SIGGRAPH2005

chai3d: An Open-source Toolkit for Haptic Rendering & Applications

Federico Barbagli

Hansen Medical & Stanford University

Goals of this talk

- what is chai3d
- what can chai3d do for you
- what has chai3d done for us
- what can you do for chai3d

Goals of this talk

SIGGRAPH2005

- what is chai3d → main functionalities
- what can chai3d do for you
- what has chai3d done for us
- what can you do for chai3d

What is chai3d?

SIGGRAPH2005

- an API that will let you write your first haptically enabled Virtual Environment, free of charge, in no more than 30 minutes

- a useful platform used by many research labs working on haptics for various purposes

Why chai3d ?

SIGGRAPH2005

- Haptic APIs = Tower of Babel
  - commercial devices come with their own API
  - research lab normally build on top of such APIs

  - lots of energy spent in re-writing same algorithms
  - no portability of code between labs, OSs, devices, compilers

Why chai3d over others?

SIGGRAPH2005

- Other APIs
  - Ghost by Sensable
  - OpenHaptics by Sensable – only Sensable devices
  - ForceDimension low-level API
  - MPB low-level API

  - e-touch by Novint: Not available any longer
  - ReachIn API – commercial, not open source
  - H3D API – only Sensable devices

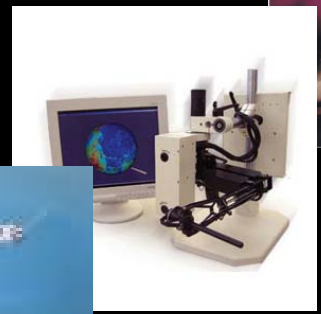
chai3d main features

- chai3d's features that you won't find in other APIs:
  - open source – www.chai3d.org
  - C++ & OpenGL based
  - supports most commercially available desktop devices
  - supports for I/O boards => custom-built devices
  - support for various compilers on various OSs
  - can be used as a
    - low level API to talk to devices
    - high level API with visual and haptic rendering support

Support for HW

SIGGRAPH2005

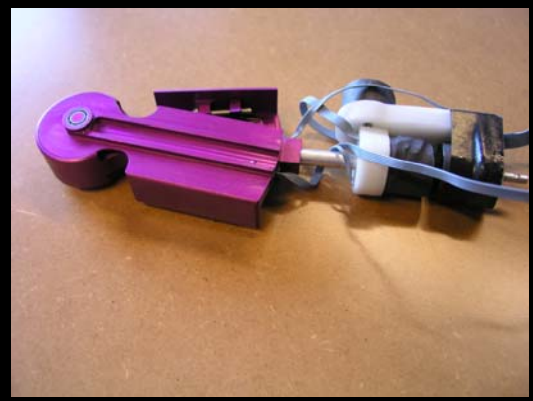
- current support for commercial devices
  - force dimension Omega and Delta
  - Sensable Phantom
  - MPB Freedom 6
- currently being tested
  - FCS Haptic Master
  - thinking about Haption

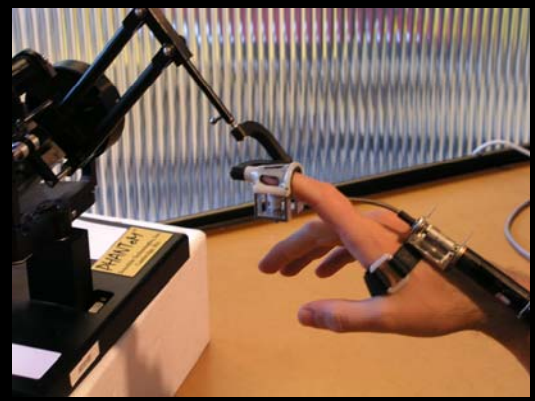


Support for HW

SIGGRAPH2005

- support for I/O boards
  - useful when prototyping your own custom devices





Support for OSs, compilers

- mainly used in windows using
  - MS Visual 6
  - MS Visual .NET
  - Borland C++ Builder
- has been ported, although not yet officially, to
  - Linux
  - QNX
  - Apple OS X
  - RTAI

Who is behind chai3d?

