey, 1996

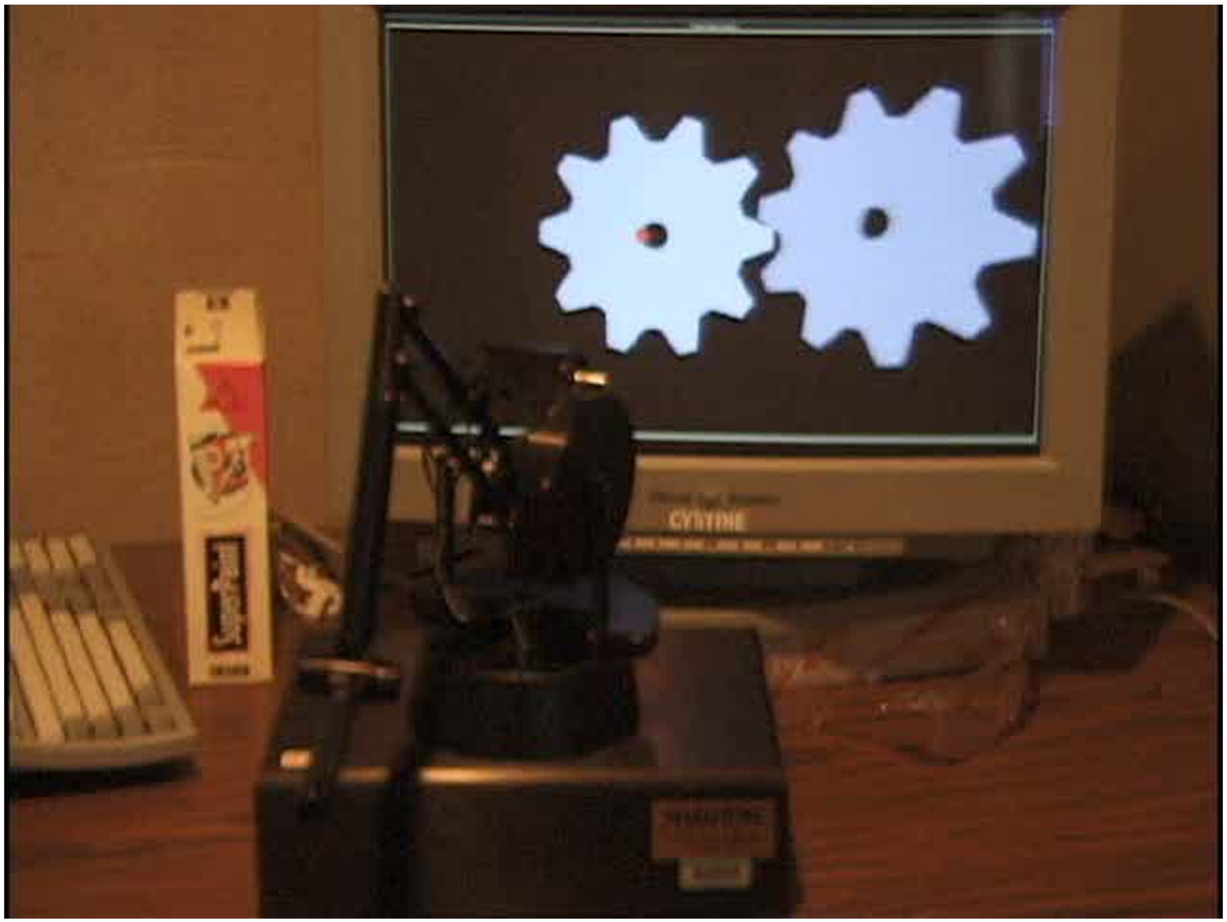
Haptic Vector Field

- Lawrence, Lee, Pau, Roman, Novoselov
  - University of Colorado at Boulder
- 5 D.O.F. in
- 5 D.O.F. out

