

11

Visualization Systems

Ronald Peikert

SciVis 2007 - Visualization Systems

11-1

Modular visualization environments

Many popular visualization software are designed as so-called modular visualization environments (MVEs):

- data flow architecture
- visual programming of applications
- API (e.g. C++) for adding new modules

Examples of products:

- commercial: AVS, Covise
- open source: OpenDX, VTK/Paraview, SCIRun

Ronald Peikert

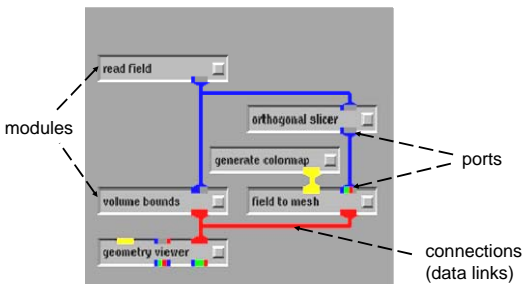
SciVis 2007 - Visualization Systems

11-2

Modular visualization environments

Components of an MVE

(1) Visual programming editor



Ronald Peikert

SciVis 2007 - Visualization Systems

11-3

Modular visualization environments

(2) Modules:

- Categories, typically:
 - Input (reading, generating data)
 - Filters (mapping to the same data type)
 - Mappers (mapping to a different data type)
 - Output (3D graphics, image, or file)
- Module libraries:
 - ordered by category, author, etc.
 - users' community contributed modules
- Implementation:
 - separate processes per module or single process

Ronald Peikert

SciVis 2007 - Visualization Systems

11-4

Modular visualization environments

- Actions (callback functions) on:
 - instantiation (loading the module into the application editor)
 - change of input data (at input port)
 - change of parameter (via widget)
 - deletion of the module
 - idle state
- Ports:
 - specification of data types
 - type checking
 - required / optional input ports
 - multiple connections allowed / disallowed
- Connections:
 - to modules on same / different host
 - shared memory or TCP/IP

Ronald Peikert

SciVis 2007 - Visualization Systems

11-5

Modular visualization environments

- Visual programs (aka networks, applications, ...)
 - directed graphs (usually acyclic)
 - graphical and/or text-based programming
 - scheduler, controlling execution order of modules (sequential or parallel execution)
 - control flow mostly follows data flow (exceptions: picking operation)

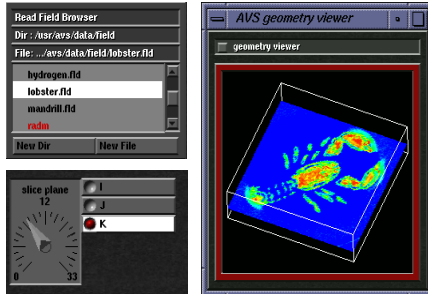
Ronald Peikert

SciVis 2007 - Visualization Systems

11-6

Modular visualization environments

(3) UI widgets (parameters, status, viewers, etc.)



Ronald Peikert

SciVis 2007 - Visualization Systems

11-7

AVS

Application Visualization System

- Advanced Visual Systems Inc. (originally by Ardent Computer)
- local / remote modules
- Unix / Linux
- products: AVS 5, AVS/Express
- 3D viewer: OpenGL based, supports stereo
- parallel execution of modules possible

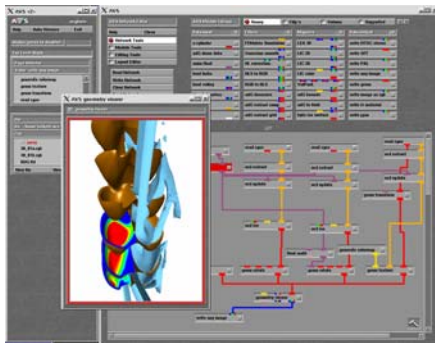
Ronald Peikert

SciVis 2007 - Visualization Systems

11-8

AVS

AVS screenshot



Ronald Peikert

SciVis 2007 - Visualization Systems

11-9

Covise

Collaborative Visualization Environment

- by HLRS, University of Stuttgart
- distributed sessions
 - viewing, modifying of application by remote users
 - master / slave mode, token-based
- 3D viewers
 - OpenInventor-based
 - Performer-based virtual environment renderer. Supports stereo, head tracking, 3D input devices
- arrays
 - modules operating on <datatype> can also handle <array of datatype>