SIGGRAPH2005

- Francois Conti

- Chris Sewell

- Dan Morris

- Fed Barbagli

chai3d Partners

- VRAI Group, EPFL - Ecole Polytechnique Fédérale de Lausanne
- Center for Product Design, ETH Zürich - Eidgenössische Technische Hochschule Zürich
- FB Informatik, Fachhochschule Dortmund
- Imaging Media Research Center, Korea Institute of Science and Technology
- Technion, Israel Institute of Technology
- School of Computer Science, KTH - Royal Institute of Technology in Stockholm
- Department of workscience, Lulea Univ of Technology
- Haptic Interface Research Lab, Purdue University
- Systems and Controls Group, Siena University
- Graphics, Usability, and Visualization Lab, Simon Fraser University
- BioRobotics Lab, Stanford University
- Dexterous Manipulation Lab, Stanford University
- Manipulation Group, Stanford University
- Institut des Sciences et Techniques de l'Ingénieur d'Angers, Université d'Angers
- Discover Lab, University of Ottawa
- Embedded Haptics and Mechatronics Lab, University of Utah

Goals of this talk

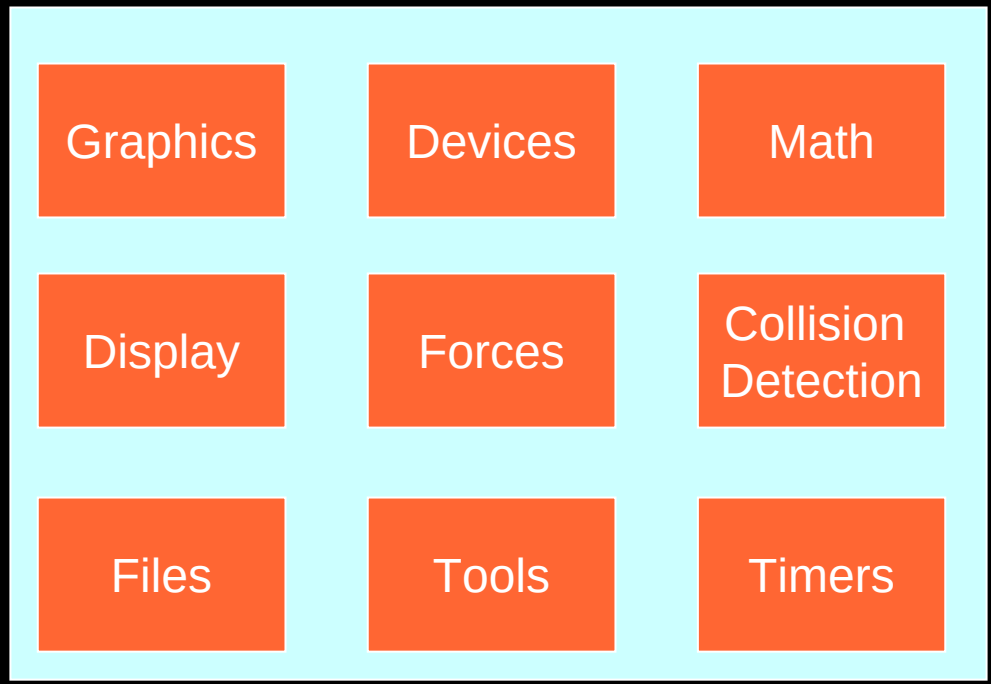
- what is chai3d
- what can chai3d do for you
- what has chai3d done for us
- what can you do for chai3d

main classes
& applications

How is chai3d organized?

