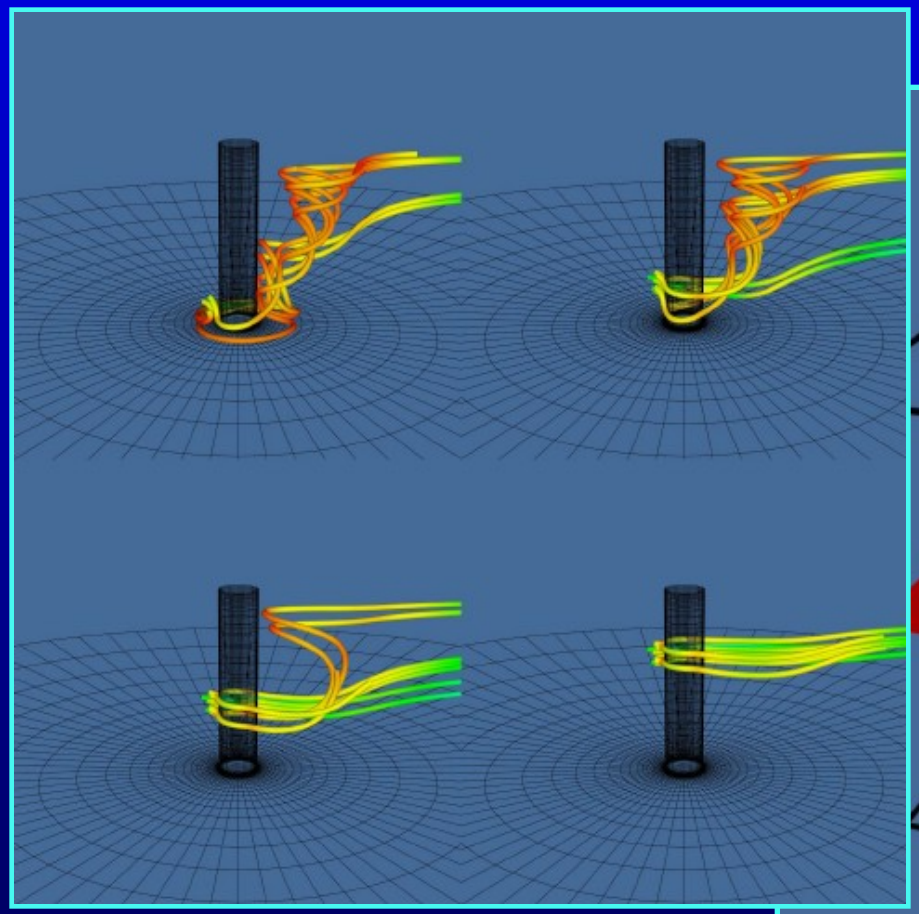
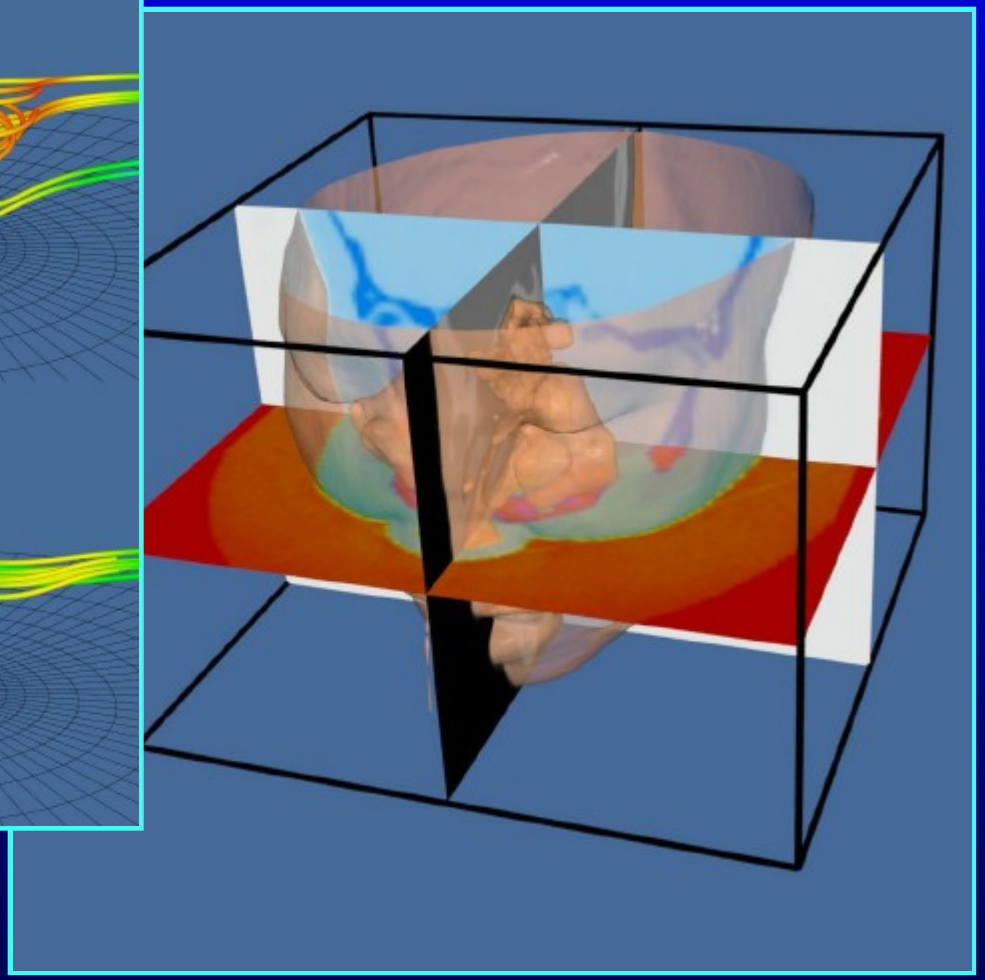
14 Years of Object-Oriented Visualization