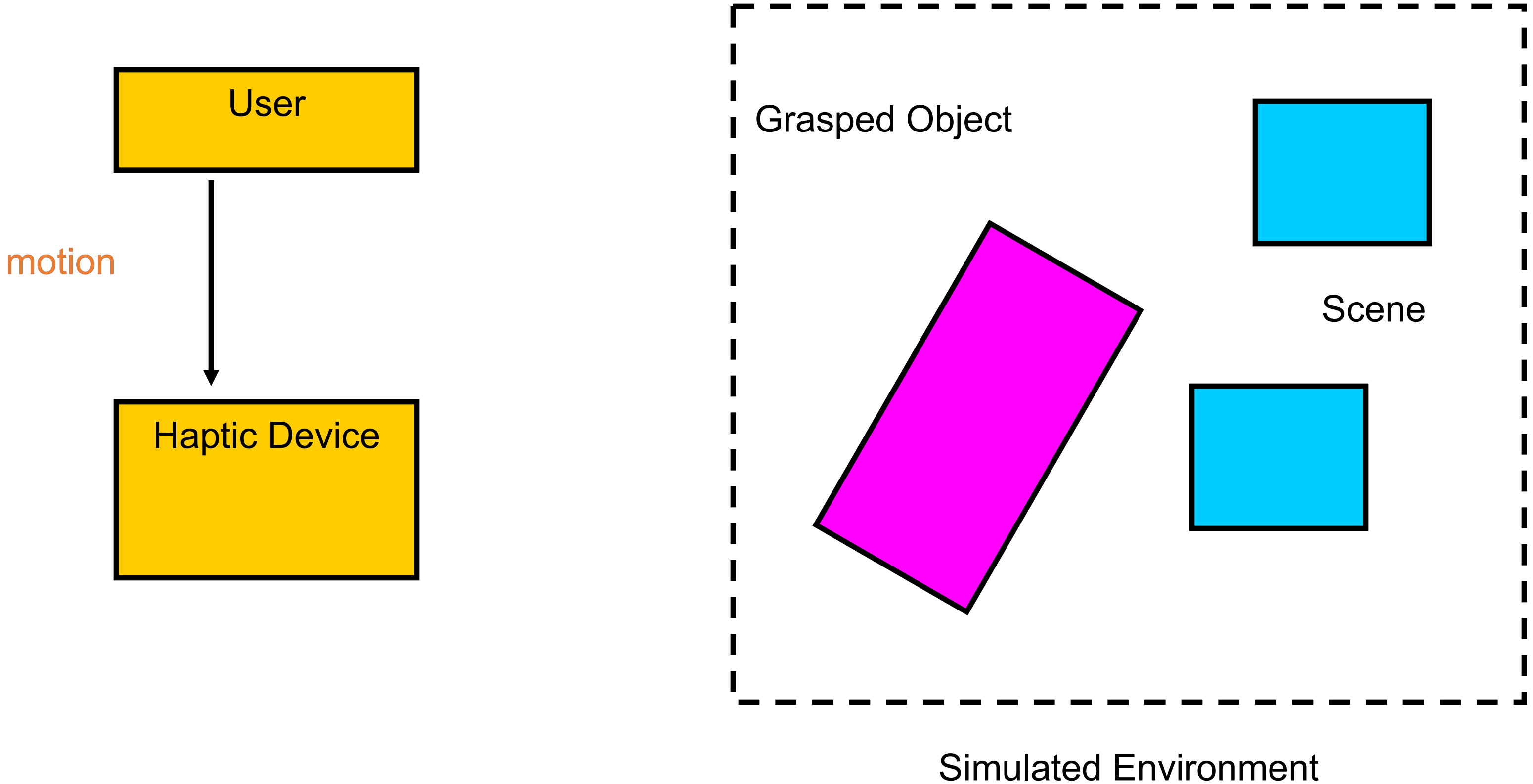


Lawrence, 2000

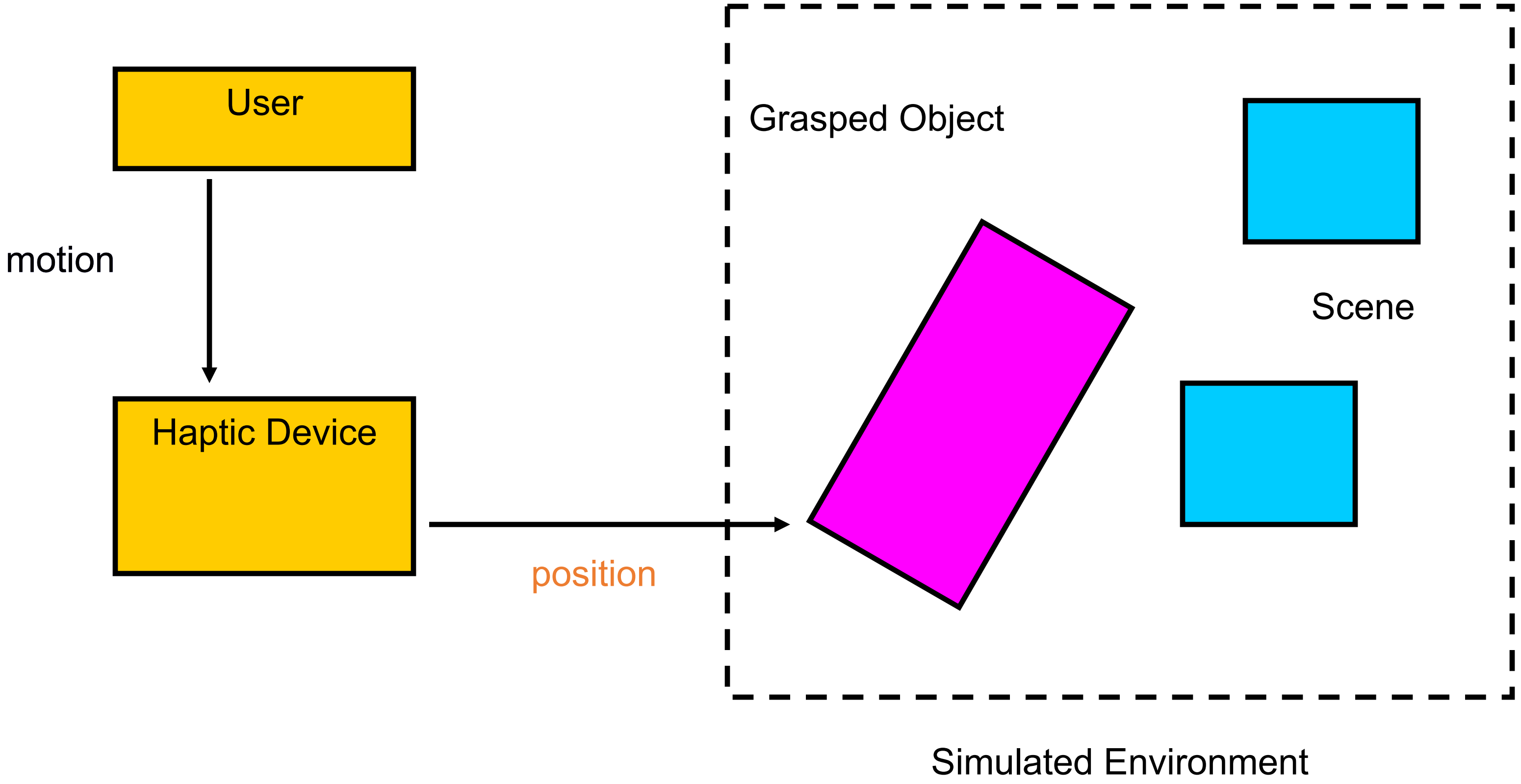
System Video



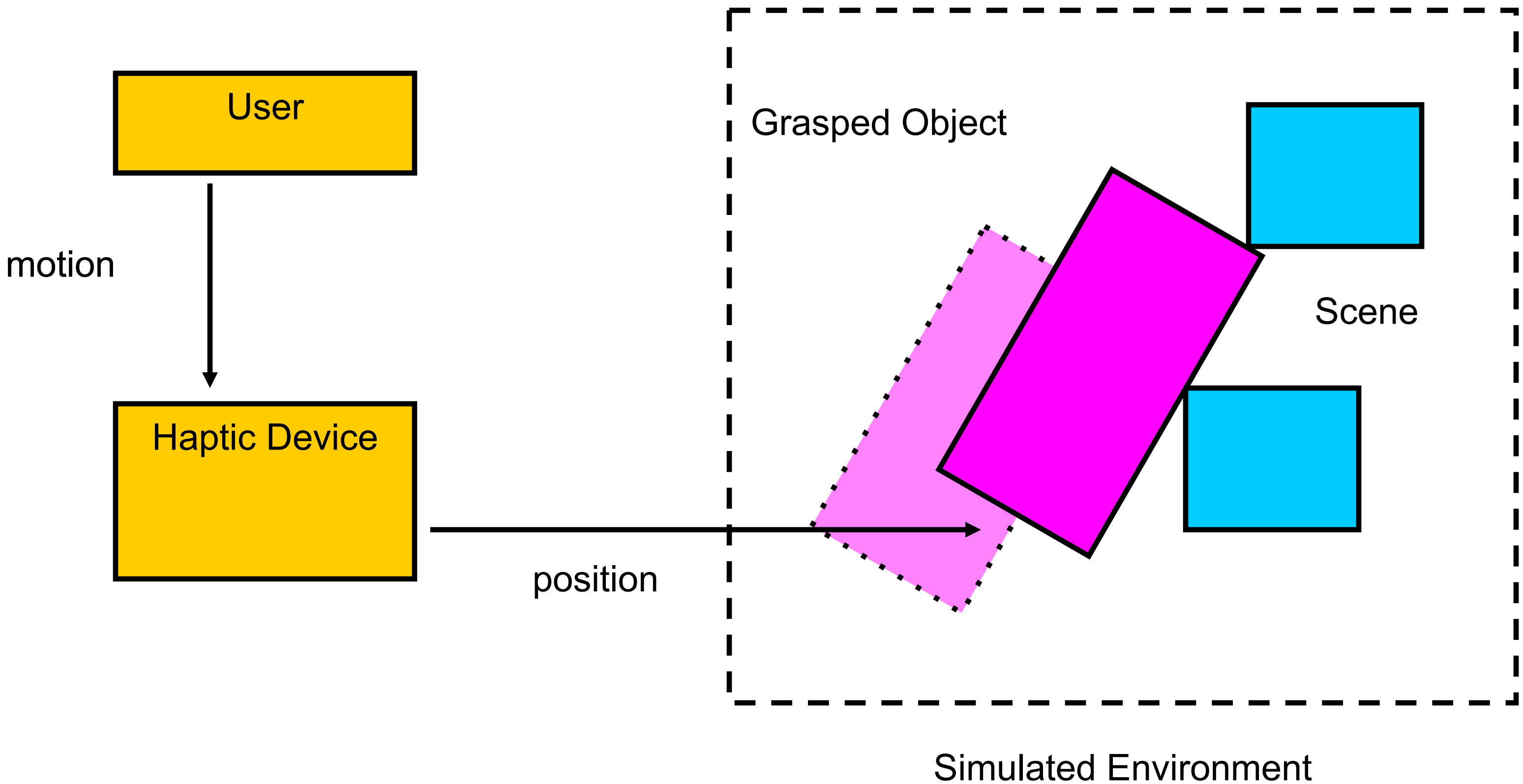
Basic Pipeline



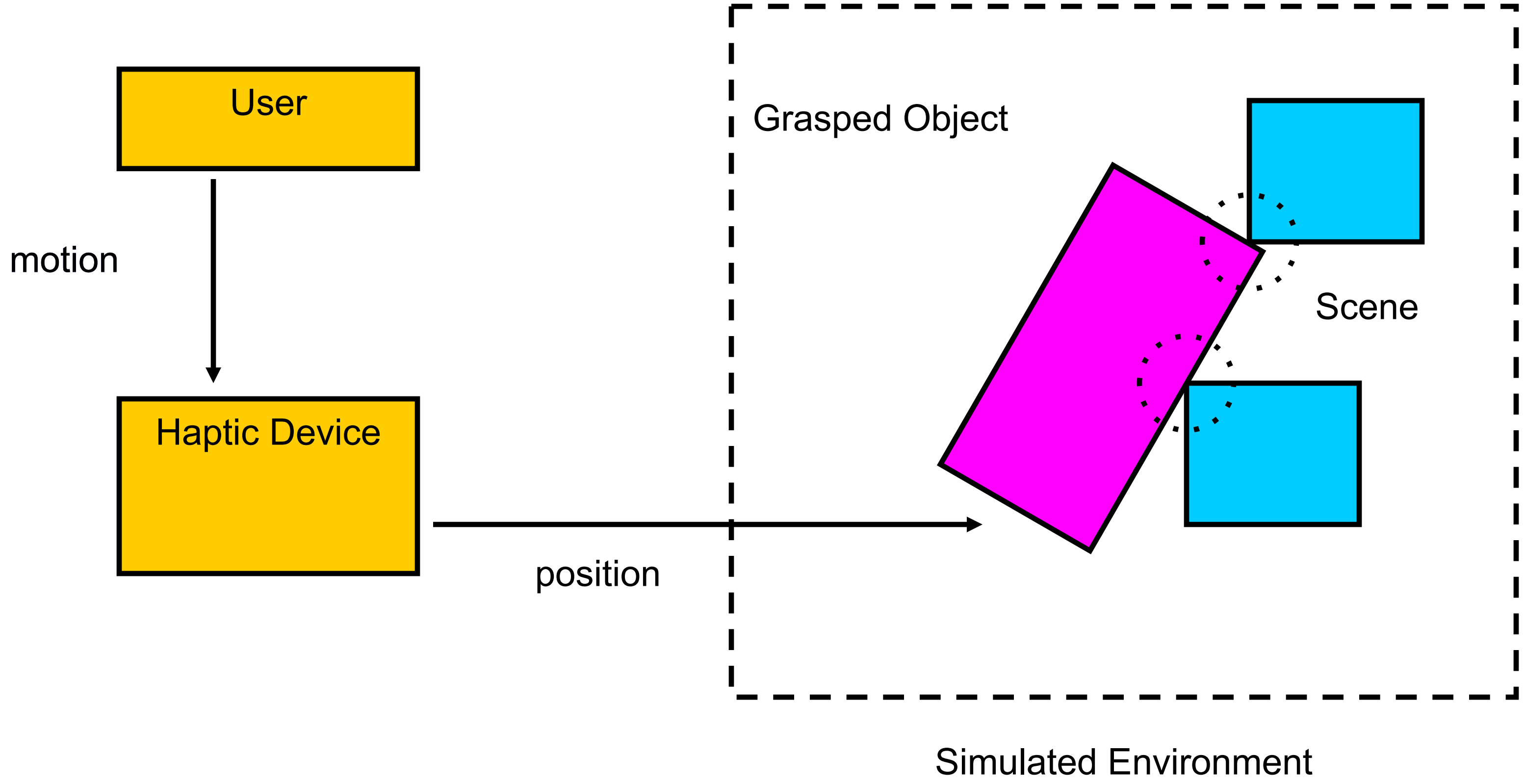
Basic Pipeline



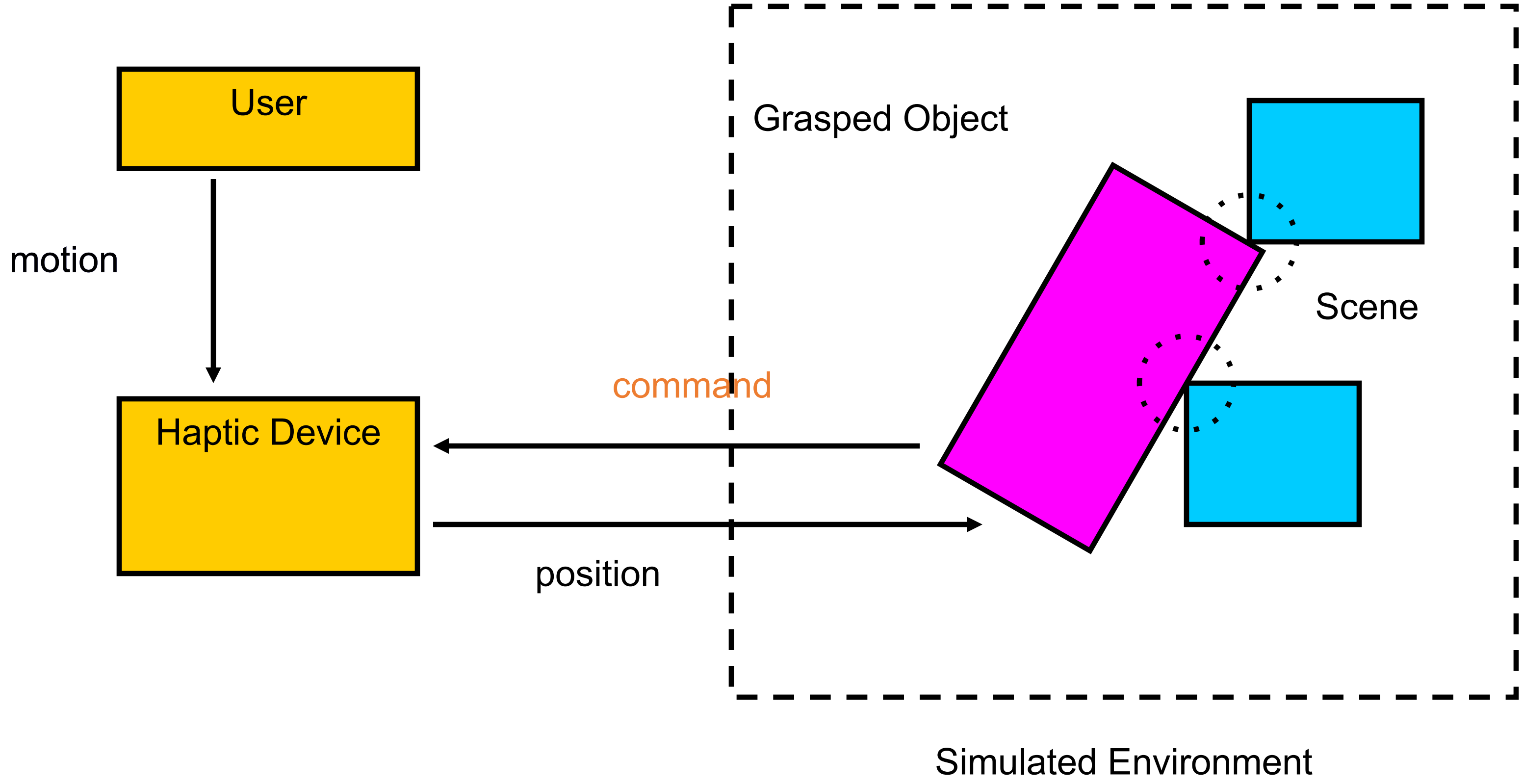
Basic Pipeline



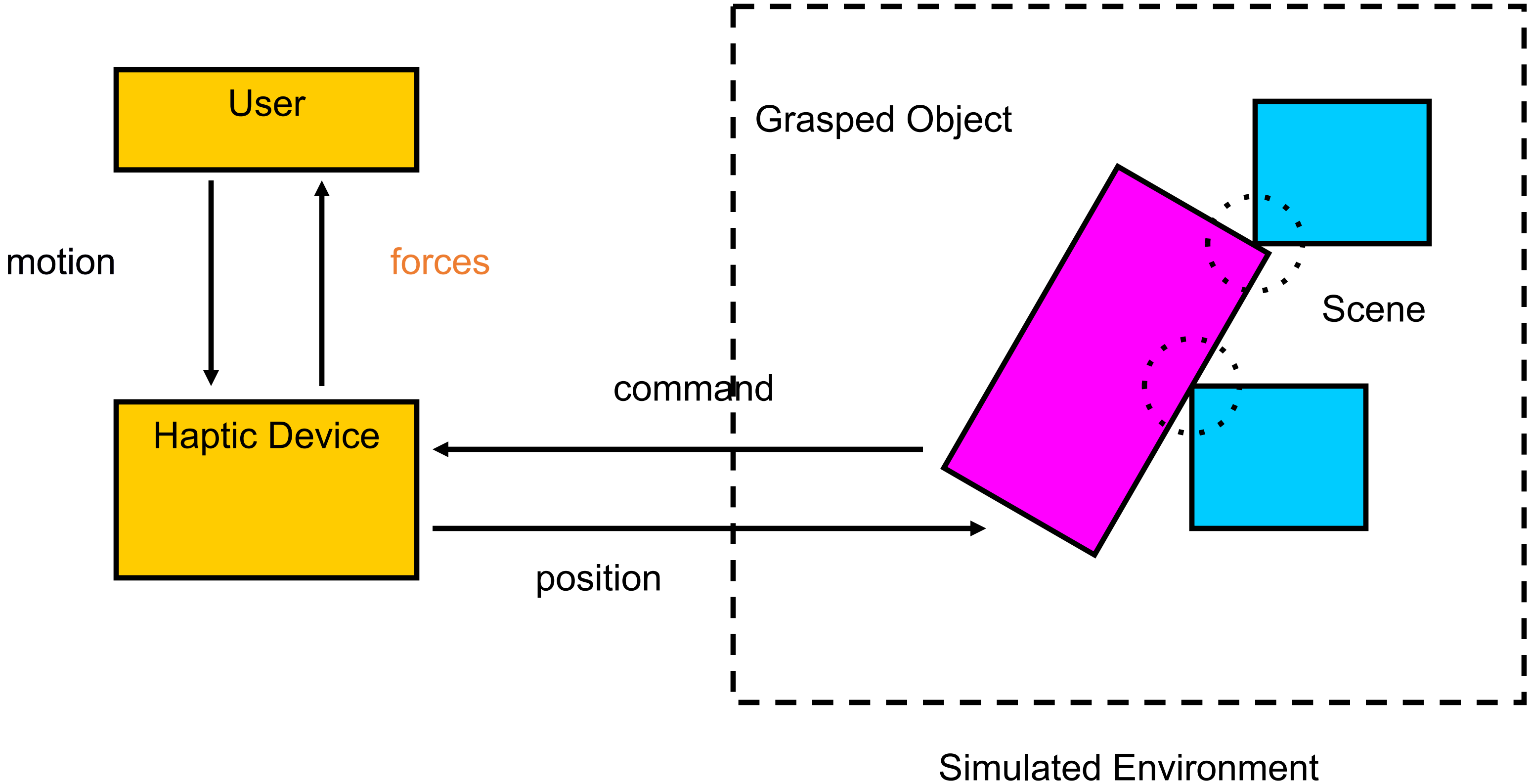
Basic Pipeline



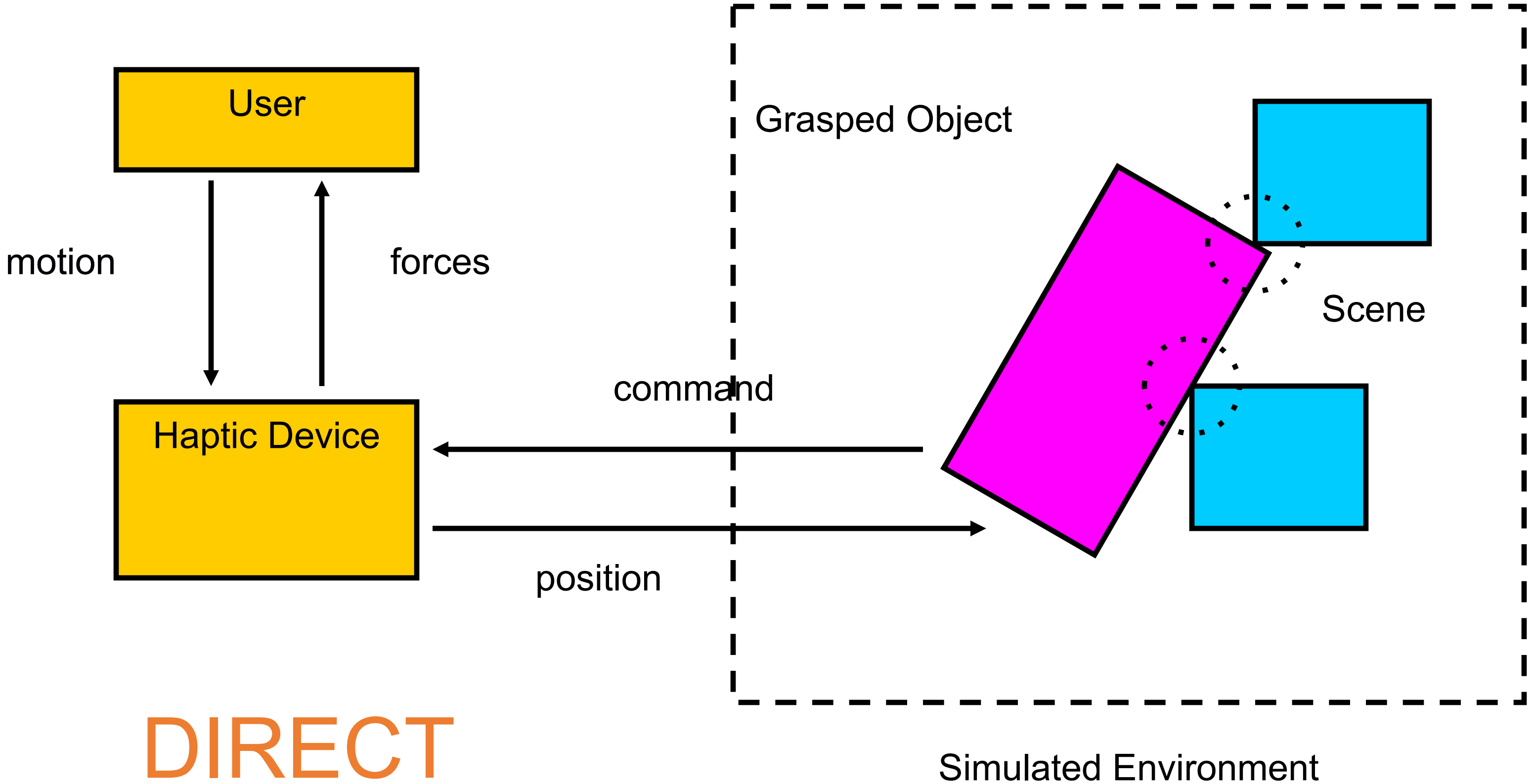
Basic Pipeline



Basic Pipeline

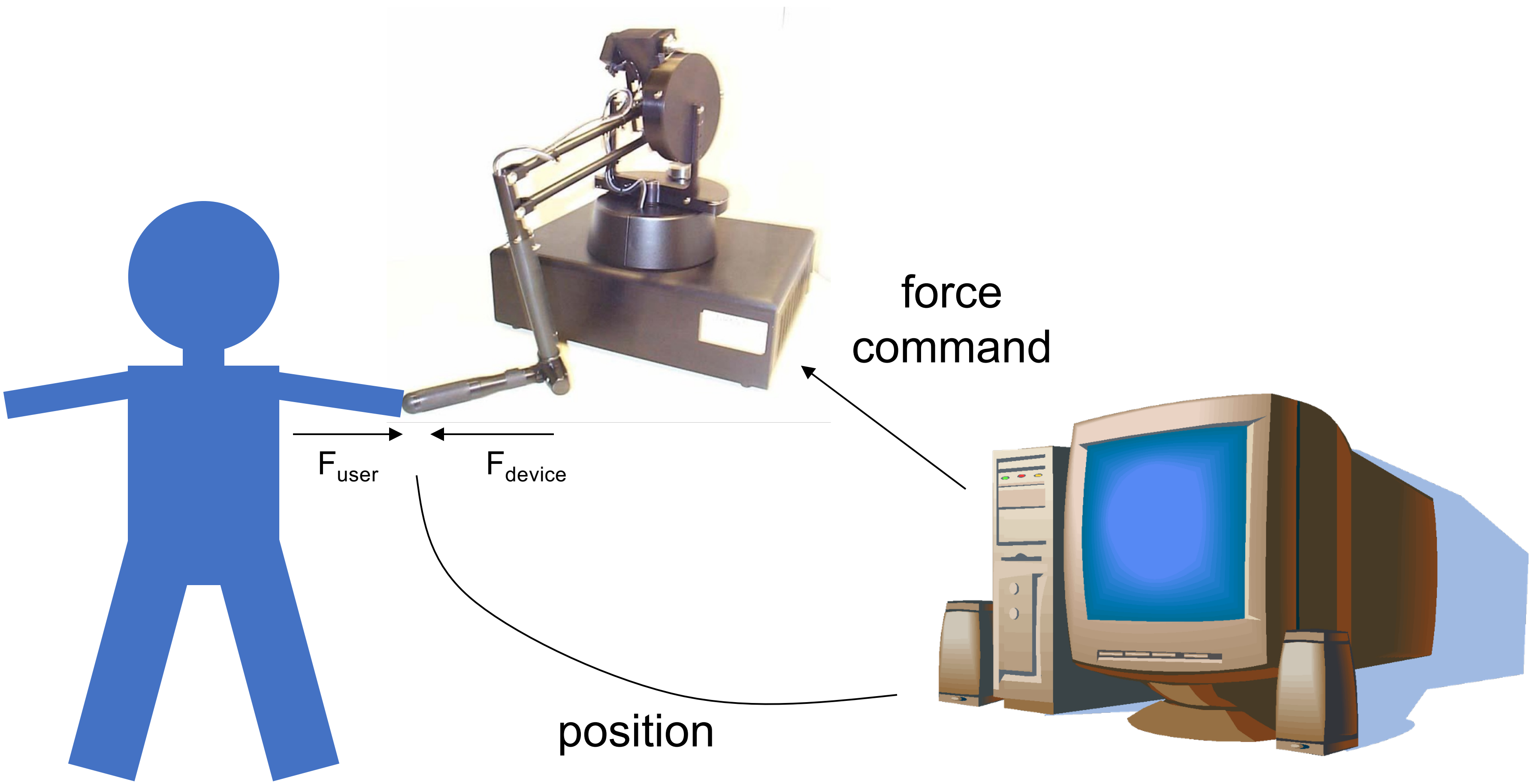


Basic Pipeline



DIRECT
RENDERING

Haptic Rendering Loop

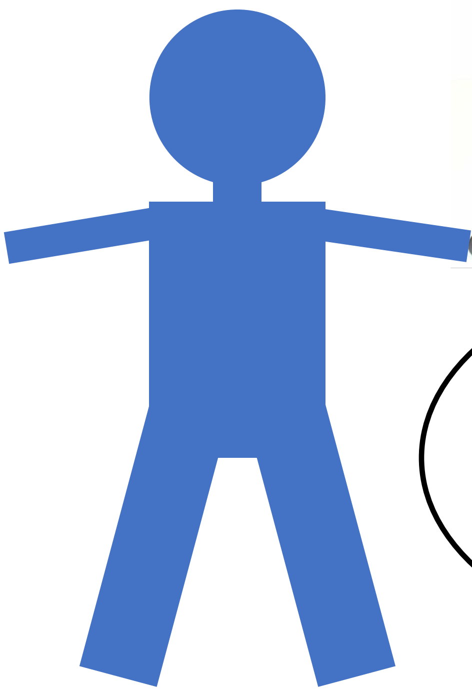


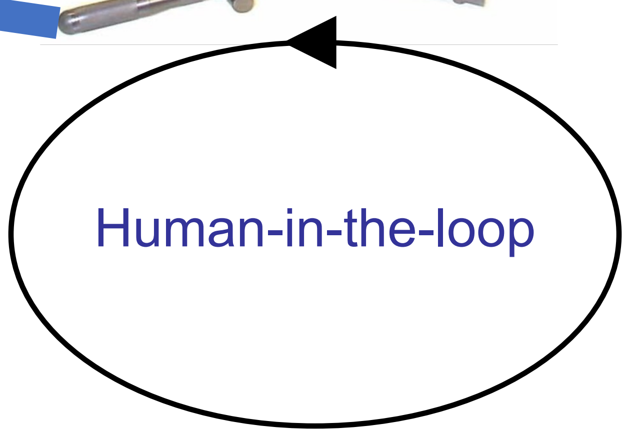
Problem of Haptic Rendering

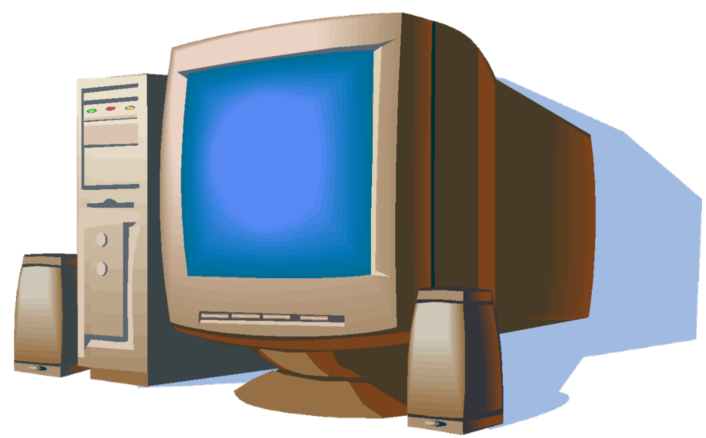
1. The user becomes part of the simulation loop.
2. 1KHz is necessary so that the whole system doesn't suffer from disturbing oscillations.
    - Think of the analogy with numerical integration of a system with spring, mass and damper, where the frequency of the haptic loop sets the integration step.
3. The Phantom haptic devices run their control loop at 1KHz.
4. Consequence: we are very limited on the amount of computation that we can do.

Haptic Rendering Loop

- High sensitivity to instabilities!!
- High update rates required!!
  (kHz for high stiffness)







Key Challenges

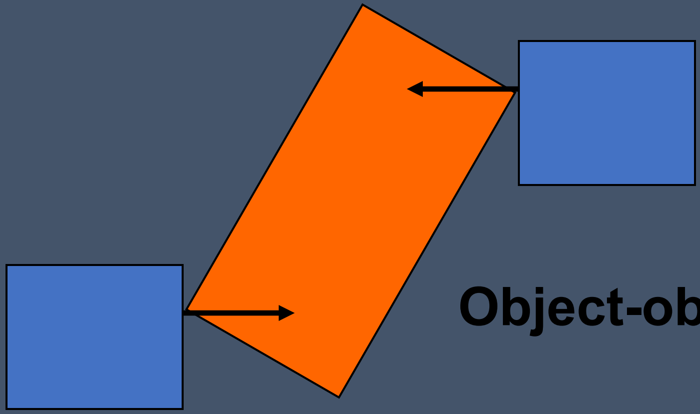
- Collision Detection
  - Choice of representation and algorithm
- Interaction Paradigm
  - Penalty forces vs. ***constraint-based optimization***
  - Virtual coupling vs. direct rendering
  - Newtonian dynamics / Quasi-static approximation
  - Single user *vs.* collaboration

Additional Issues