SIGGRAPH2005

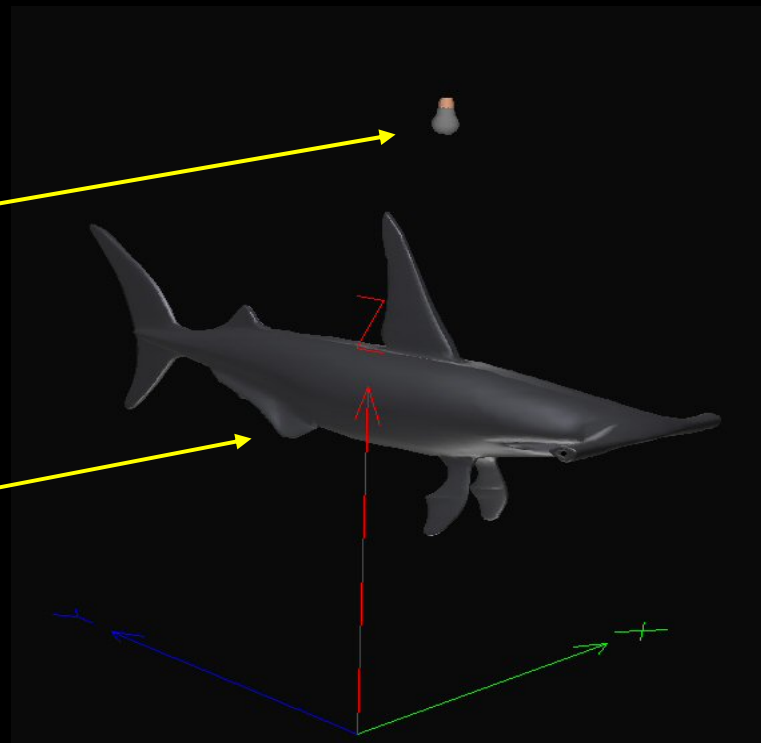
- C++ classes subdivided in 9 groups
- I won’t go in detail on each but
- will try to present each group with meaningful examples



chai3d classes: graphics

SIGGRAPH2005

- Supports multiple scene-graphs
- A standard scene contains
  - a world
  - lights
  - one or more cameras
  - meshes to be visualized
- A viewport is used to render the scene as seen through the camera



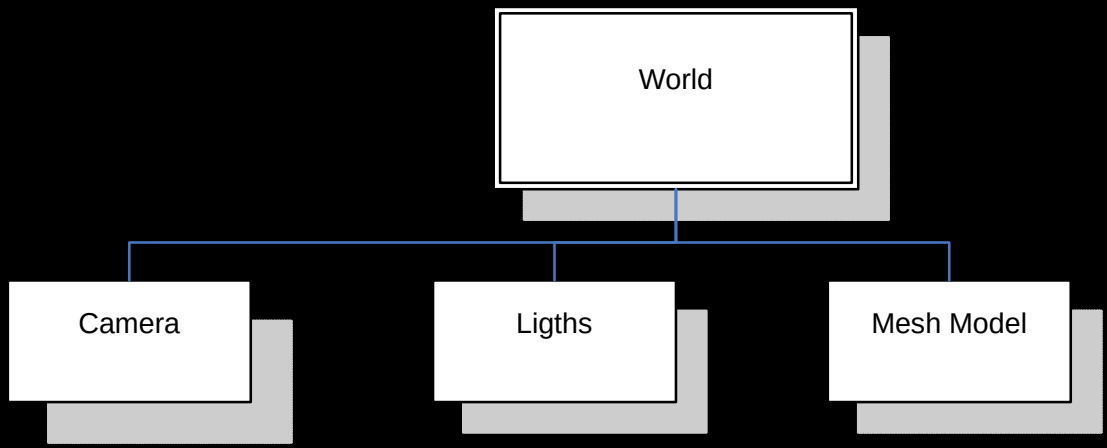
chai3d classes: graphics

- Most objects are inherited from the `cGenericObject` class
- Objects have various graphical features
  - graphic textures
  - vertex colors
  - transparency
- In addition they also have haptic features
  - stiffness
  - frictional properties