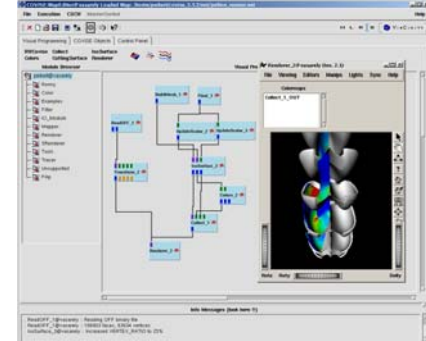
Ronald Peikert

SciVis 2007 - Visualization Systems

11-10

Covise

Covise screenshot



Ronald Peikert

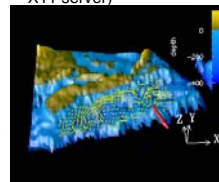
SciVis 2007 - Visualization Systems

11-11

OpenDX

Open Data eXplorer

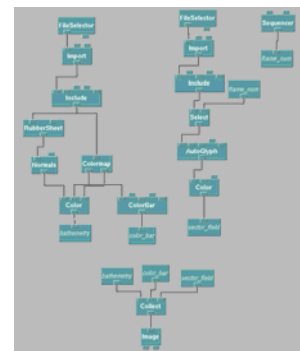
- formerly IBM Data Explorer
- Linux and Windows
- Open source (but windows version requires commercial X11 server)



Ronald Peikert

SciVis 2007 - Visualization Systems

11-12



OpenDX

Modules can be created in 3 fashions:

- built into the DX executable
- runtime loadable
- stand-alone program (using DX libraries)

Parallelism:

- parallel tasks within a module

Visual programming editor:

- multi-page
- modules and control-panel widgets (widgets are not part of modules)

Ronald Peikert

SciVis 2007 - Visualization Systems

11-13

OpenDX

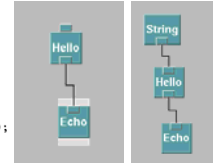
Module programming

- description file hello.mdf

```
MODULE Hello;
CATEGORY Greetings
DESCRIPTION Prefixes "hello" to the input string
INPUT value; string; "world"; input string
OUTPUT greeting; string; prefixed string
```

- source file hello.c :

```
01 #include <dx/dx.h>
02
03
04 Error m_Hello(Object *in, Object *out)
05 {
06     char message[30], *greeting;
07
08     if (!in[0])
09         sprintf(message, "hello world");
10     else {
11         DXExtractString(in[0], &greeting);
12         sprintf(message, "%s %s", "hello", greeting);
13     }
14
15     out[0] = DXNewString(message);
16     return OK;
17 }
```



Ronald Peikert

SciVis 2007 - Visualization Systems

11-14

VTK

Visualization Toolkit

- by Kitware, Inc. (Schroeder, Martin, Lorensen)
- open source
- scripting (Tcl, Python, ...)
- application programming in C++ or Java
- Paraview: extension for graphical application programming
- ITK (insight toolkit): extension for medical vis. (incl. segmentation, registration)

Ronald Peikert

SciVis 2007 - Visualization Systems

11-15

VTK

Platforms:

- Unix incl. Linux, Windows

3D Viewers:

- OpenGL, SGI GL, HP Starbase, Sun XGL, VolumePRO

Focus:

- isosurfaces, decimation
- direct volume rendering
- implicit functions
- tensor fields

Ronald Peikert

SciVis 2007 - Visualization Systems

11-16

VTK

Module programming:

- C++

Application programming:

- C++, Tcl / Tk, (Python, Java)