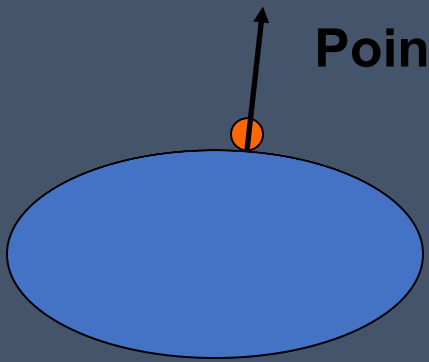
- Decouple haptic and simulation loops?
  - Use intermediate representations?
- Force type and quality
  - How hard does hard contact feel?
  - How free does free-space feel?
    - Repulsive forces?
    - Force artifacts / stability considerations

3DOF Haptics: Introduction

- Output: 3D force -> 3DOF haptics
- Limited to applications where point-object interaction is enough.
  - Haptic visualization of data
  - Painting and sculpting
  - Some medical applications



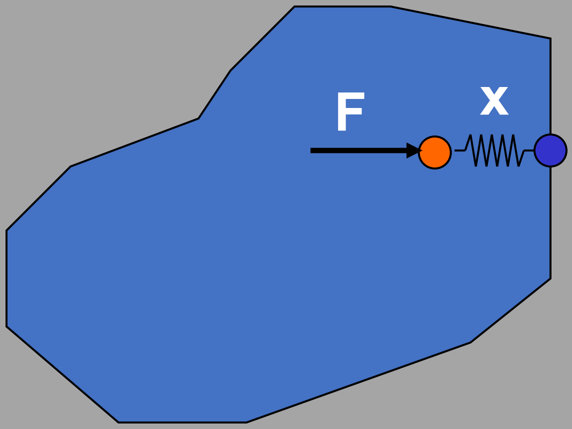
Object-object



Point-object

3DOF Haptics: Basic approach

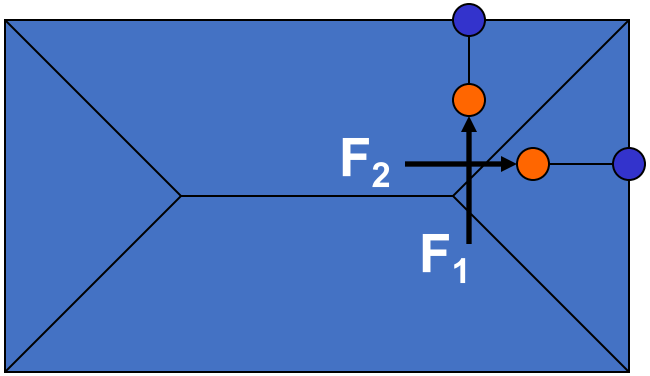
- Check if point penetrates an object.
- Find closest point on the surface.
- ***Penalty-based force***.



$$F = k \cdot x$$

3DOF Haptics: The problems

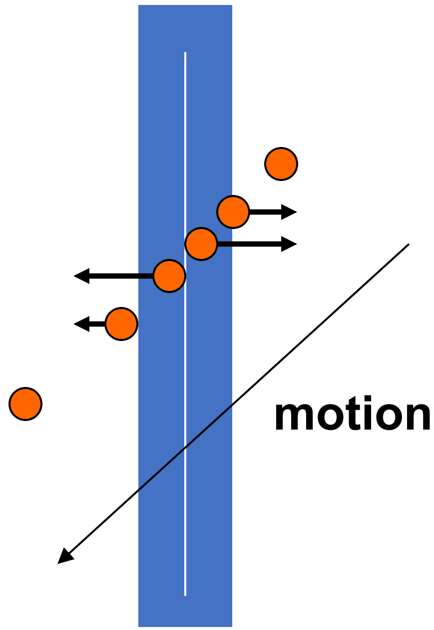
- Force discontinuities when crossing boundaries of internal Voronoi cells.



Unexpected force discontinuities (both in magnitude and direction) are very disturbing!

3DOF Haptics: The problems

- Pop-through thin objects.



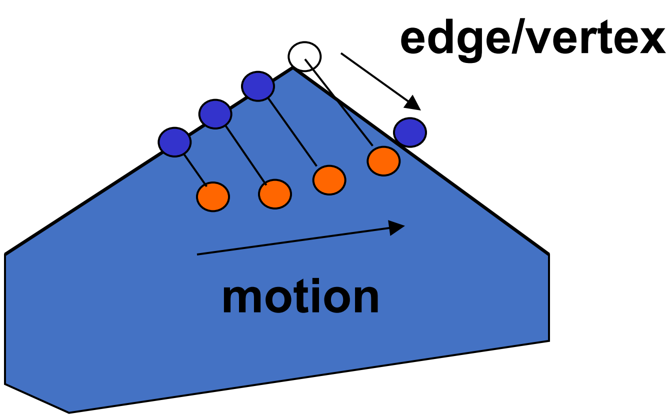
After the mid line is crossed, the force helps popping through.

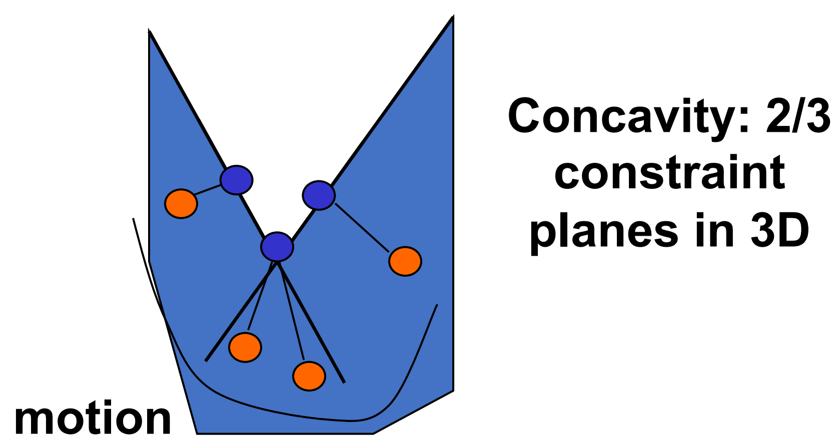
3DOF Haptics: ***God-object***

- Zilles and Salisbury, Haptics Symp. 1995.
- Use the position of the haptic interface point (HIP) and a set of local constraint surfaces to compute the position of god-object (GO).
- Constraint surfaces defined using heuristics.
- Compute GO as the point that minimizes the distance from HIP to the constraint surfaces. Lagrange multipliers.