Bill Lorensen
General Electric
Corporate Research and Development
lorensen@crd.ge.com

Object-Oriented Visualization





Outline

- Beginnings
- Object-Oriented Visualization
- Motivation
- Systems versus Toolkits

Outline

- A System: LYMB
- A Toolkit: VTK
- Lessons Learned
- 1998 and Beyond

Computing in 1984

- Vax 11/780
  - 1 mip
  - time shared
  - alphanumeric terminals

Graphics Hardware in 1984

- Tektronix Storage Tubes
- Plotters
- Framebuffers
  - Lexidata 3400 (640x512x12)
  - Raster Technologies (640x512x24)
- Hardcopy
  - 16 mm Cameras

Graphics Software in 1984

- Tektronix Plot10
  - Chart/Line drawing
- Evans and Sutherland
  - Proprietary, Vector refresh
- Movie.BYU
  - Shaded, batch mode

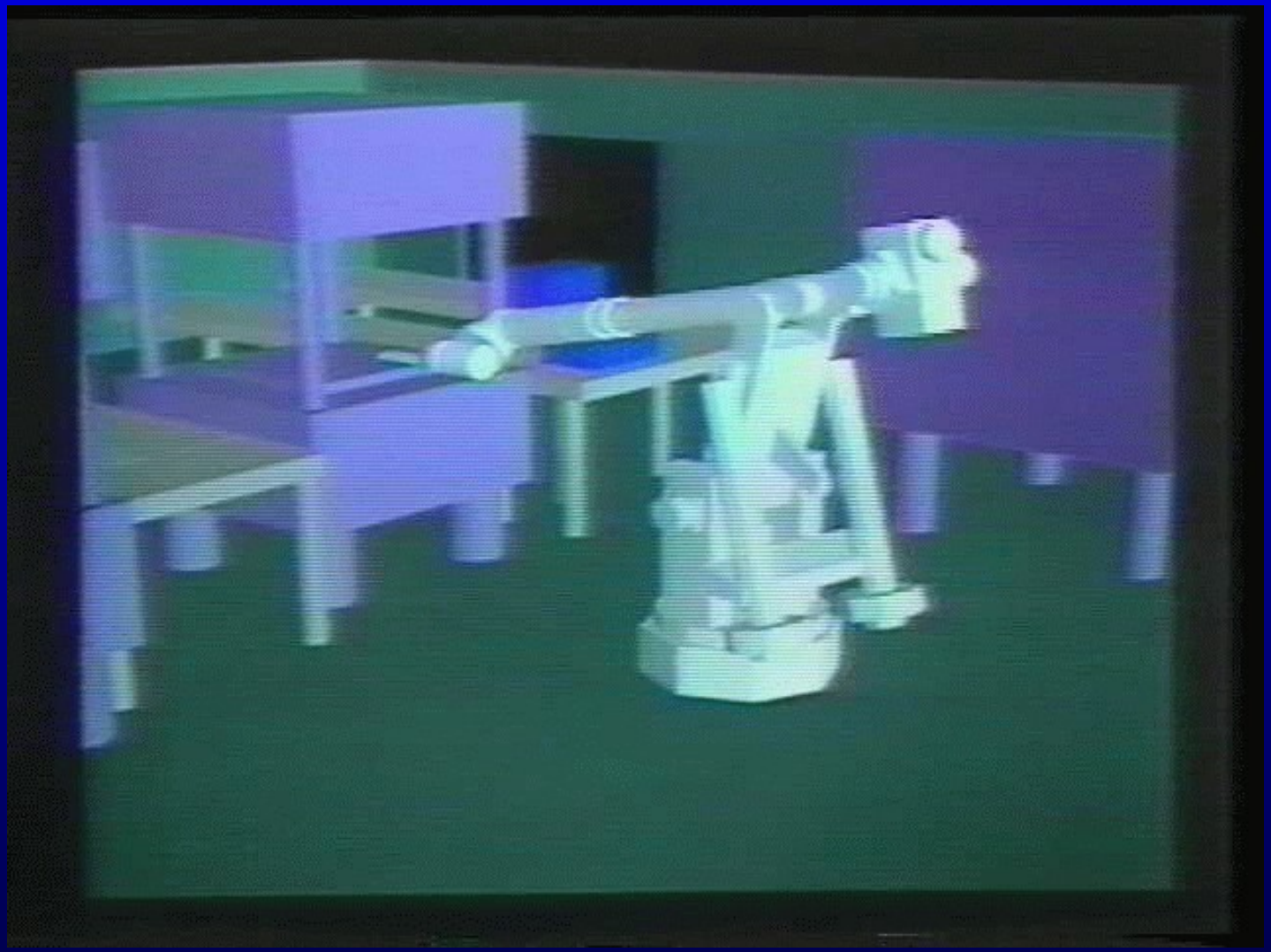
No such animal as a graphics API!

Software Methods in 1984

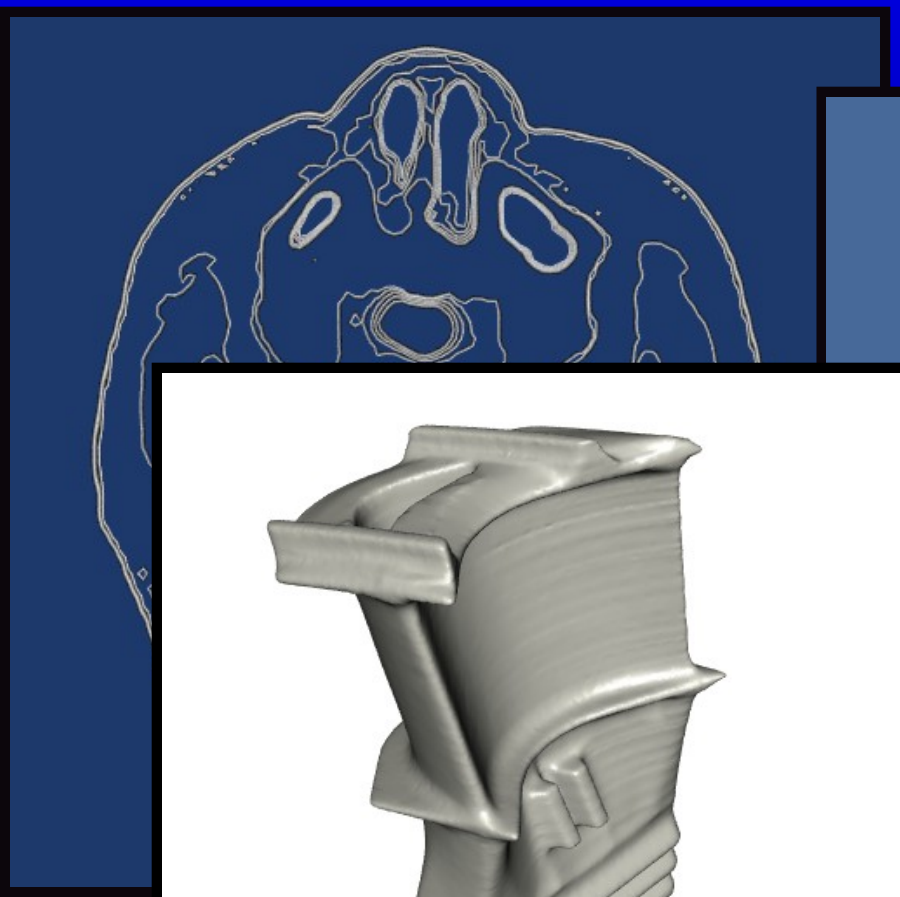
- Peak of structured programming
- Beginnings of OO
  - Simula 67
  - Smalltalk
  - Lisp Machine
  - Objective-C

