CS 277 - Experimental Haptics
Lecture 5

CHAI3D
Overview

- CHAI3D framework organization
- Haptic interfaces
- Coordinates
- Building a world
- Scene graph
- Objects
- Tools
- Force algorithms
- Examples
Organization

- executable files
  DLL libraries

- external libraries used inside CHAI3D

- modules (ODE, GEL)

- application templates

- project and solution files

- examples with source code

- additional libraries used by examples

- CHAI3D source code

- utilities
Examples
Examples
Examples
Haptic Device Handler

application

haptic device handler
(cHapticDeviceHandler)
Haptic Devices

```c
class cGenericHapticDevice

int open()
int close()

int initialize()

int getPosition(cVector3d& a_position)
int getLinearVelocity(cVector3d& a_linearVelocity)
int getRotation(cMatrix3d& a_rotation)

int setForce(cVector3d& a_force)
int getUserSwitch(int a_switchIndex, bool& a_status)

cHapticDeviceInfo getSpecifications()
```
Haptic Devices

// create a haptic device handler
handler = new cHapticDeviceHandler();

// get access to the first available haptic device
cGenericHapticDevice* hapticDevice;
handler->getDevice(hapticDevice, 0);

// retrieve information about the current haptic device
cHapticDeviceInfo info;
if (hapticDevice)
{
    info = hapticDevice->getSpecifications();
}

(...)

cHapticDeviceInfo

string m_manufacturerName;
double m_maxForce;
double m_maxForceStiffness;
double m_workspaceRadius;
bool m_sensedPosition;
bool m_sensedRotation;
bool m_actuatedPosition;
bool m_actuatedRotation;
(...
Coordinate System
Virtual World

world

light

camera

objects
Reference frames
Scene

- Defining independent objects in the world
- Defining relationships between these objects
Expressing the vertices of the object (car) in reference with the world coordinate system.
Moving the object would require computing a new position for each vertex.

\[ \{W\} : \text{World Reference Frame} \]
Moving the object would require computing a new position for each vertex.
Reference Frames

Defining a local reference frame for each independent object.

{W} : World Reference Frame

{A} : Reference Frame of Object A

World Origin
Reference Frames

Defining a local reference frame for each independent object.

{W} : World Reference Frame

{A} : Reference Frame of Object A
Reference Frames

World Origin

{D3}

{D4}

{D1}

{D2}
Scene Graph

```
world (root node)

light

camera

object

light

object

mesh

shape

tool
```
Shapes
Shapes
(cShapeSphere, cShapeCylinder, cShapeTorus, cShapeBox)
Example

haptic rate: 21441 [Hz]
Materials
(cMaterial)

GRAPHIC PROPERTIES:
cColorf m_ambient;  Ambient color.
cColorf m_diffuse;  Diffuse color.
cColorf m_specular;  Specular color.
cColorf m_emission;  Emissive color.
GLuint m_shininess;  Shininess

HAPTIC PROPERTIES:
double m_viscosity;  Viscosity constant.
double m_stiffness;  Stiffness constant.
double m_static_friction;  Static friction constant.
double m_dynamic_friction;  Dynamic friction constant.
double m_vibrationFrequency;  Frequency of vibrations
double m_vibrationAmplitude;  Amplitude of vibrations.
double m_magnetMaxForce;  Maximum force applied by magnet effect.
double m_stickSlipForceMax;  Force threshold for stick and slip effect.
double m_stickSlipStiffness;  Spring stiffness of stick slip model.
Materials
(cMaterial)

GRAPHIC PROPERTIES:
cColorf m_ambient;
Ambient color.
cColorf m_diffuse;
Diffuse color.
cColorf m_specular;
Specular color.
cColorf m_emission;
Emissive color.
GLuint m_shininess;
Shininess

Ambient + Diffuse + Specular = Phong Reflection
Colors
Colors
(cColorf)

// set material color by name
object->m_material->setBlueCornflower();