3DOF Haptics: God-object

- Constraint surfaces:
  - Surfaces impeding motion
  - GO is outside (orientation test) and in the extension of the surface.
  - The HIP is inside the surface.



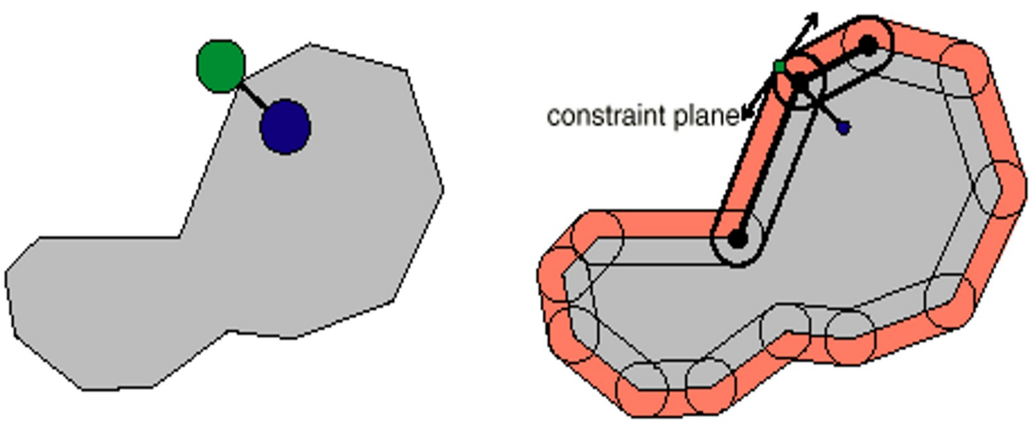


3DOF Haptics: ***Virtual proxy***

- Ruspini et al., SIGGRAPH 1997.
- Based on god-object.
- Virtual proxy is a small sphere, instead of a point. Use configuration-space obstacles (C-obstacles), from robotics.
- More formal definition of constraint planes.
- Implementation of additional features, based on relocation of the virtual proxy.

3DOF Haptics: Virtual proxy

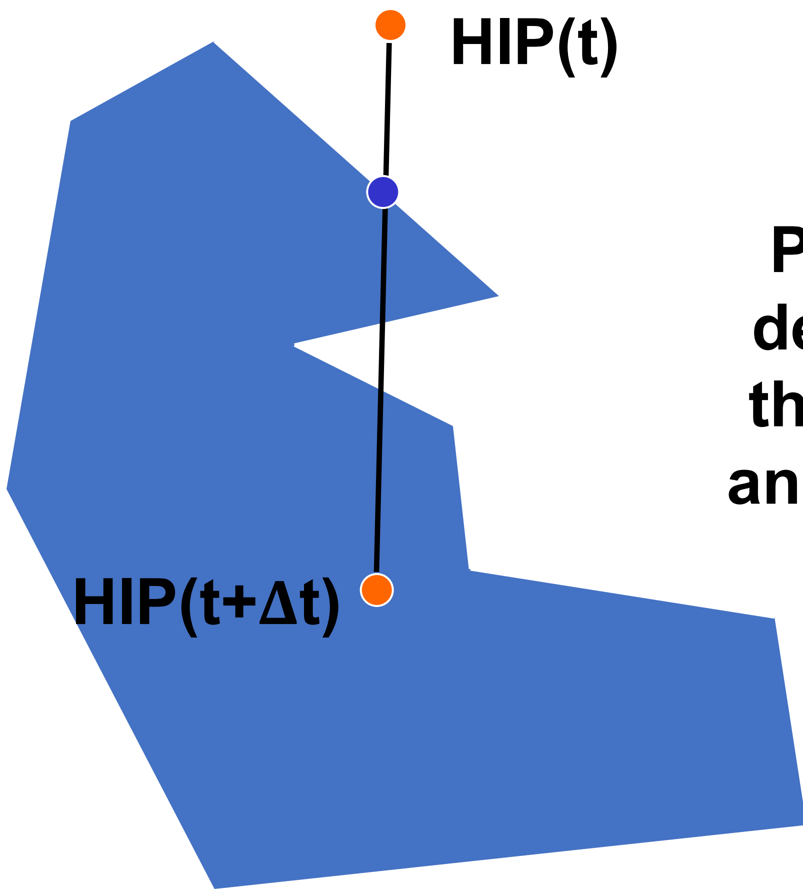
- C-obstacles: for a spherical object, is reduced to computing offset surfaces at a distance equal to the radius of the sphere.
- Check the HIP against the offset surface.
- This is done to avoid problems with small gaps in the mesh.



3DOF Haptics: Virtual proxy

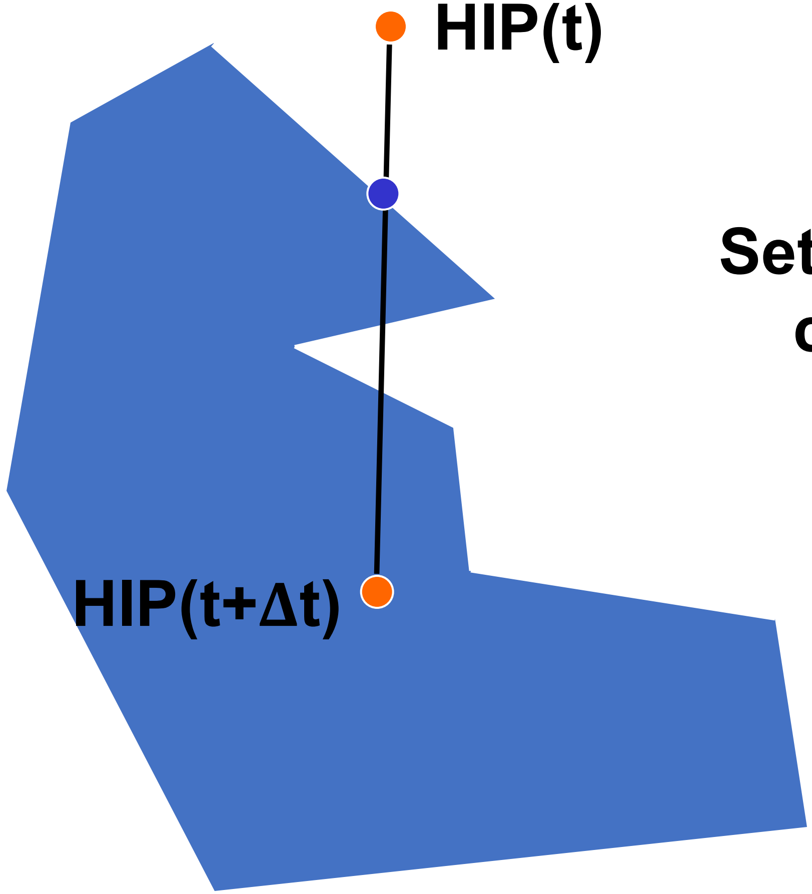
- Finding the virtual proxy is based on an iterative search.
- Basically, find subgoals based on the same distance minimization as for the god-object.
- At each subgoal, all the planes that go through that point are potential constraints. The minimum set of active constraints is selected.
- If the subgoal is in free space, set as new subgoal the HIP. The path might intersect the C-obstacles. Add the first plane intersected as a constraint and the intersection point as the current subgoal.
- The process ends when the virtual proxy becomes stable.

3DOF Haptics: Virtual proxy



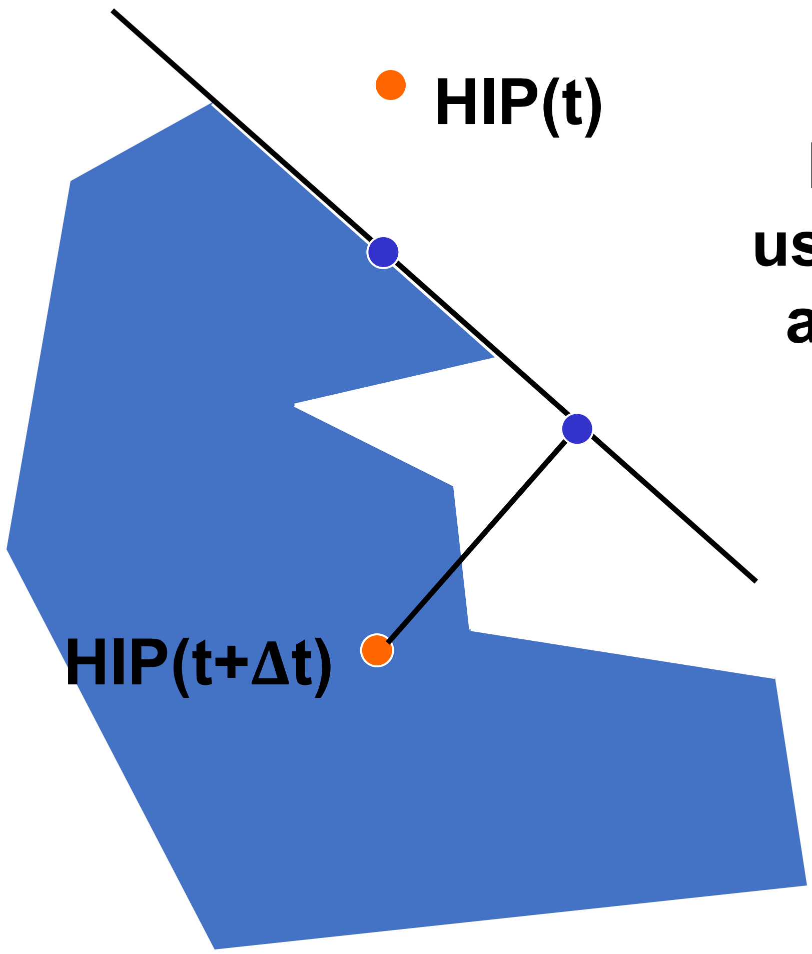
Perform collision detection between the path of the HIP and the C-obstacles

3DOF Haptics: Virtual proxy



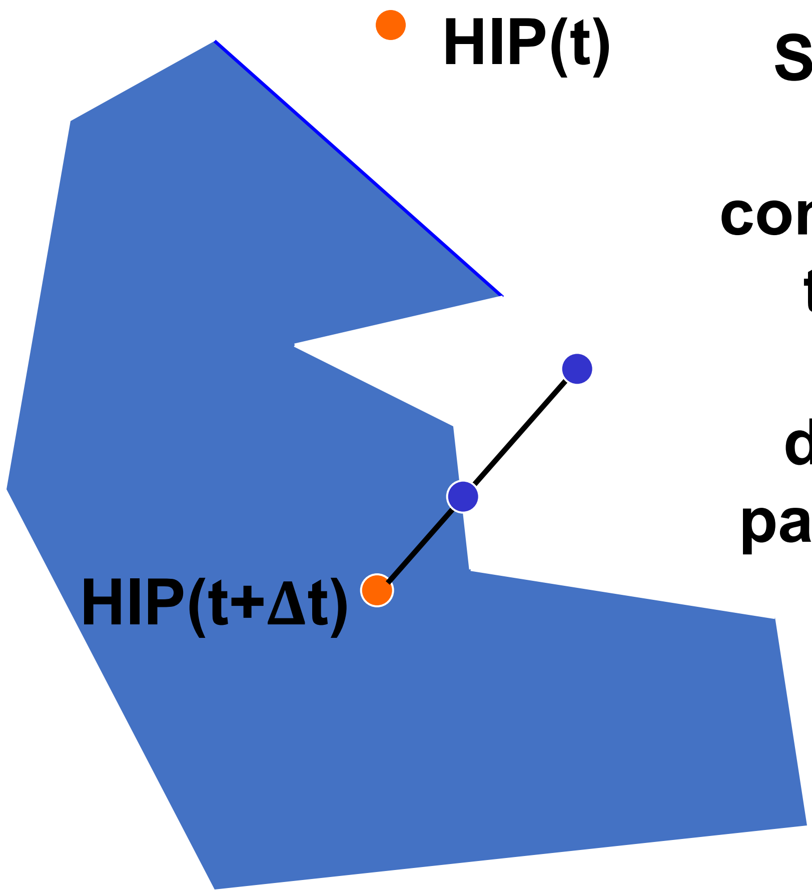
Set the subgoal and the constraint plane(s)

3DOF Haptics: Virtual proxy



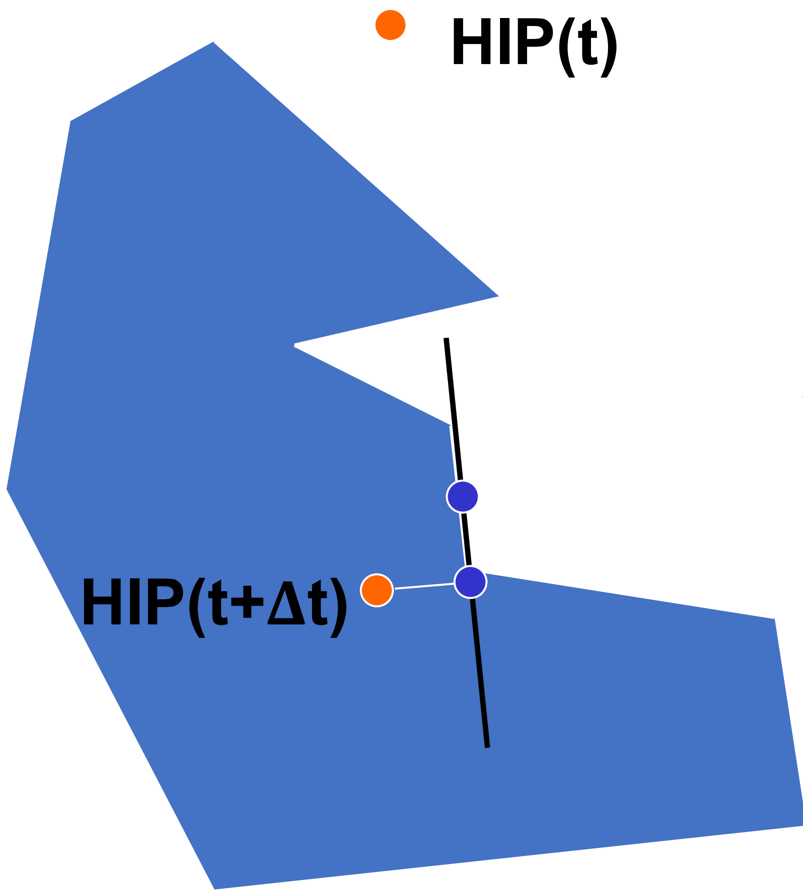
Find a new subgoal using the active planes and the minimization based on Lagrange multipliers