Example of graphics class

SIGGRAPH2005

- You can use chai3d as an openGL-like environment for graphics



demo: basic_shapes

chai3d classes: devices and tools

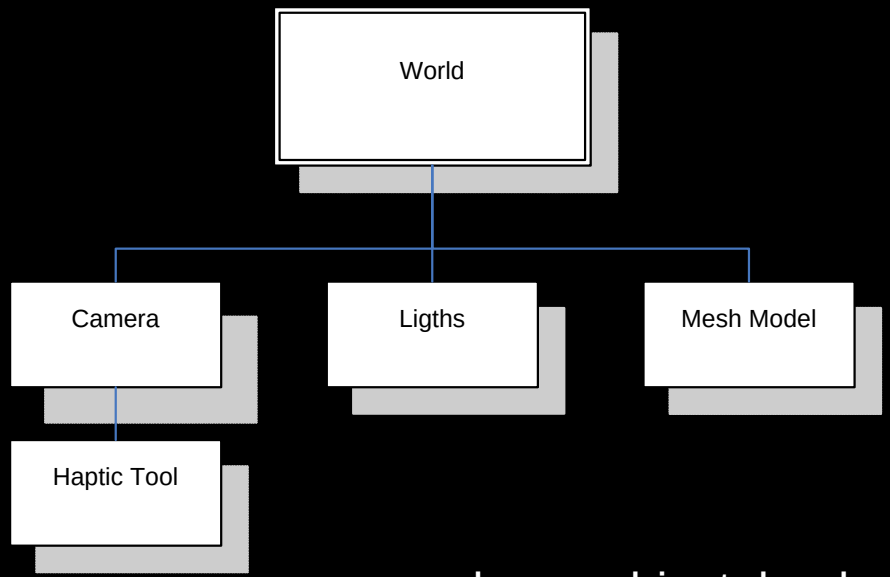
- The `Devices` classes are low-level interfaces to haptic devices
  - start, stop
  - read position/forces
  - command force/positions
- The `Tools` classes are high-level interfaces to haptically-enabled virtual tools
  - they are typically part of the scenegraph
  - they call collision detection and force rendering algorithms to compute forces
  - they communicate to devices associated with them

chai3d classes: devices and tools

- The `cPhantomDevices` class allows you to directly talk a Phantom device
  - useful if you already have your own virtual environment setup and just need to talk to a device
- The `cPhantom3dofPointer` allows you to have a three degrees of freedom pointer to poke objects present in the chai3d scenegraph
  - useful if you are using chai3d to manage virtual environment and force rendering algorithms

Example of using tools: basic shapes

- Can add a 3dof tool to our scenegraph to touch the mesh in the system
- All collision detection and force rendering calls are handled transparently

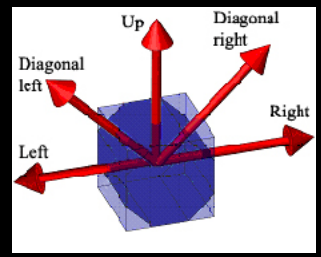


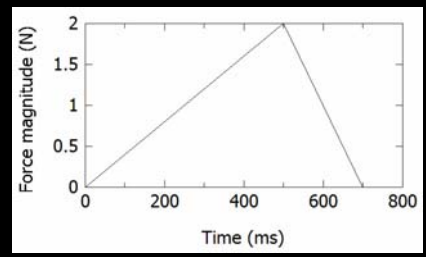
demo: object_loader

Example of using devices

SIGGRAPH2005

- Experiment on the discrimination threshold of force directions under the three display conditions
  - haptics alone (H)
  - haptics plus congruent vision (HVcong)
  - haptics plus incongruent vision (HVincong)
- Collaboration with Hong Tan, Charles Spence, Cristy Ho, Ken Salisbury
- chai3d was used to
  - display forces through a Phantom Premium 1.5
  - create visual cues to the user
  - the two were disconnected, i.e. the user was not touching the 3D objects being displayed on the screen





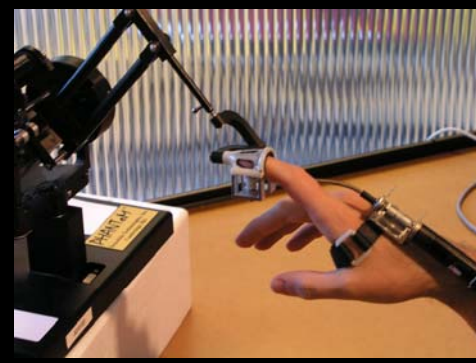
Creating new devices-tools

- New haptic interface prototypes can be included in chai3d easily following 2 steps:
  - create a new `device` class to talk to read/write from the device
  - create a new `tool` class to connect the device to a virtual environment tool