// set material color by values (R-G-B)
object->m_material->setColorf(0.2, 0.1, 0.1);

// set material color by components (R-G-B)
object->m_material->m_ambient->set(0.2, 0.2, 0.2);
object->m_material->m_diffuse->set(0.5, 0.5, 0.5);
object->m_material->m_specular->set(1.0, 1.0, 1.0);

// define color
ccColorf color;
color.setColorf(0.2, 0.1, 0.1);
Creating Vertices

```
// create mesh
CMesh* mesh = new CMesh();

// add mesh to world
world->addChild(mesh);

// create 3 vertices
unsigned int vertexIndex0 = mesh->m_vertices->newVertex();
unsigned int vertexIndex1 = mesh->m_vertices->newVertex();
unsigned int vertexIndex2 = mesh->m_vertices->newVertex();
```
Vertex Properties

// define vertex position
cVector3d position(2.0, 3.0, 4.0);
mesh->m_vertices->setLocalPos(vertexIndex0, position);

// define vertex normal
cVector3d normal(1.0, 0.0, 0.0);
mesh->m_vertices->setNormal(vertexIndex0, normal);

// define texture coordinate
mesh->m_vertices->setTexCoord(vertexIndex0, 0.2, 0.3);

// define vertex color
cColorf color;
color.setBlueCadet();
mesh->m_vertices->setColor(vertexIndex0, color);
Creating Triangles

```cpp
// create triangle
mesh->m_triangles->newTriangle(vertexIndex0, vertexIndex1, vertexIndex2);
```
Textures

- mesh object (cMesh)
- texture map (cTexture2D)
- mesh object with texture properties defined
- single textured triangle and its 3 vertices
cMultiMesh
Haptic Effects
(cGenericEffect)

HAPTIC PROPERTIES: (cMaterial)
double m_viscosity;
double m_stiffness;
double m_static_friction;
double m_dynamic_friction;
double m_vibrationFrequency;
double m_vibrationAmplitude;
double m_magnetMaxForce;
double m_stickSlipForceMax;
double m_stickSlipStiffness;

Viscosity constant.
Stiffness constant.
Static friction constant.
Dynamic friction constant.
Frequency of vibrations
Amplitude of vibrations.
Maximum force applied by magnet effect.
Force threshold for stick and slip effect.
Spring stiffness of stick slip model.
Haptic Effects
(cGenericEffect)

// create a haptic vibration effect
object->createEffectVibration();

// create a haptic surface effect
object->createEffectSurface();
Camera
Camera

near clipping plane

far clipping plane
Camera

orthographic

perspective
Camera
(cCamera)

// create a camera and insert it into the virtual world
camera = new cCamera(world);
world->addChild(camera);

// position and orient the camera
camera->set( cVector3d (0.5, 0.0, 0.0), // camera position (eye)
    cVector3d (0.0, 0.0, 0.0), // look at position (target)
    cVector3d (0.0, 0.0, 1.0)); // direction of the (up) vector

// set the near and far clipping planes of the camera
camera->setClippingPlanes(0.01, 10.0);
Stereo Camera
Stereo Camera
(cCamera)

// set stereo mode
camera->setStereoMode(stereoMode);

// set stereo eye separation and focal length (applies only if stereo is enabled)
camera->setStereoEyeSeparation(0.005);
camera->setStereoFocalLength(0.5);

// set vertical mirrored display mode
camera->setMirrorVertical(mirroredDisplay);
Example
Widgets

haptic device: omega.3
position [m]: -0.067, 0.035, 0.020

haptic rate: 998 [Hz]
Example
Traversing a Scene Graph

object

tool
Traversing a Scene Graph
Traversing a Scene Graph

1. Read the position of the haptic device
2. For each object (potential field) in the environment, compute the position of the tool in relation to the local reference frame
3. Compute the reaction force in the local frame
4. Convert the reaction force in the world frame
5. Send the force to the haptic device