3DOF Haptics: Virtual proxy



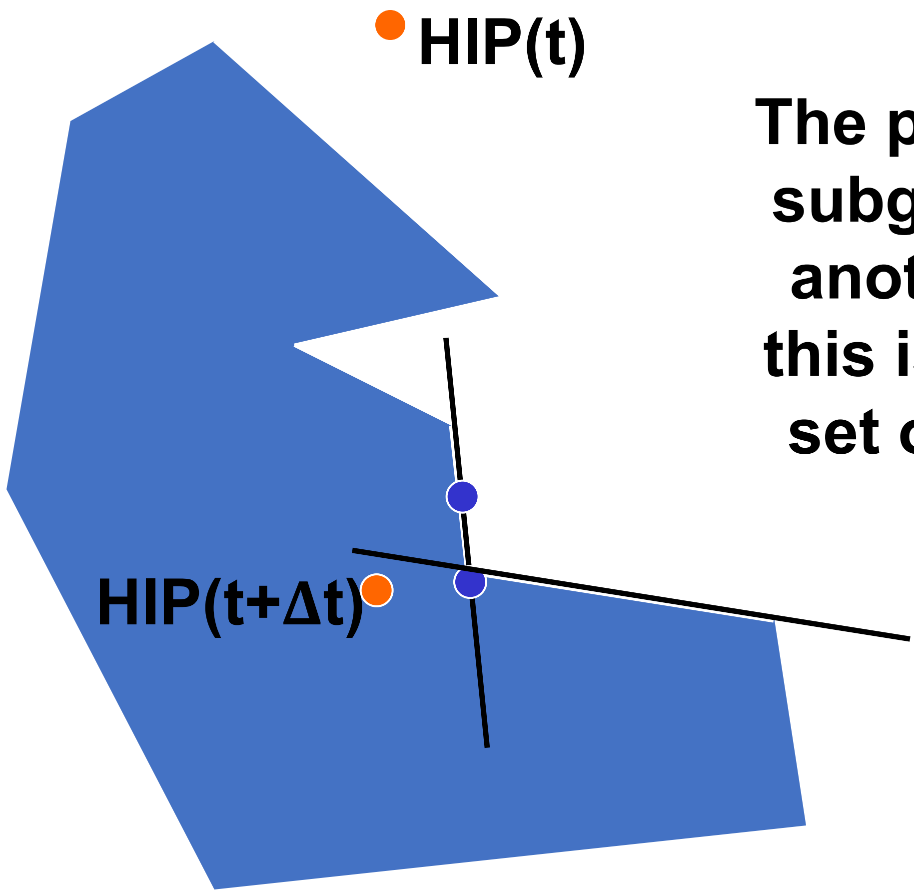
Since the subgoal is in free space, drop the constraints, set the HIP as the new subgoal and perform collision detection between the path and the C-obstacles

3DOF Haptics: Virtual proxy



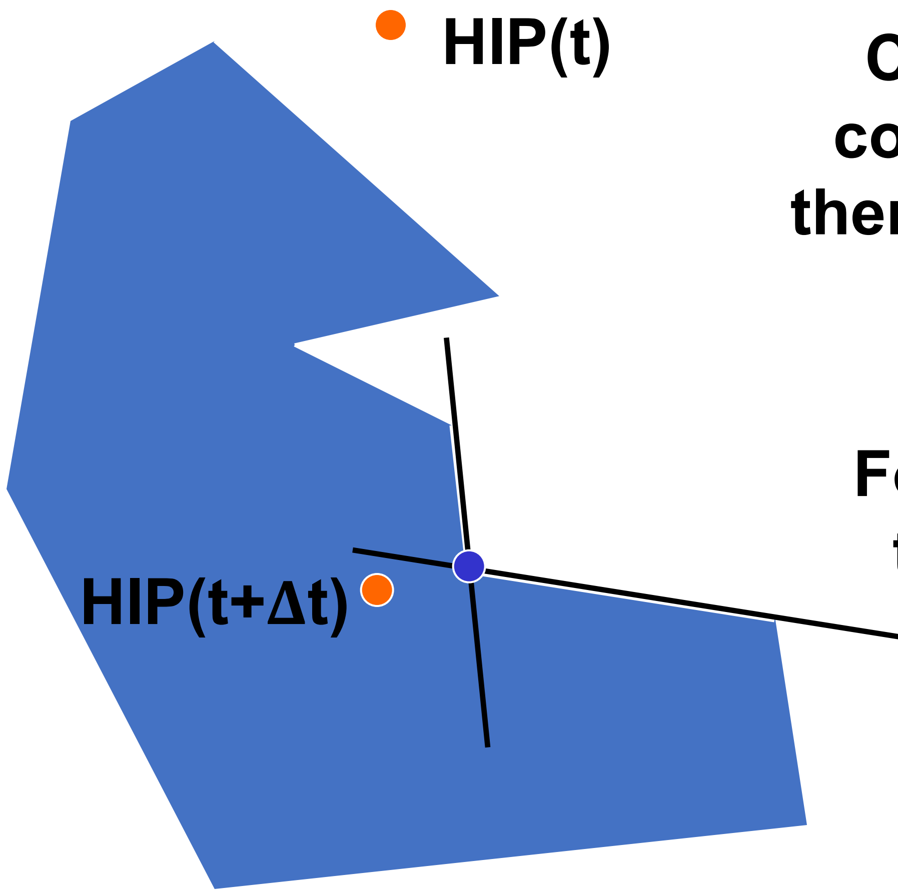
Recompute subgoal with new constraints

3DOF Haptics: Virtual proxy



The path to the new subgoal intersects another plane, so this is added to the set of constraints

3DOF Haptics: Virtual proxy



Compute active constraints (in 2D there are only 2) and find subgoal

For this example, this is the final position of the virtual proxy

3DOF Haptics: Virtual proxy

- Quadratic programming approach:
  - The constraint planes define an open convex region (bounded by the plane at infinity).
  - The function to minimize is the distance from the haptic device (HIP) to the new subgoal (VP_{i+1}):

$$C = \left\| VP_{i+1} - HIP(t + \Delta t) \right\| \qquad \textbf{Quadratic function}$$

  - The translation from the current location to the new subgoal cannot intersect the constraint planes. Define linear constraints based on the normals of the planes.

$$N_i \cdot \left(VP_{i+1} - VP_i \right) \geq 0 \qquad \textbf{Linear constraints}$$

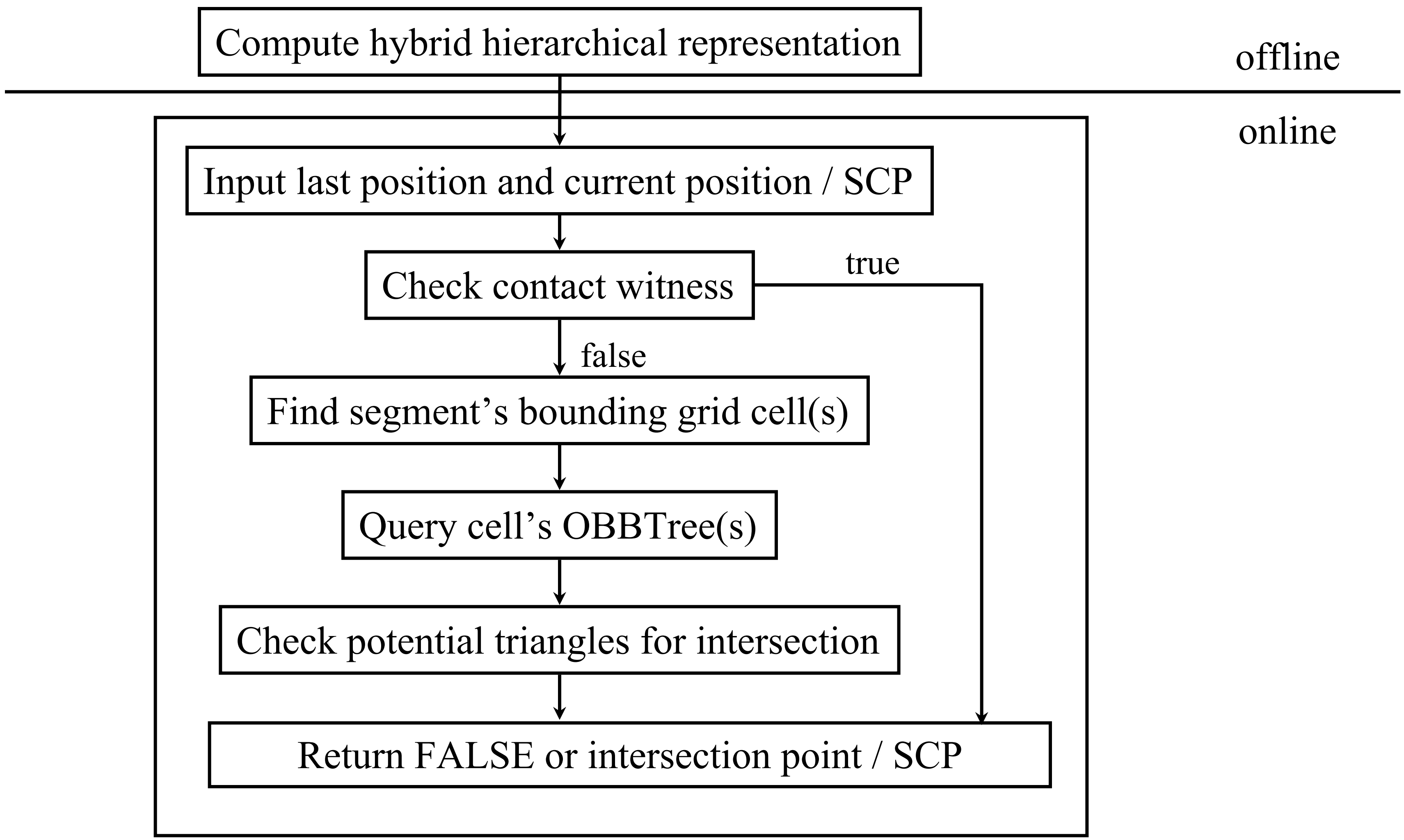
3DOF Haptics: H-Collide

- Collision Detection for 3DOF Haptics
  - Spatial Partitioning
    - Uniform grid implemented using hashing
  - Bounding Volume Hierarchy
    - Oriented bounding box trees
  - Frame-to-Frame Coherence
    - Caching the last “witness”

Gregory, Gottschalk, Taylor & Lin

[VR’99, CGTA’00]

System Architecture

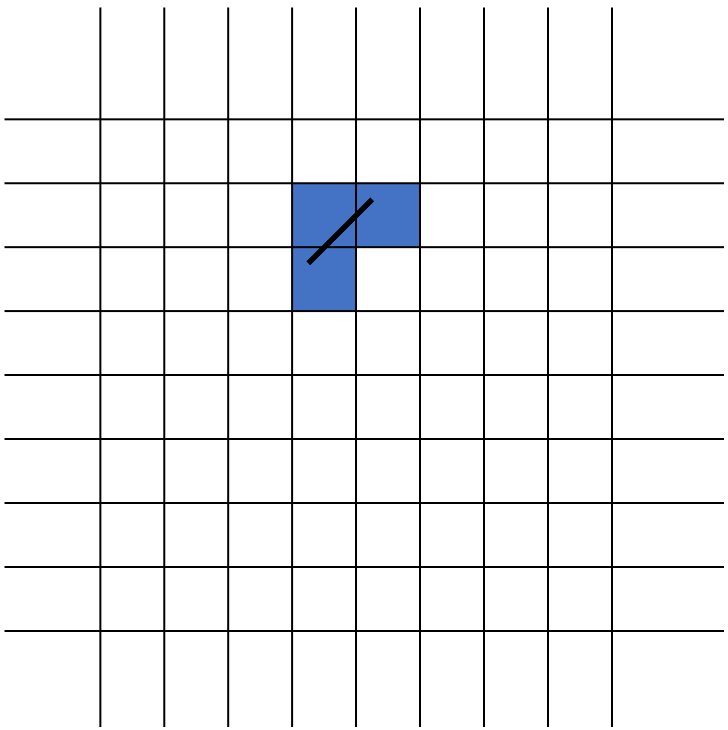


HCOLLIDE Overview (I)

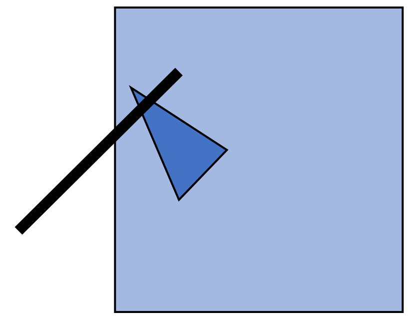
- OFFLINE PROCESS
  - Pre-compute hybrid representation, consisting of uniform grids and each contains an OBBTree.
- RUNTIME PROCESS
  - Identify “contact region” by uniform spatial partitioning (implemented with hash table)
  - Locate the exact contact points by querying and traversing the OBBTrees
  - Frame-to-frame coherence by caching the last “witness”
  - Find the projected surface contact point

H-Collide Overview (II)