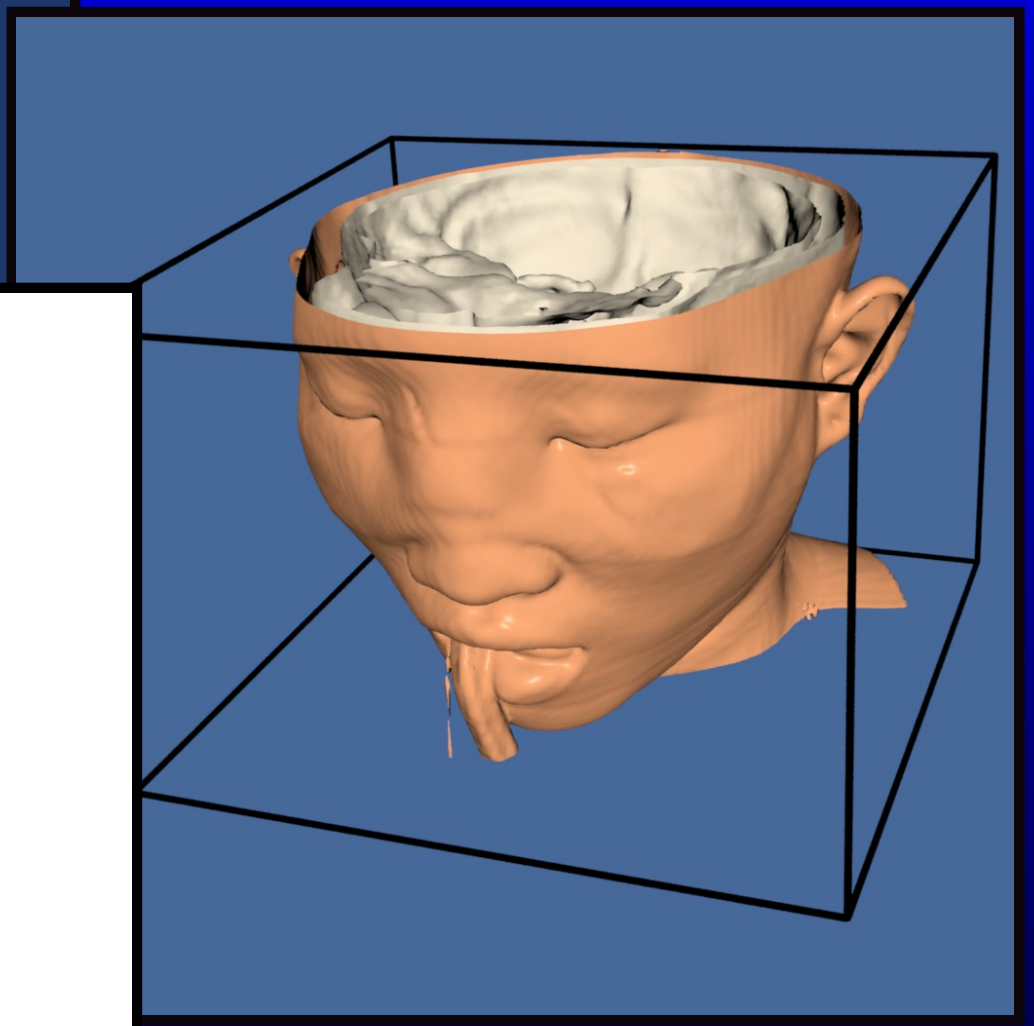
Few college courses and texts

