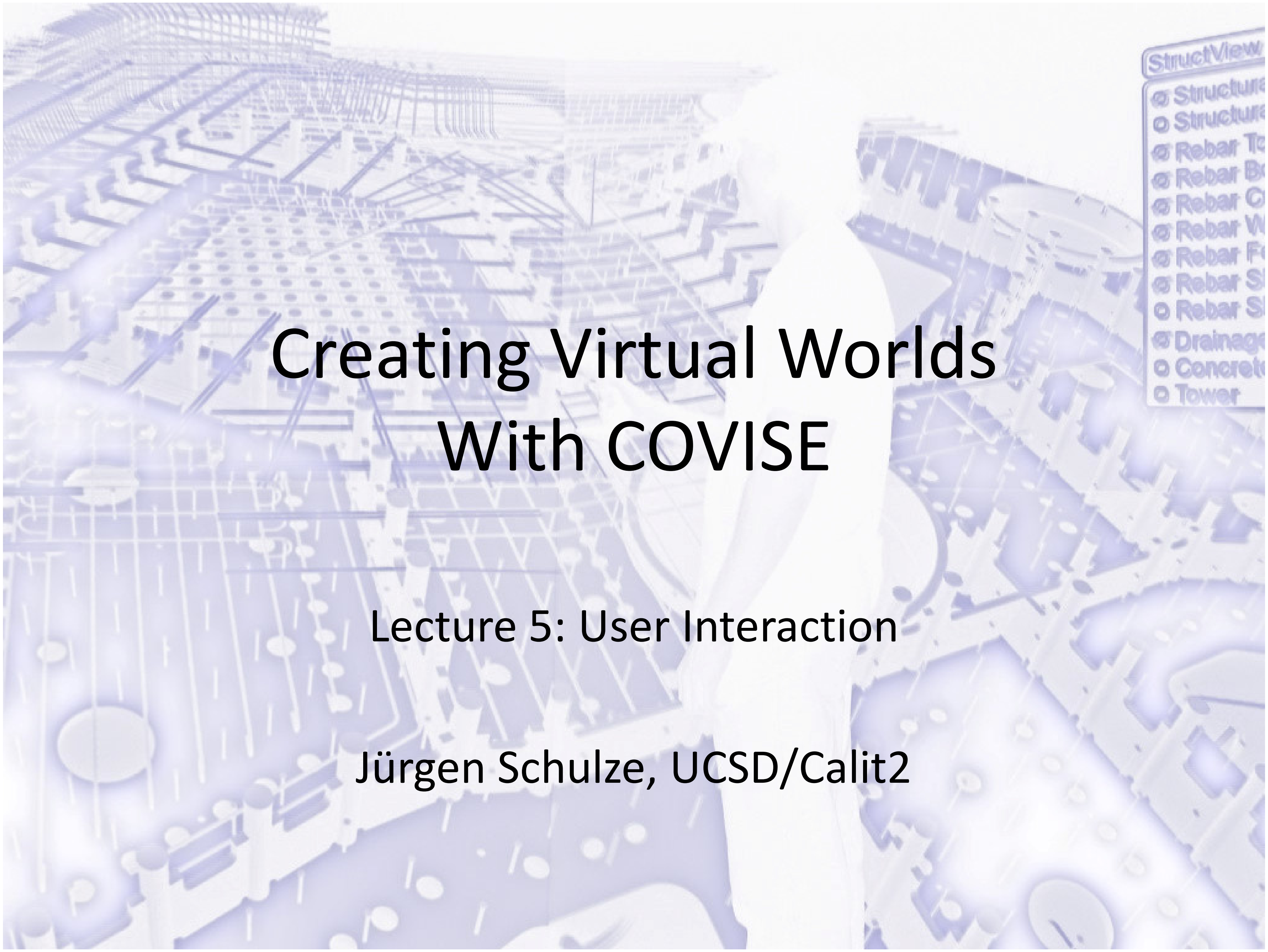


Creating Virtual Worlds With COVISE

Lecture 5: User Interaction

Jürgen Schulze, UCSD/Calit2

Course Overview

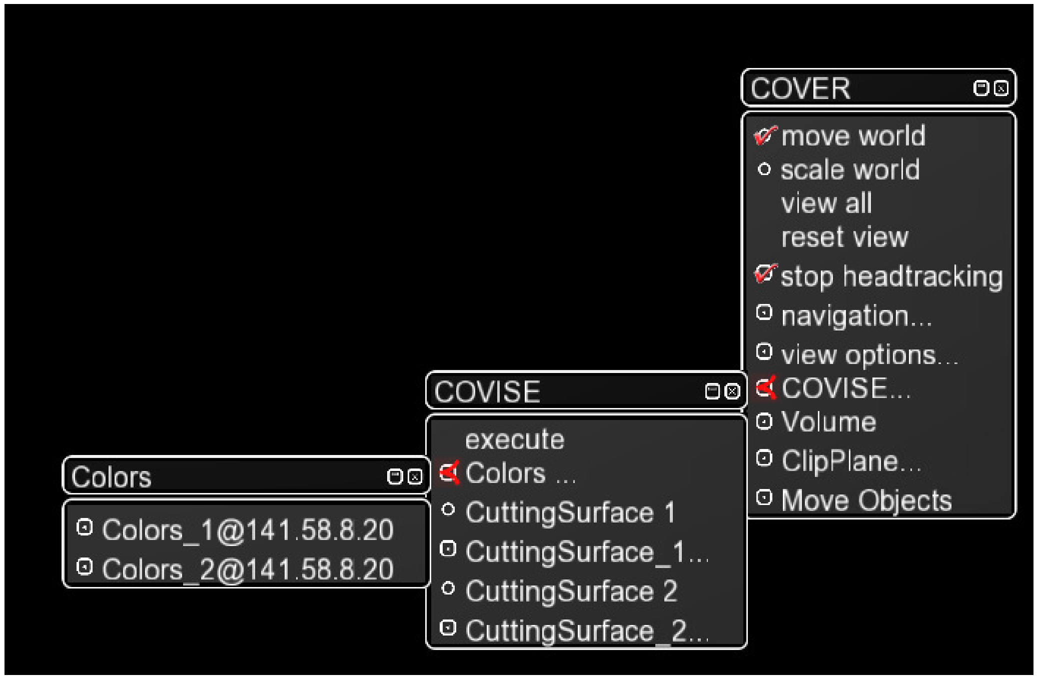
- Lecture 1: COVISE Overview
- Lecture 2: Map Editor and Modules
- Lecture 3: OpenCOVER and Plugins
- Lecture 4: OpenSceneGraph
- Lecture 5: User Interaction
- Lecture 6: Collaborative Applications

Overview

- OpenVRUI menu entries
- Accessing tracker data
- Moving an object
- Button handling
- OSGCaveUI

OpenVRUI Menu Entries

- Create the following menu entries:
  1. SubMenuItem „**Colors …**“ in menu „**COVISE**“
  2. RowMenu „**Colors**“
  3. SubMenuItems in „**Colors**“



OpenVRUI Menu Entries

- In plugin‘s init() routine:

```
coMenu *coviseMenu = NULL;
VRMenu *menu = VRPinboard::instance()->namedMenu("COVISE");
if(menu)
{
  coviseMenu = menu->getCoMenu();

  // Create button entry:
   colorSubMenuItem = new coSubMenuItem("Colors ...");
   colorRowMenu = new coRowMenu("Colors");
   colorSubMenuItem ->setMenu(colorRowMenu);

   coviseMenu->add(colorSubMenuItem);
}
```

OpenVRUI Menu Entries