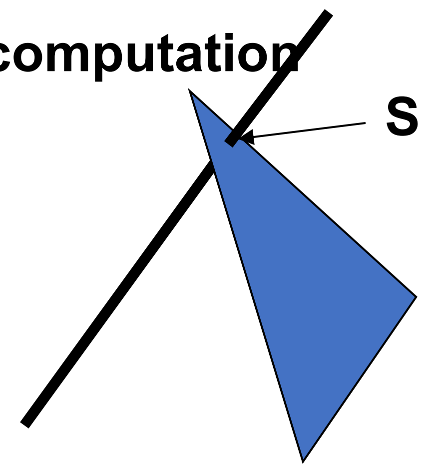
Spatial partitioning with hash table



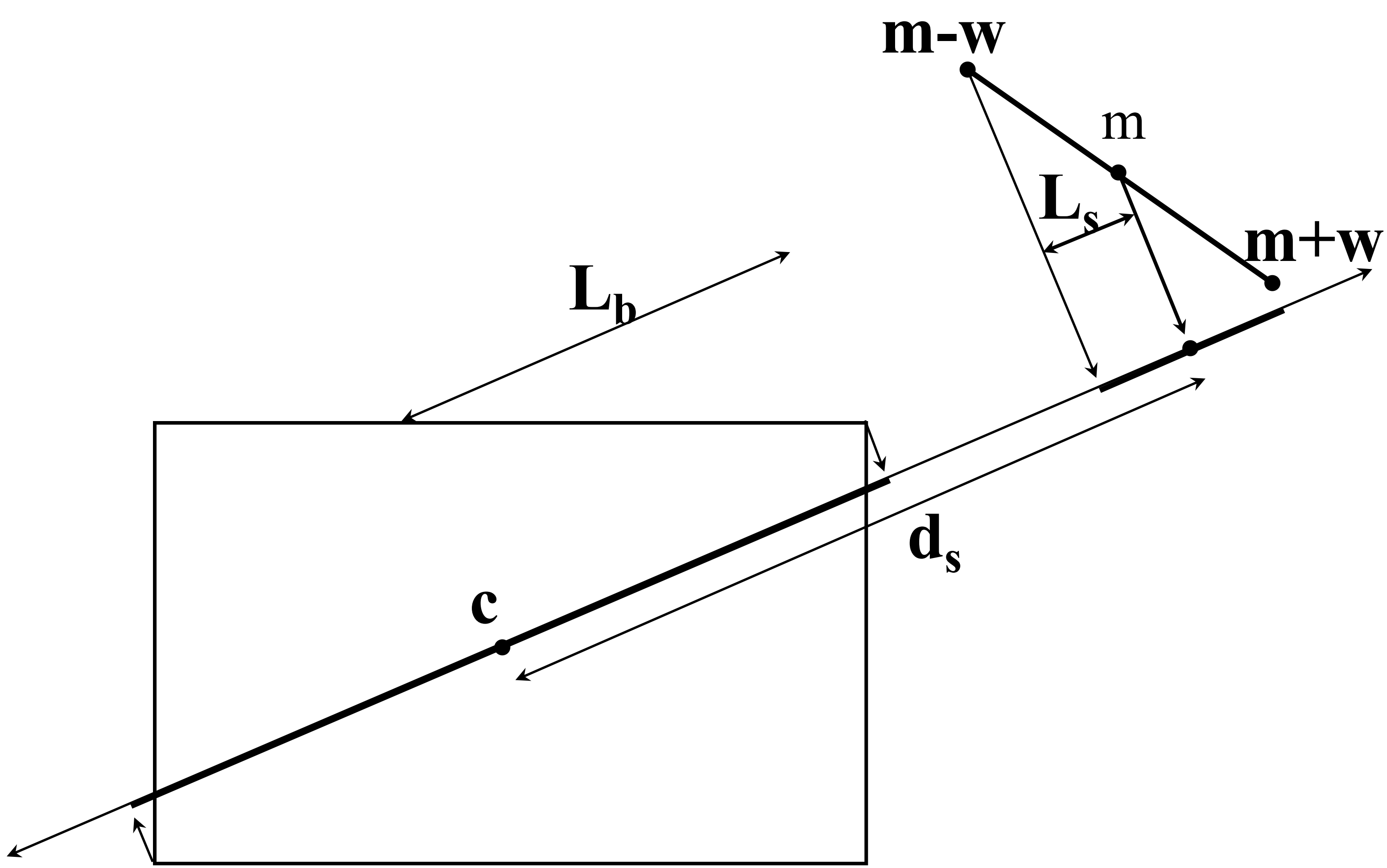
Ray Vs. OBBTree test



SCP computation



Line-OBB Overlap Test



HCOLLIDE Pseudo Code

$X = | w_x |$

$Y = | w_y |$

$Z = | w_z |$

if $| m_x | > X + t_x$ return disjoint

if $| m_y | > Y + t_y$ return disjoint

if $| m_z | > Z + t_z$ return disjoint

if $| m_y w_z - m_z w_y | > t_y Z + t_z Y$ return disjoint

if $| m_x w_z - m_z w_x | > t_x Z + t_z X$ return disjoint

if $| m_x w_y - m_y w_x | > t_x Y + t_y X$ return disjoint

Specialized Overlap Test

- Simple control loop
  - good for micro-coding & SIMD implementation
- Cost: 42-72 arithmetic operations
  - 9 absolute values
  - 6 comparisons
  - 9 addition/subtraction
  - 12 multiplication
  - 36 ops for transformation

HCOLLIDE Hashing

- Hashing Function:

$$h(k) = x + y * num\_cell + z * (num\_cell)^2$$

$$TableLoc = random(h(k)) \ \% \ TableLength$$

- Grid Size Selection:
  - difficult to compute an optimal value for all input models with varying triangulation
  - set the grid size to be the *averaged* edge length of the input model

Optimal Grid Size (I)

Assume --

- Line segment swept out by the probe is small compared to the optimal grid size
- There is only one contacting point with the surface of the object and one triangle in contact with the probe
- The triangulation of the object is uniform & all triangles have nice aspect ratio
- All objects in the scene are static and rigid

Optimal Grid Size (II)

- N: total number of triangle
- M: averaged number of triangles per cell

$$C_r = (2 \log N + 1)\, C_{obb} + C_{tri}$$

$$C_g = M\, C_{tri} + C_l$$

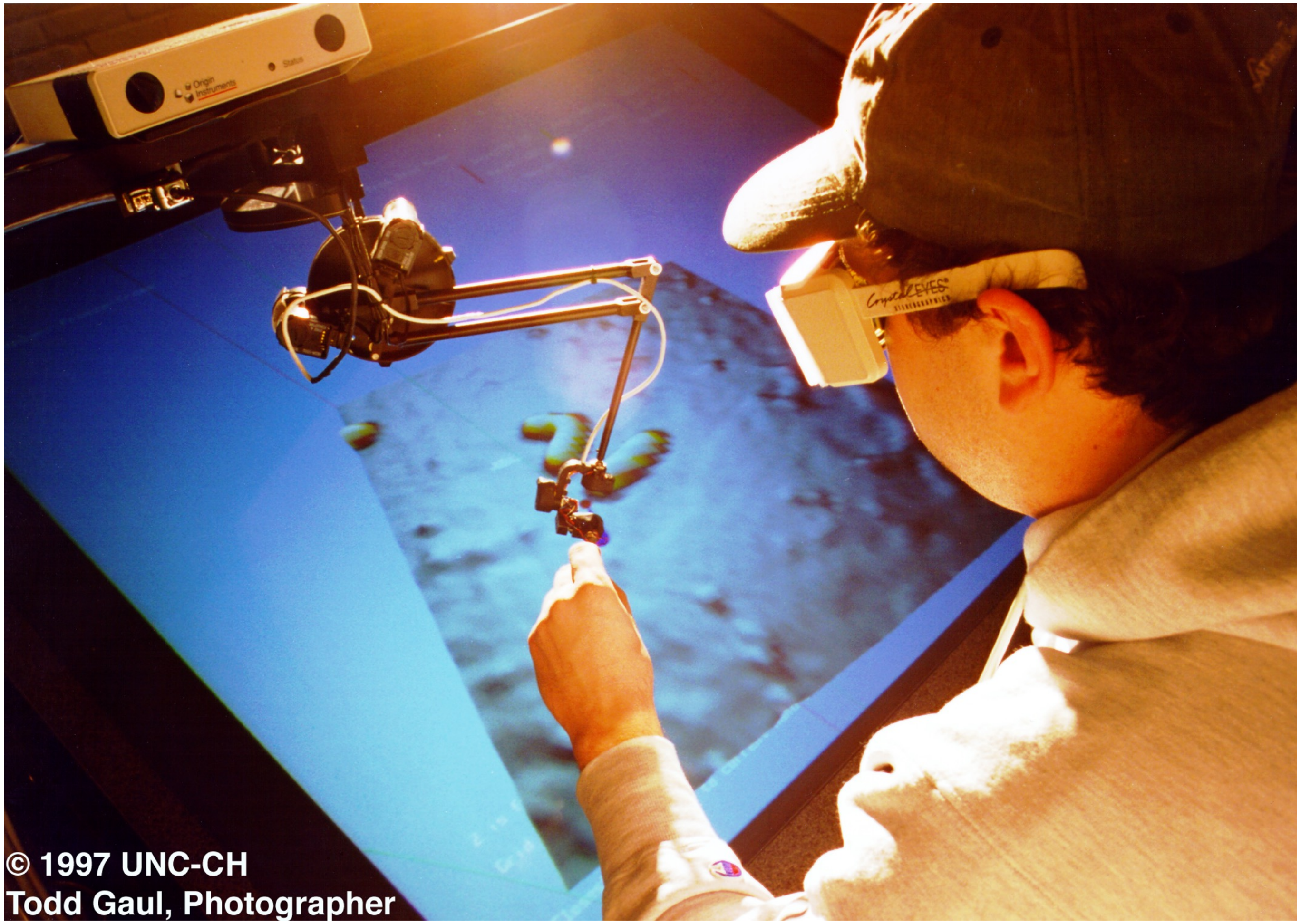
$$C_h = (2 \log M + 1)\, C_{obb} + C_{tri} + C_l$$

$$(2 \log M * C_{obb}/C_{tri} + 1 + C_{obb}/C_{tri}) < M < N / 2^{C_l/2C_{obb}}$$

According to our implementation:

C_l / C_{obb}:0.9-5.5 and C_{obb}/C_{tri} : 0.764-4.0

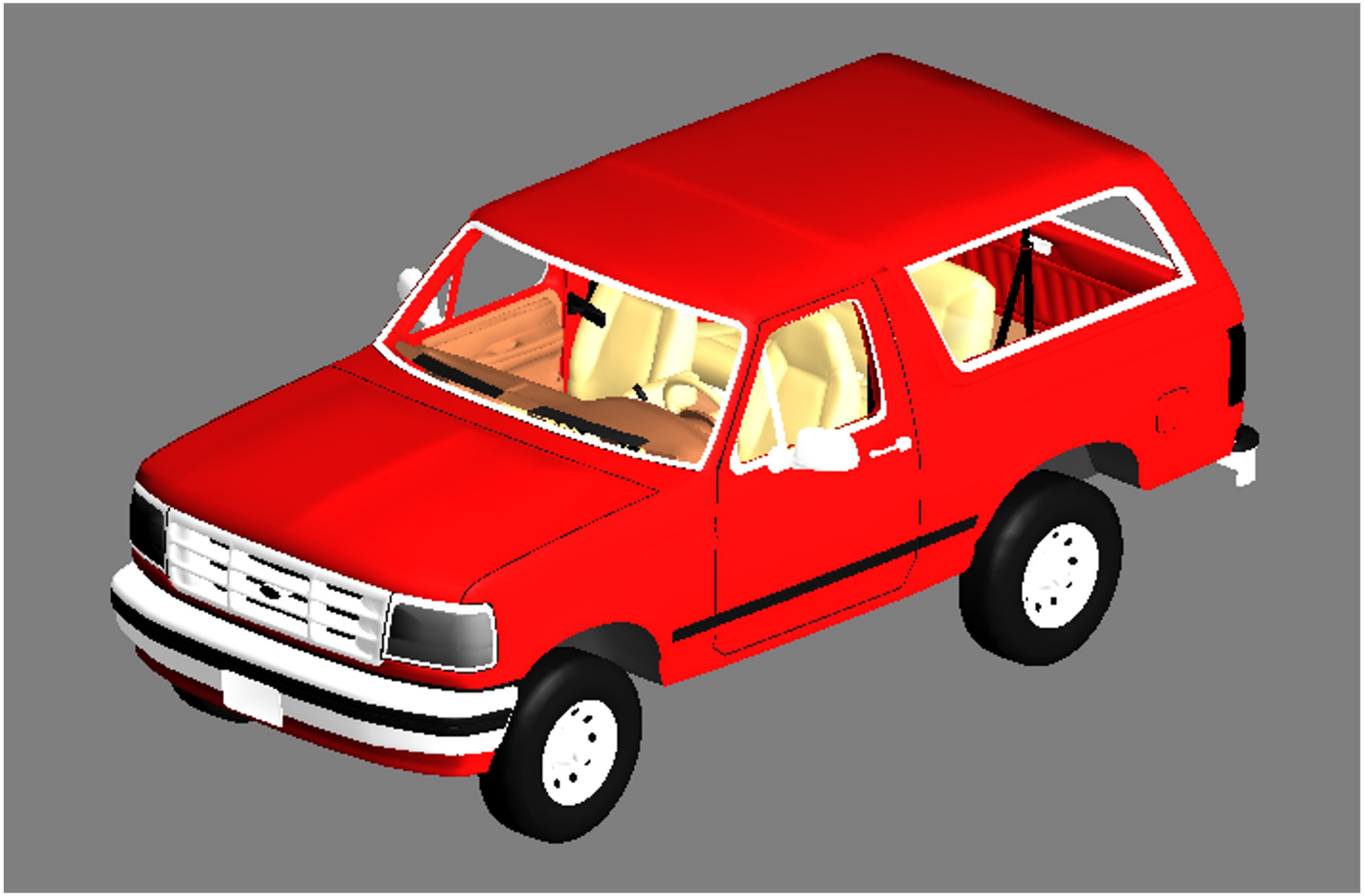
Nano-Surfaces



HCOLLIDE: Timing on Nano-Surfaces (msec)

Method	**Hash Grid**	**Hybrid**	**OBB Tree**	**Ghost**
Ave Col. Hit	0.0138	0.0101	0.0134	0.332
Worst Col. Hit	0.125	**0.168**	0.0663	**0.724**
Ave Col. Miss	0.00739	0.00508	0.00422	0.0109
Worst Col. Miss	0.0347	**0.0377**	0.0613	**0.210**
Ave Int. Hit	0.0428	0.0386	0.0447	0.0851
Worst Int. Hit	0.0877	**0.102**	0.0690	**0.175**
Ave Int. Miss	0.0268	0.0197	0.0213	0.0545
Worst Int. Miss	0.0757	**0.0697**	0.0587	**0.284**
Ave. Query	0.022	**0.016**	0.039	**0.18**

Ford Bronco



HCOLLIDE: Timing on Ford Bronco (msec)

Method	Hash Grid	Hybrid	OBB Tree	Ghost
Ave Col. Hit	0.0113	0.00995	0.0125	0.104
Worst Col. Hit	0.136	**0.132**	0.177	**0.495**
Ave Col. Miss	0.0133	0.00731	0.0189	0.0280
Worst Col. Miss	0.128	**0.0730**	0.137	**0.641**
Ave Int. Hit	0.0566	0.0374	0.609	0.0671
Worst Int. Hit	0.145	**0.105**	0.170	**0.293**
Ave Int. Miss	0.0523	0.0225	0.0452	0.0423
Worst Int. Miss	0.132	**0.133**	0.167	**0.556**
Ave. Query	0.027	**0.014**	0.028	**0.048**

Butterfly

HCOLLIDE: Timing on Butterfly (msec)

Method	**Hash Grid**	**Hybrid**	**OBB Tree**	**Ghost**
Ave Col. Hit	0.0232	0.0204	0.0163	1.33
Worst Col. Hit	0.545	**0.198**	0.100	**5.37**
Ave Col. Miss	0.00896	0.00405	0.00683	0.160
Worst Col. Miss	0.237	**0.139**	0.121	**3.15**
Ave Int. Hit	0.228	0.0659	0.0704	0.509
Worst Int. Hit	0.104	**0.138**	0.103	**1.952**
Ave Int. Miss	0.258	0.0279	0.0256	0.229
Worst Int. Miss	0.0544	**0.131**	0.0977	**3.28**
Ave. Query	0.030	**0.016**	0.016	**0.32**