Robot Simulation 1984

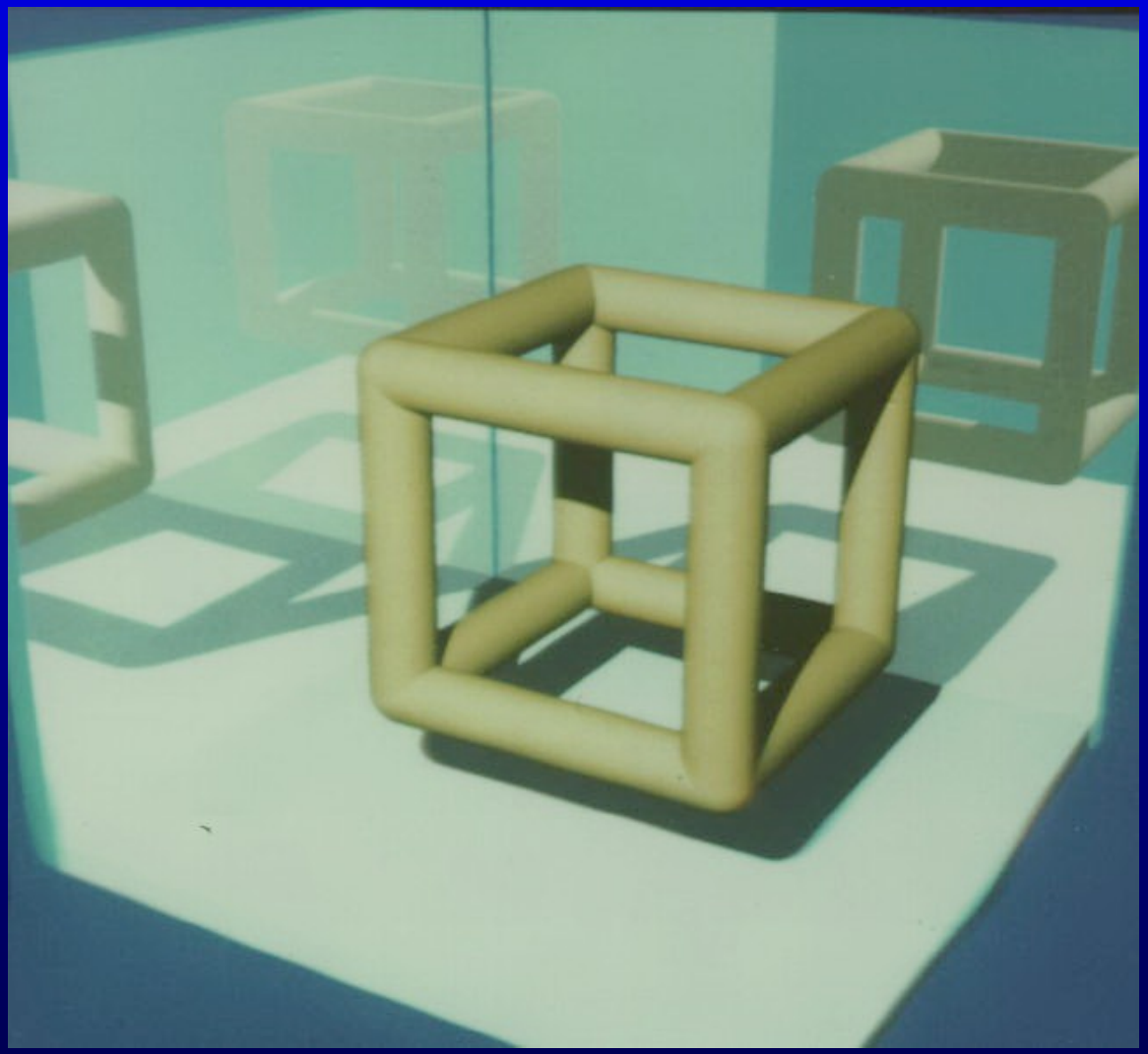


Marching Cubes 1984