Animation

- Loops in C++ or script languages

Ronald Peikert

SciVis 2007 - Visualization Systems

11-17

VTK

Example: Render a cube

```
// C++ version
#include "vtk.h"
main() {
    vtkRenderer* ren
    = vtkRenderer::New();
    vtkRenderWindow* win
    = vtkRenderWindow::New();
    vtkCubeSource* src
    = vtkCubeSource::New();
    vtkPolyDataMapper* mpr
    = vtkPolyDataMapper::New();
    vtkActor* act
    = vtkActor::New();
    mpr->SetInput(
    src->GetOutput());
    cube->SetMapper(mpr);
    ren->AddActor(act);
    win->AddRenderer(ren);
    win->Render();
}

# Tcl version
catch {load vtk Tcl}
vtkRenderer ren
vtkRenderWindow win
vtkCubeSource src
vtkPolyDataMapper mpr
vtkActor act
mpr SetInput [ \
scr GetOutput ]
act SetMapper mpr
ren AddActor act
win AddRenderer ren
win Render
```

Ronald Peikert

SciVis 2007 - Visualization Systems

11-18

VTK

VTK screenshot

- 3D viewer
- control panel
- interactor (Tcl/Tk widgets)

Ronald PeikertSciVis 2007 - Visualization Systems11-19

VTK

ParaView screenshot

Ronald PeikertSciVis 2007 - Visualization Systems11-20

SCIRun

Scientific Computing and Imaging Institute (SCI), Univ. of Utah.

- SCIRun/BioPSE - Problem Solving Environment for BioMedical Applications
- PowerApps
 - BioImage - Application for displaying and analyzing biomedical images
 - BioTensor- Application for displaying and analyzing tensor data
 - BioFEM - Application for calculating electric fields in a FEM mesh
- Seg3D - Volume processing and segmentation software.

Ronald PeikertSciVis 2007 - Visualization Systems11-21

SCIRun

SCIRun screenshot

Ronald PeikertSciVis 2007 - Visualization Systems11-22

Other visualization systems

Other (non MVE) visualization systems:

Commercial:

- Amira (Zuse Institute Berlin)
- SimVis (VRVis Research Vienna)
- EnSight (CEI, originally by Cray)
- TecPlot (TecPlot Inc.)

Open Source:

- VisIt (Lawrence Livermore National Laboratory)

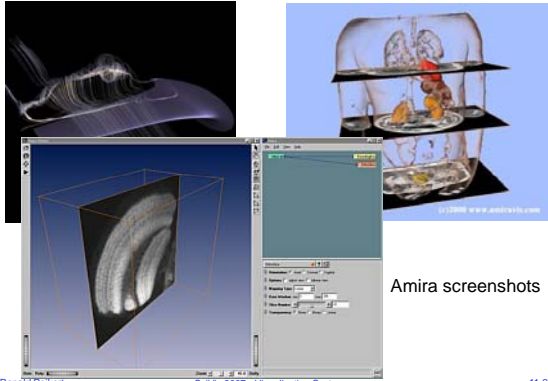
Ronald PeikertSciVis 2007 - Visualization Systems11-23

Amira

- by Zuse Institute Berlin
- Focus:
 - medical visualization, segmentation, registration
 - flow visualization, topology
- Windows, Linux, Unix
- based on OpenGL / OpenInventor
- Tcl scripting
- not (yet) extendable

Ronald PeikertSciVis 2007 - Visualization Systems11-24

Amira



Amira screenshots

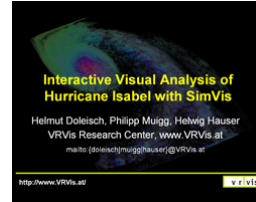
Ronald Peikert

SciVis 2007 - Visualization Systems

11-25

SimVis

- by VRVis Research, Vienna
- combines SciVis and InfoVis
- linked views (scatter plots etc.)
- focus+context, brushing with soft boundaries



Ronald Peikert

SciVis 2007 - Visualization Systems

11-26