Example of new device-tool

SIGGRAPH2005

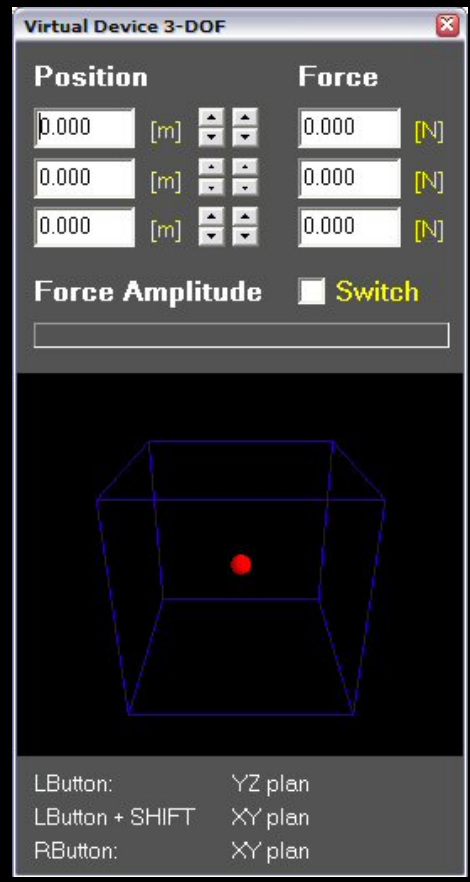
- Rotational Friction Display (RFD) built by Will Provancher (University of Utah)
- Device is position controlled through a Sensoray626 board
  - i.e. interface is read/command angular position
- Tool is a 4 DOF which
  - commands 3DOF forces to a Phantom and 1DOF positions to the RFD
- Used to simulate slip when grasping virtual objects



The Virtual Device

SIGGRAPH2005

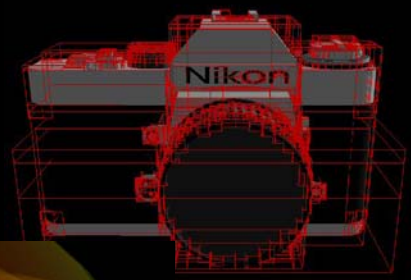
- Some drawbacks of most haptic devices:
  - expensive
  - not easy to carry around
  - very hard to debug
  - easy to break
- invented for CS277 Experimental Haptics class at Stanford by Francois Conti
- it allows to
  - move a pointer in 3D with a mouse
  - visualize forces returned from the chai3d application

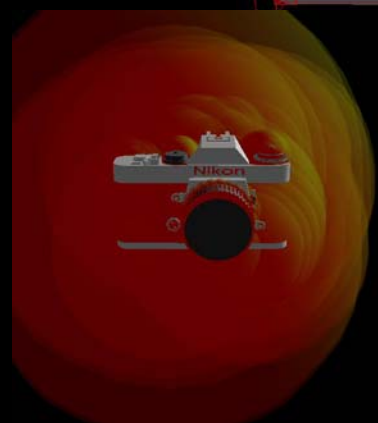


demo: object_loader

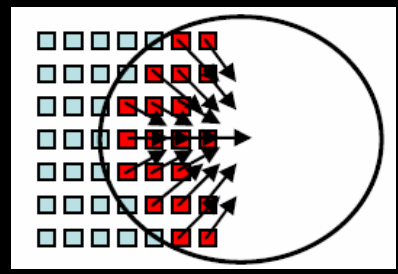
chai3d classes: forces and collision detection, files

- Force rendering algorithms supported (only 3dof):
  - god-object algorithm with friction (Zilles ‘95)
  - potential fields
  - we’re working on implicit function proxy (Salisbury ’97)
- Collision detection algorithms supported:
  - line to mesh using AABB trees
  - line to mesh using Spheres hierarchies
- File types supported for meshes
  - OBJ – Alias Wavefront file format
  - 3Ds – 3d Studio Max file format