- Entries in Colors RowMenu:

```
newSubMenuItem = new coSubMenuItem(moduleName);
  …
colorRowMenu->add(newSubMenuItem)
```

- Other menu entries:

```
Slider = new coSliderMenuItem(Name,Min,Max,Value);
Checkbox = new coCheckboxMenuItem(Name,state);
Button  = new coButtonMenuItem(Name);

colorRowMenu->add(Slider);
colorRowMenu->add(Checkbox );
colorRowMenu->add(Button);

Slider->setMenuListener(this);
Checkbox->setMenuListener(this);
Button->setMenuListener(this);
```

OpenVRUI Menu Entries

- Menu Events:

```
class SamplePlugin: public coMenuListener
{       …
        void menuEvent(coMenuItem*);
        void menuReleaseEvent(coMenuItem*);
        …
}

void SamplePlugin::menuEvent(coMenuItem* menuItem)
{     …
      if(menuItem==Slider) …
      …
}
```

Accessing Tracker Data

- Get pointer (=wand) position (pos1) and a point 1000 millimeters from it (pos2) along the pointer line:

```
osg::Vec3 pointerPos1Wld = cover->getPointerMat().getTrans();
osg::Vec3 pointerPos2Wld = osg::Vec3(0.0, 1000.0, 0.0);
pointerPos2Wld = pointerPos2Wld * cover->getPointerMat();
```

- Get head position in world coordinates:

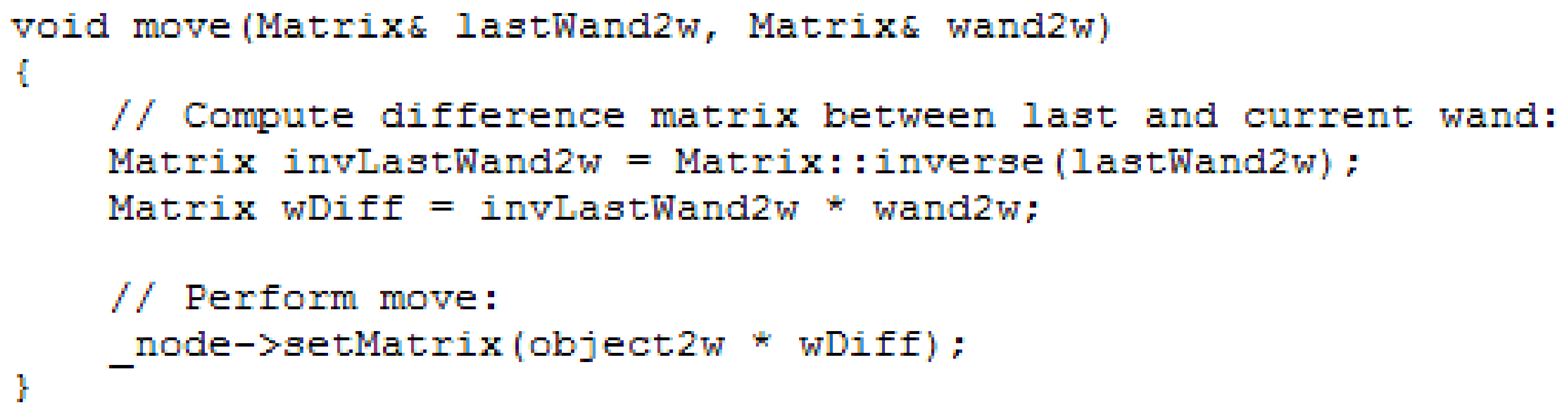
```
Vec3 viewerPosWld = cover->getViewerMat().getTrans();
```

- Head position in object coordinates:

```
Vec3 viewerPosWld = cover->getViewerMat().getTrans();
Vec3 viewerPosObj = viewerPosWld * cover->getInvBaseMat();
```

Moving an Object With the Pointer

- object2w:
  Object's transformation matrix in world coordinates
- lastWand2w and wand2w:
  Wand matrices from previous and current frames
  (from cover->getPointer())



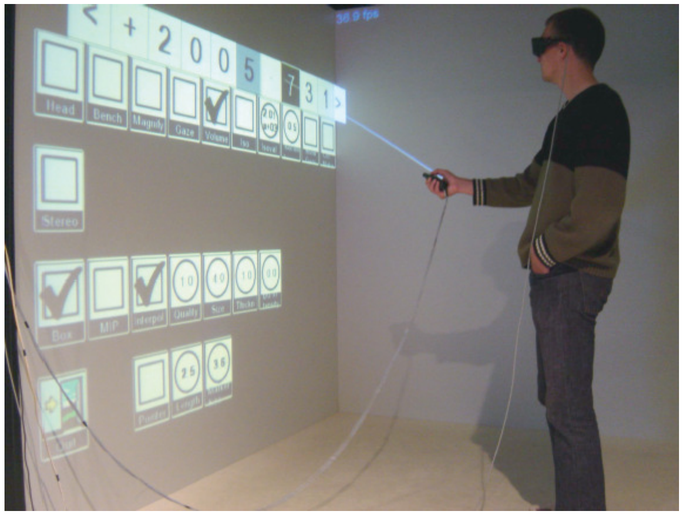
```
void move(Matrix& lastWand2w, Matrix& wand2w)
{
    // Compute difference matrix between last and current wand:
    Matrix invLastWand2w = Matrix::inverse(lastWand2w);
    Matrix wDiff = invLastWand2w * wand2w;

    // Perform move:
    _node->setMatrix(object2w * wDiff);
}
```

Button Handling

- class `TrackerButtonInteraction`:
  Supports interaction with wand, including registering a button while pressed so that it does not accidentally trigger functions in other plugins or the COVER menu.
- Include:
  `#include <OpenVRUI/coTrackerButtonInteraction.h>`
- In constructor: create interaction for button A, which is the left wand button:
  `interaction = new coTrackerButtonInteraction(coInteraction::ButtonA,"MoveObject",coInteraction::Menu);`
- In destructor:
  `delete interaction;`
- The code for handling the interaction needs to go in the preFrame() function. To register your interaction and thus disable button A interaction in all other plugins call the following function.
  `if(!interaction->registered) { coInteractionManager::the()->registerInteraction(interaction); }`
- To do something just once, after the interaction has just started:
  `if(interaction->wasStarted()) { }`
- To do something every frame while the interaction is running:
  `if(interaction->isRunning()) { }`
- To do something once at the end of the interaction:
  `if(interaction->wasStopped()) { }`
- To unregister the interaction and free button A for other plugins:
  `if(interaction->registered && (interaction->getState()!=coInteraction::Active)) { coInteractionManager::the()->unregisterInteraction(interaction); }`

OSGCaveUI

- Source files at:
  `covise/src/renderer/OpenCOVER/osgcaveui/`
- PickBox:
  logical structure which allows interaction with sparse data sets
- Calculator:
  Pocket calculator-like utility
- FloatOMeter:
  Input of floating point numbers



More Information

- IVL Wiki:

  `http://ivl.calit2.net/wiki/index.php/COVISE_and_OpenCOVER_support`