HCOLLIDE: Algorithm Analysis

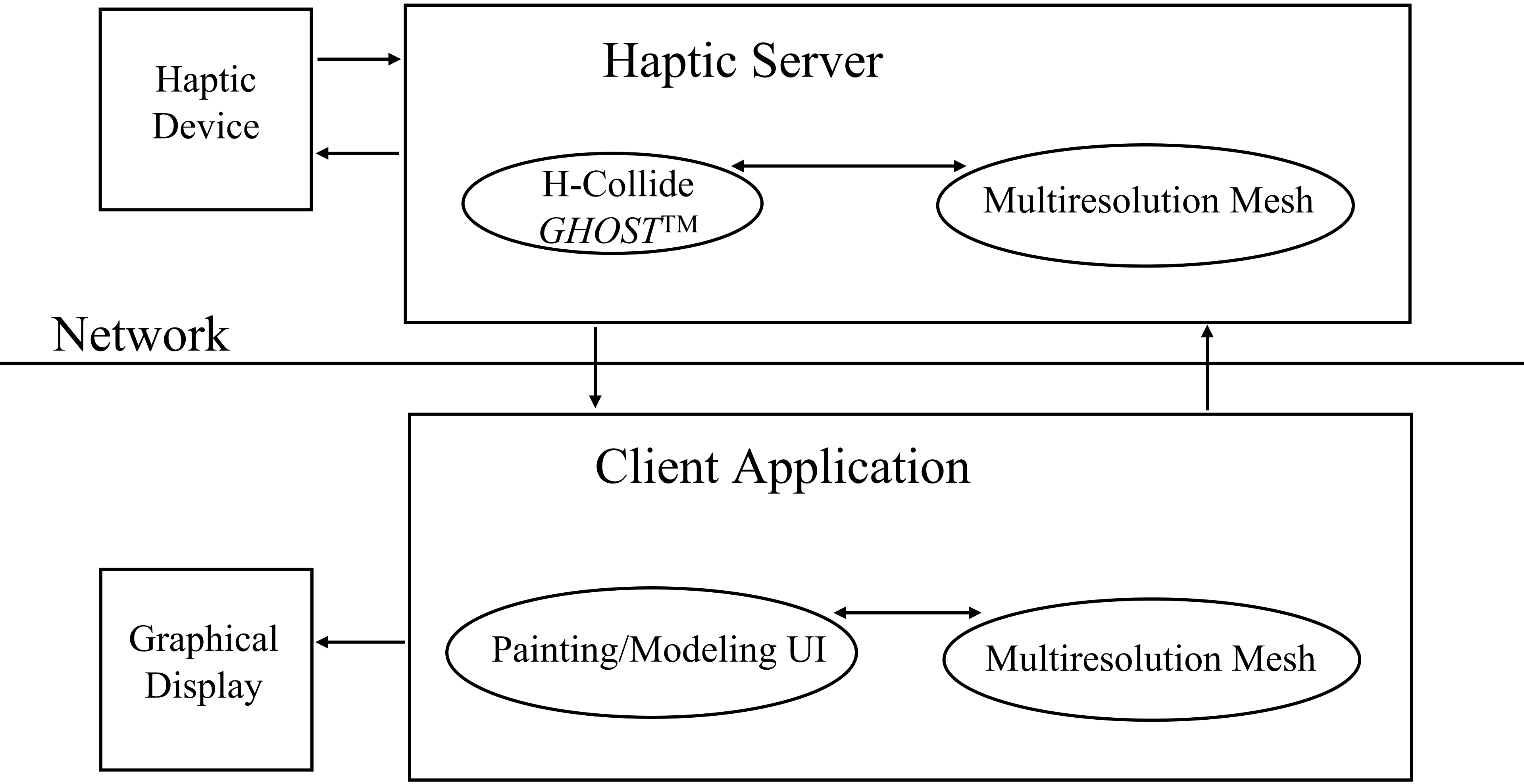
- At least 2-20 times faster than *GHOST* on the models we have tested on
- Hybrid is the most favorable and capable of maintaining kHZ rate
- If the number of triangles per grid cell is relatively small compared to the model size, then hybrid method runs in *constant time*

Applications: *inTouch*

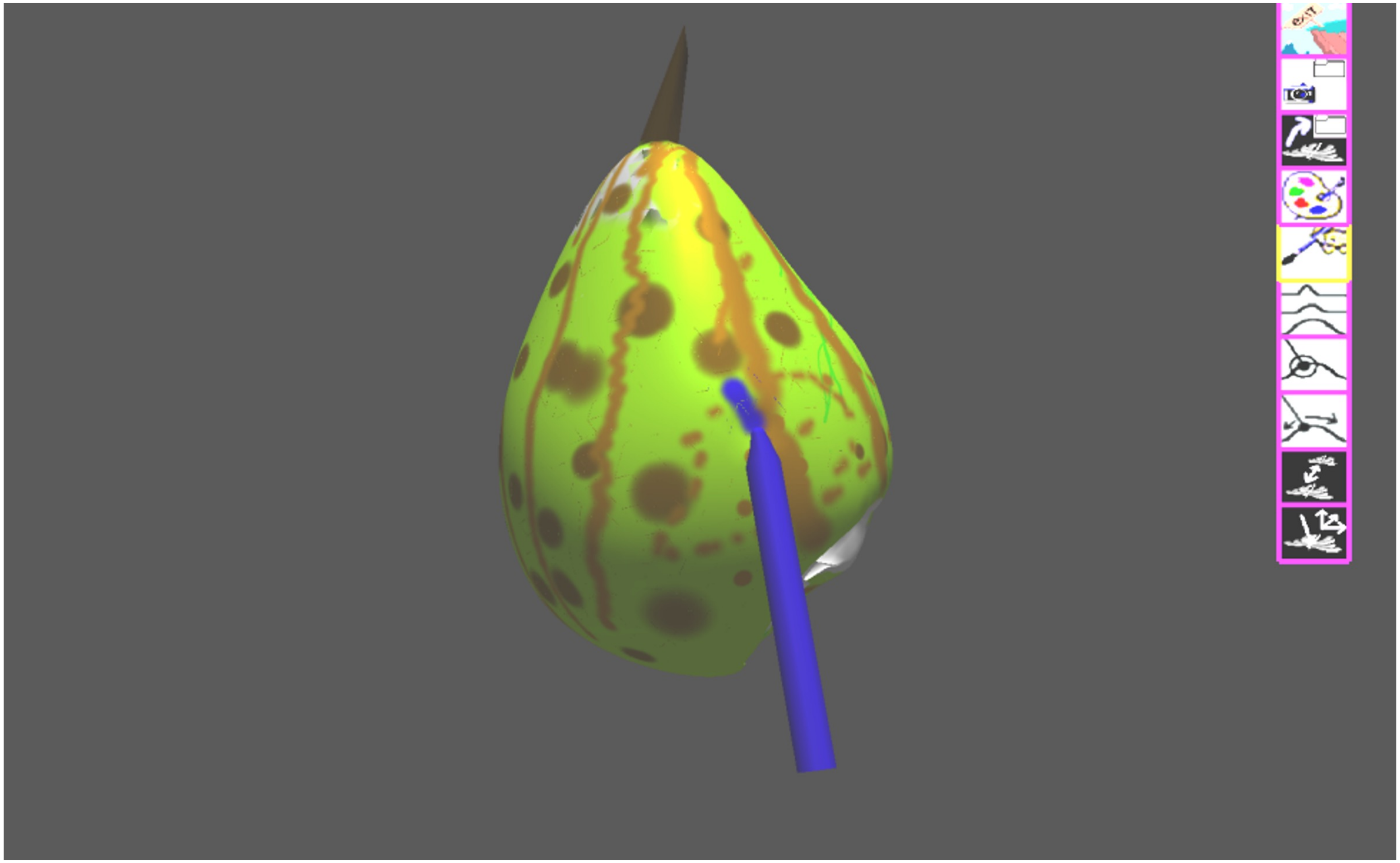
- Direct Haptic Interaction
- Multiresolution Modeling
- 3D Painting on Polygonal Meshes

Gregroy, Ehmann, Lin [VR 2000]

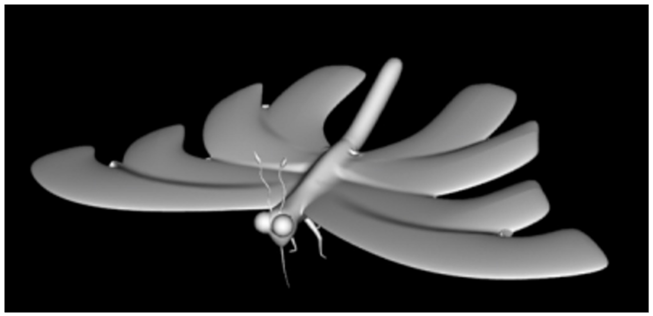
inTouch: System Architecture



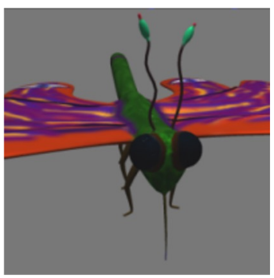
inTouch: User Interface

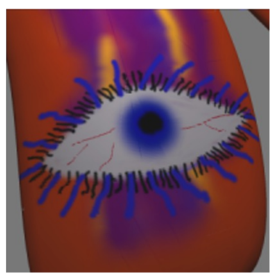


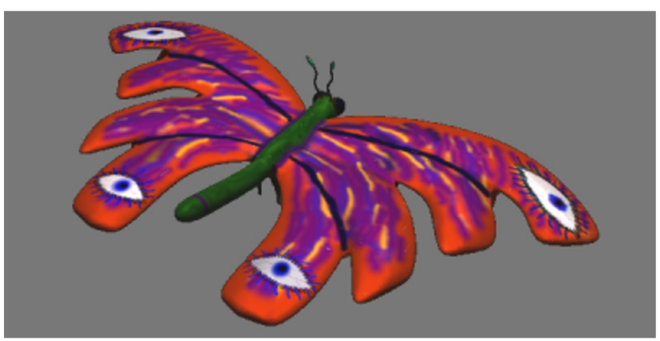
inTouch: Examples



Painted Butterfly (~80k triangles)



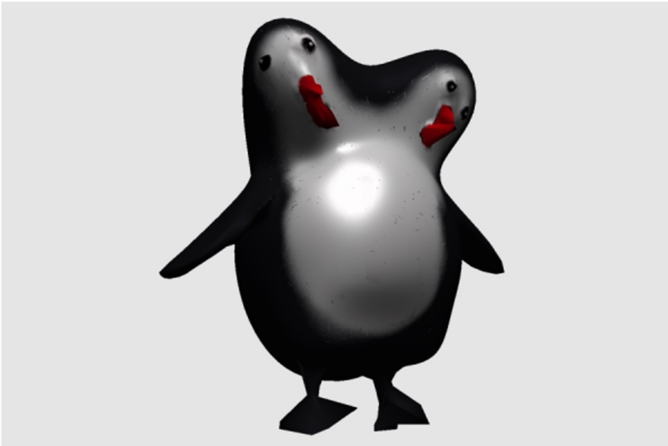




http://www.cs.unc.edu/~geom/inTouch

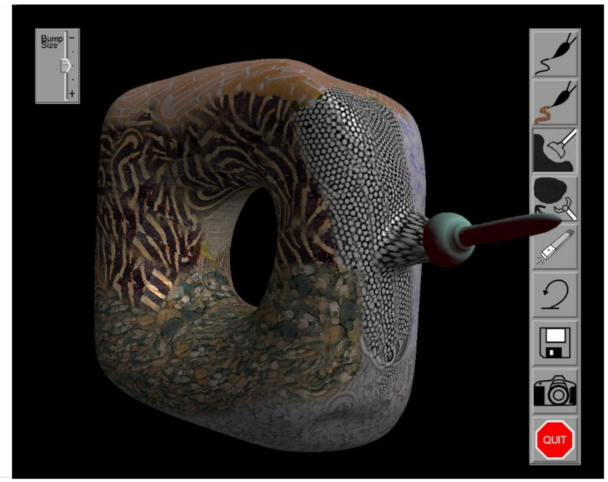
inTouch: Examples




http://www.cs.unc.edu/~geom/inTouch

ArtNova: Touch-Enabled 3D Model Design



- Interactive texture painting
- User-centric viewing
- Realistic force response

http://gamma.cs.unc.edu/ArtNova

Foskey, Otaduy & Lin [VR 2000]

6-DOF Haptic Display Using Localized Computations

- Decompose objects into convex pieces and compute a set of localized pairwise PD's
- Use dual-space expansion to quickly estimate the PD between convex polytopes
- Cluster nearby surface contacts for localized force computation based on PD estimates and predictive methods

http://gamma.cs.unc.edu/6DOFLCC/

Kim, Otaduy, Lin & Manocha [HS 2002]

Collision Detection – SWIFT++

- A fast collision detection library using bounding volume hierarchies of convex hulls
- The overlap test between two convex bounding boxes is performed using fast *Voronoi Marching*
- When collisions occur, needs penetration depth

http://gamma.cs.unc.edu/SWIFT++

Ehmann & Lin [Eurographics 2001]

Minkowski Sum/Difference

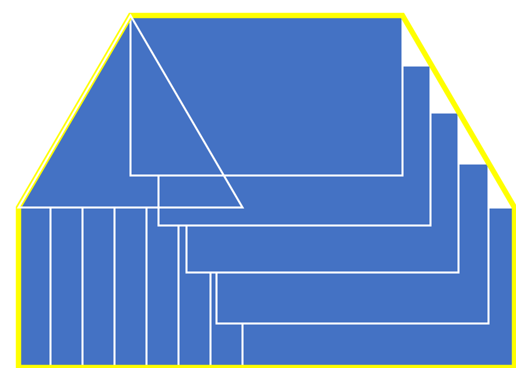
- Minkowski Sum:

  $P+Q = \{p+q \mid p \in P, q \in Q\}$

- Minkowski Difference:

  $P-Q = \{p-q \mid p \in P, q \in Q\}$

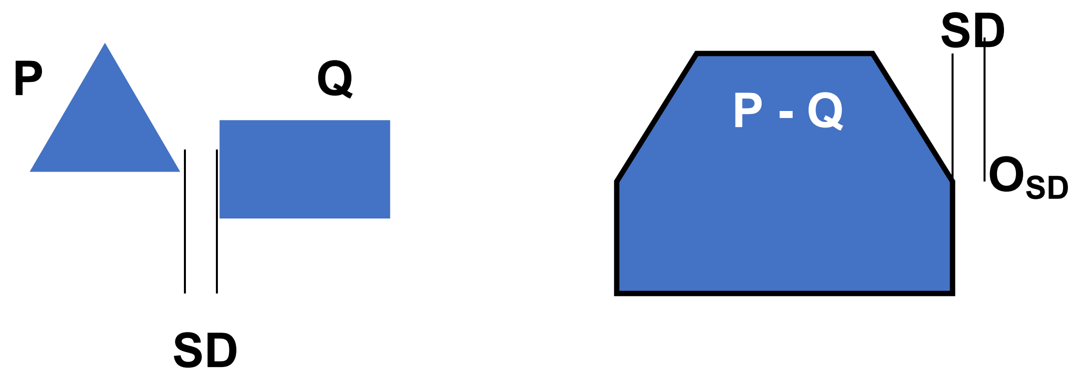
  Negate Q and compute P+(-Q)



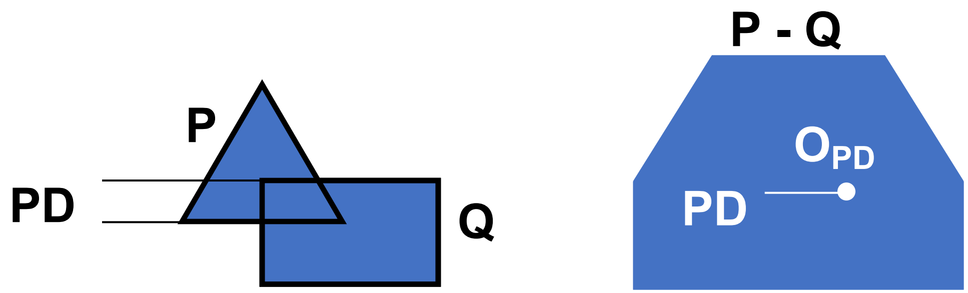
Penetration Depth (PD)

- Minimum translational distance to make *P* and *Q* disjoint over all possible directions.
- Minimum distance from origin $O_{Q\text{-}P}$ to surface $\partial(P\text{-}Q)$ of Minkowski difference.