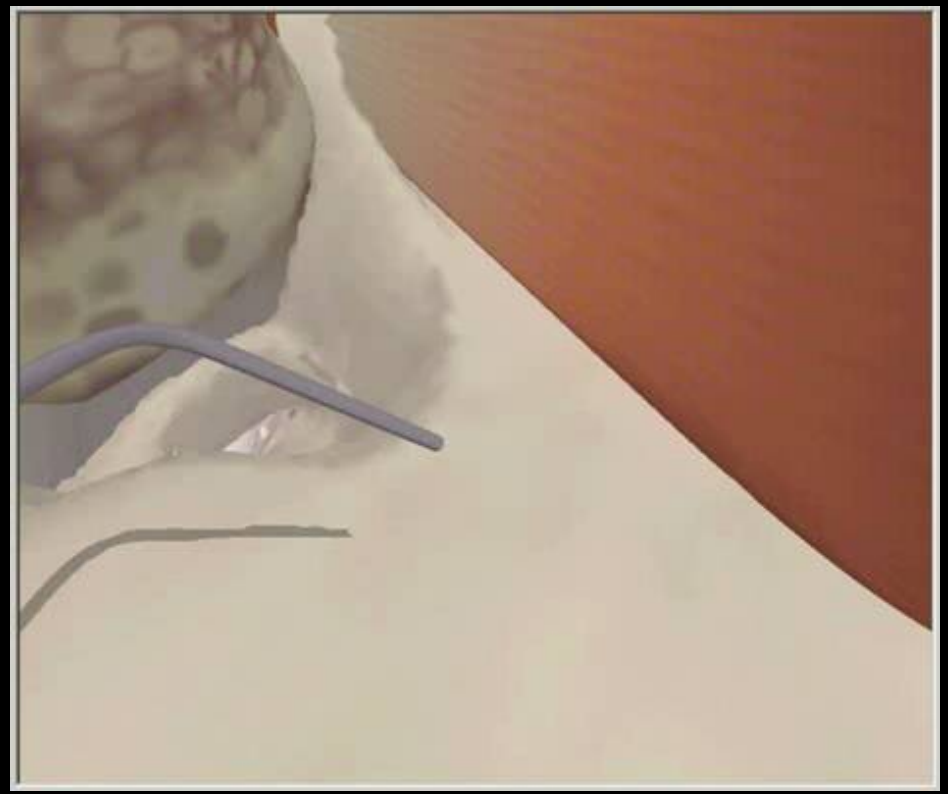




Example of new force rendering algorithms



- Temporal bone simulation (mastoidectomy, acoustic neuroma resection, cochlear implantation)
- Work by D. Morris, C. Sewell, N. Blevins @ Stanford
- Bone is represented using voxels
- Interaction forces are computed by
  - intersecting a voxel representation of the drill and the bone
  - drill point found inside a bone voxel generates contribution to overall force toward the tool center



chai3d classes: timers

- Haptic rendering of stiff surfaces becomes unstable for low servo rates
- Typical rates are 1KHz
- This can be hard under Windows and thus
  - most haptic devices have some way to emulate Real Time behavior
  - chai3d approach: use Windows Multimedia Timers, accurate up to 1KHz

A typical chai3d application

- **Initialization**
  - instantiate world, camera, light, objects, tool
  - add everything to world
  - instantiate viewport and timer
- **Haptic callback (~1KHz)**
  - read tool position, compute forces, apply them
- **Visual callback (~30Hz)**
  - render viewport

A typical chai3d application


```
world = new cWorld();

camera = new cCamera(world);
camera->set(cVector3d(2, 0, 0), cVector3d(0, 0 , 0), cVector3d(0, 0, 1));
world->addChild(camera);

light = new cLight(world);
light->setEnabled(true);
camera->addChild(light);

object = new cMesh(world);
world->addChild(object);
object->loadFromFile("a3dcamra.obj");
object->scale(0.075);
object->createAABBCollisionDetector(true,true);
object->computeGlobalPositions();

tool = new cMeta3dofPointer(world, 0);
camera->addChild(tool);
tool->setPos(-2.0, 0.0, 0.0);
tool->setRadius(0.02);
tool->initialize();

viewport = new cViewport(Panel1->Handle, camera, true);

timer = new cPrecisionTimer();
timer->set(0,HapticLoop,NULL);
```

A typical chai3d application


```
viewport->render();
```

A typical chai3d application


```
tool->updatePose();
tool->computeForces();
tool->applyForces();
```

demo: object_loader

Goals of this talk

- what is chai3d
- what can chai3d do for you
- what has chai3d done for us
- what can you do for chai3d

how to contribute

Learning chai3d

- download chai3d from

  www.chai3d.org
- build libraries using your favorite compiler
- add the `chai3d/bin` folder to your path
- look at the examples
- look at Doxygen documentation

Contributing to chai3d

SIGGRAPH2005

- chai3d is written by academics for anyone who needs it…
- it is not perfect nor complete…
- you can help by
  - reporting bugs
  - contributing with new code/algorithms
- look at coding conventions online

The end!