Separation Distance (SD)

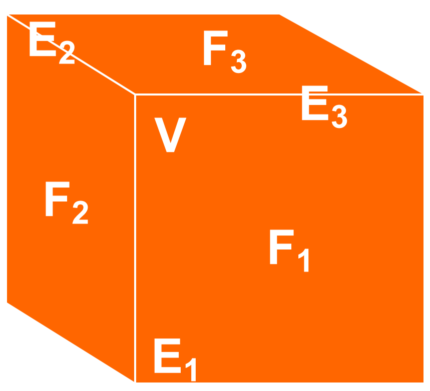


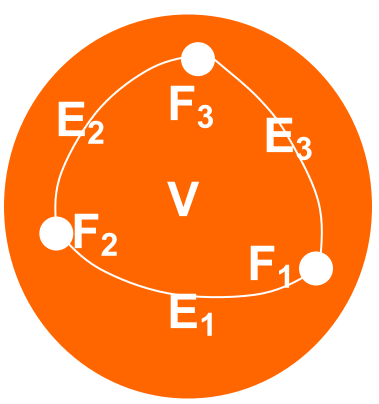
Penetration Depth (PD)



Gauss Map

- Mapping from a feature (V,E,F) in 3D to a unit sphere, according to surface normal
  - Face (F) □ Point in Gauss map
  - Edge (E) □ Great arc in Gauss map
  - Vertex (V) □ Region bounded by great arcs



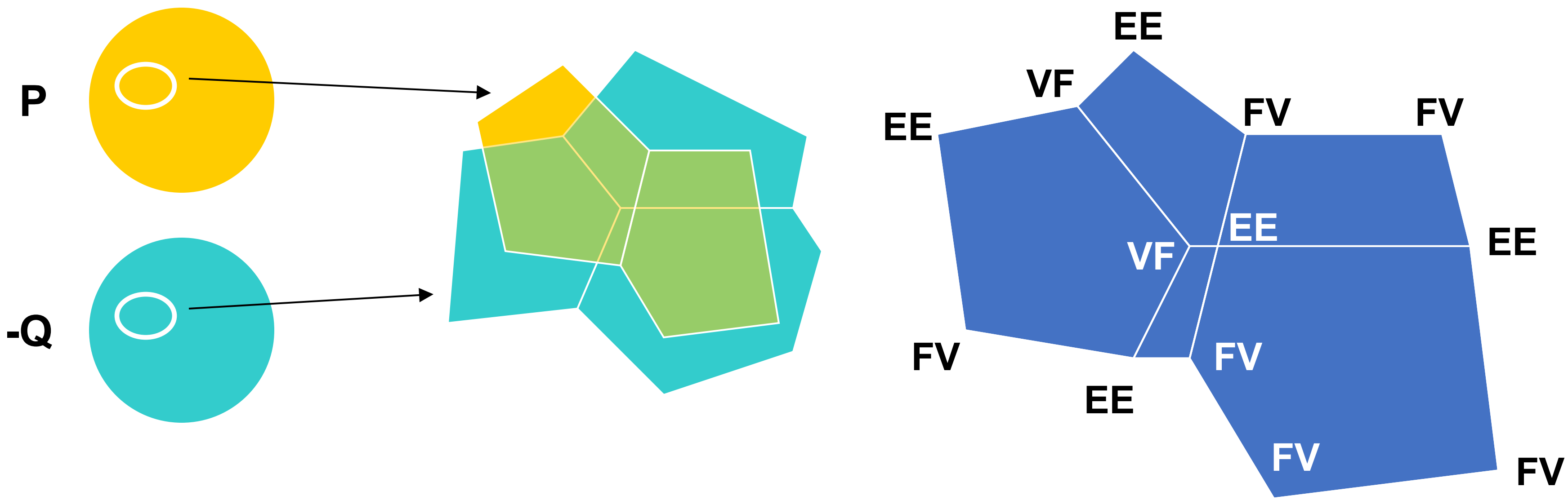


- The Minkowski sum of two convex objects is computed from the overlay of their Gauss maps.

Overview of the Algorithm

- Precomputation
  - Decomposition of an object into convex pieces using the surface decomposition.
- Runtime
  1. Intersection Test
     - Hierarchical pairwise test for intersection.
     - Identify intersecting convex pieces.
  2. PD Computation
     - Walk on the surface of the Minkowski difference of the intersecting convex pieces by minimizing the distance from the origin to the surface.

Minkowski Difference from Gauss Map



Gauss maps of the intersected convex pieces

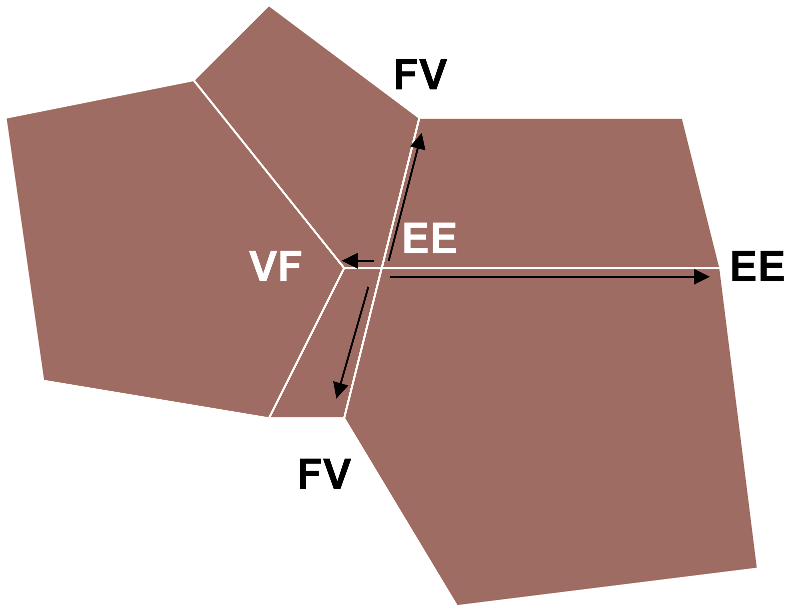
Overlaid and projected Gauss maps

Overlay region that implicitly defines the Minkowski difference

FV, VF and EE pairs correspond to the vertices of the Minkowski difference.

Incremental Search of PD

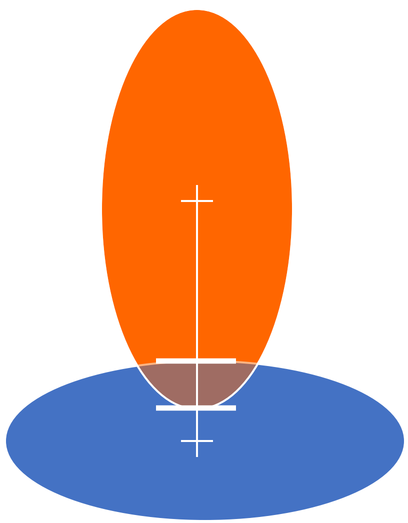
- At a certain vertex in the overlay, check its corresponding PD with that of its neighbors. March toward that vertex that minimizes the PD.
- The actual Minkowski difference is locally computed when needed.



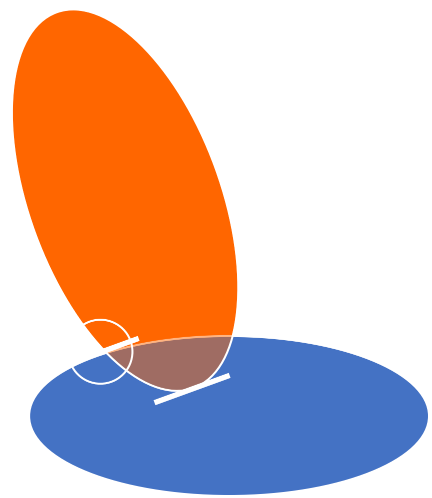
Exploit motion coherence in the initialization

Initialization

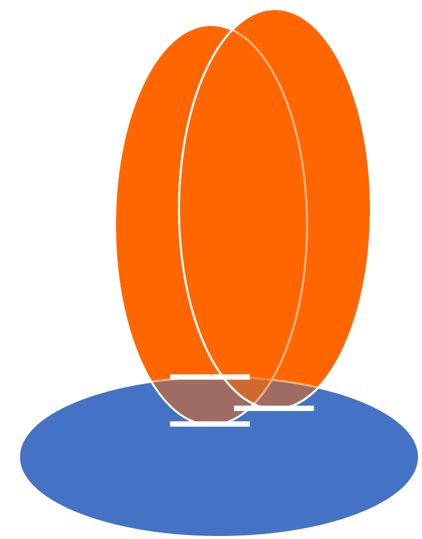
Centroid difference vector



Penetration witness features



Motion coherence



Use the one that minimizes PD

Possible Problems

- Local minima: distance function from the origin to the surface of Minkowski difference can have multiple local minima. Good initialization provides a good result.
- Degeneracies:
  - Coplanar faces. Mapped to the same point in Gauss map. Treat them as a single point, and join the neighbors.
  - Central projection of Gauss map. Solved by local computation at each iteration.

Extension to Non-convex Objects

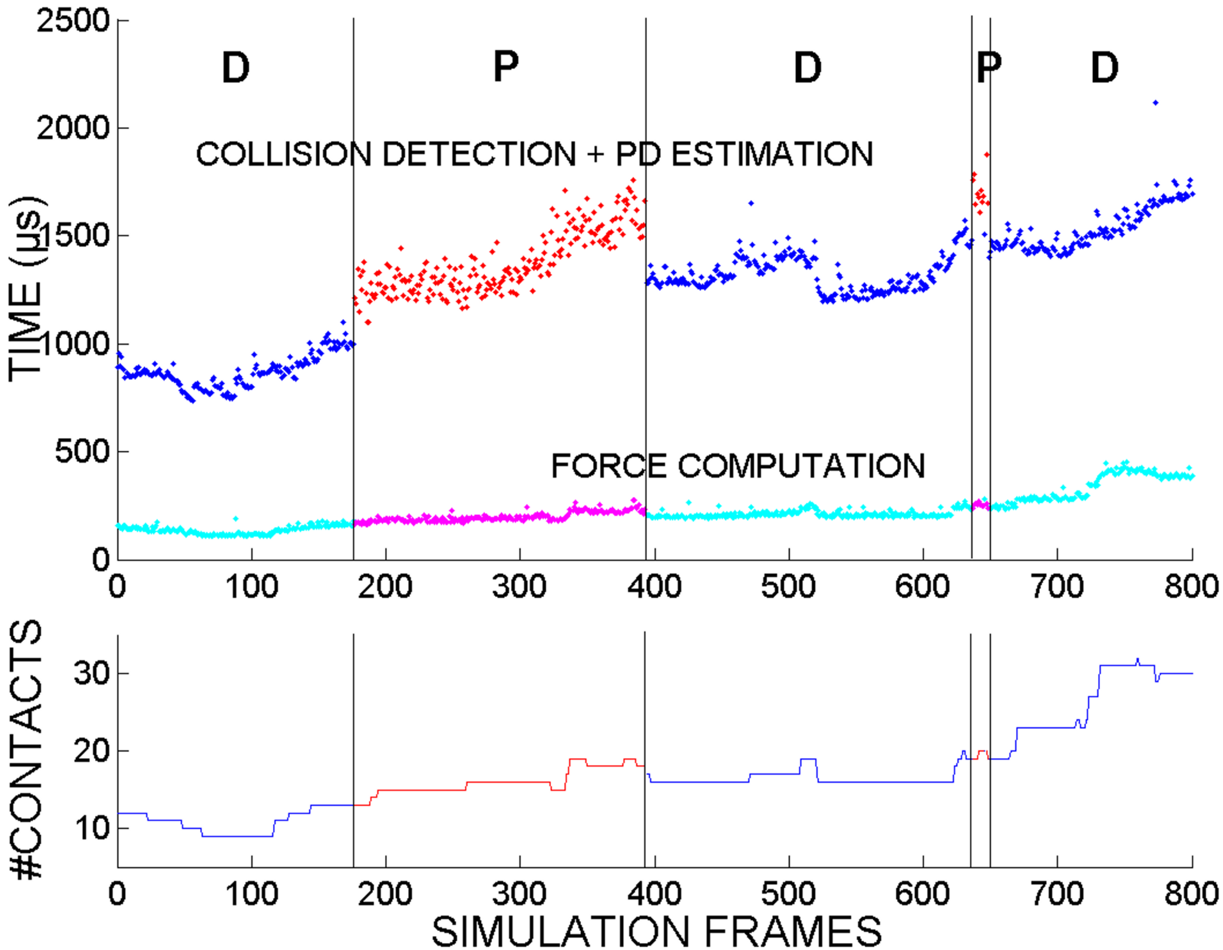
- Pairwise computation of PD.
- Problems originated from surface convex decomposition:
  - Convex pieces completely penetrating the other object
  - PD returns a “virtual feature” that does not exist in the original model
  - We circumvent the problem by traversing to the neighboring features.

Contact Clusters

- Variable number of contacts translates into a variable stiffness.
- Cluster contacts based on the distance between them.
- Compute a new contact (PD, normal, application point) as a weighted average, where weight = PD.
- In practice, the force output is smooth.

Results

Results



D
P
D
P
D
COLLISION DETECTION + PD ESTIMATION
FORCE COMPUTATION
TIME (µs)
2500
2000
1500
1000
500
0
0
100
200
300
400
500
600
700
800
#CONTACTS
30
20
10
0
100
200
300
400
500
600
700
800
SIMULATION FRAMES

The End

For more information, see

http://gamma.cs.unc.edu/interactive

http://gamma.cs.unc.edu/HCollide

http://gamma.cs.unc.edu/inTouch

http://gamma.cs.unc.edu/ArtNova

http://gamma.cs.unc.edu/6DOFLCC/