GERT - GE Ray Tracer 1985

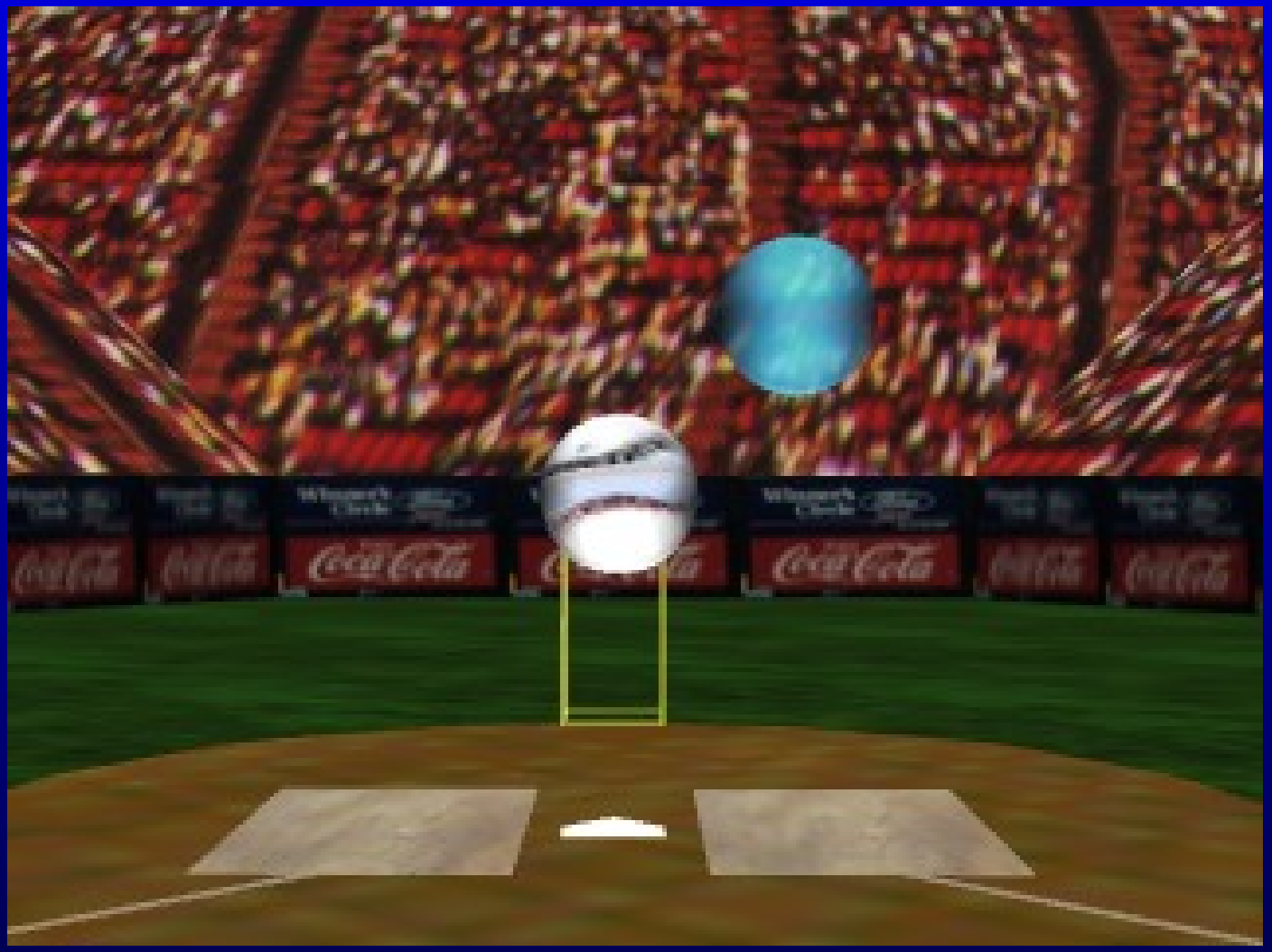


Siggraph ‘87

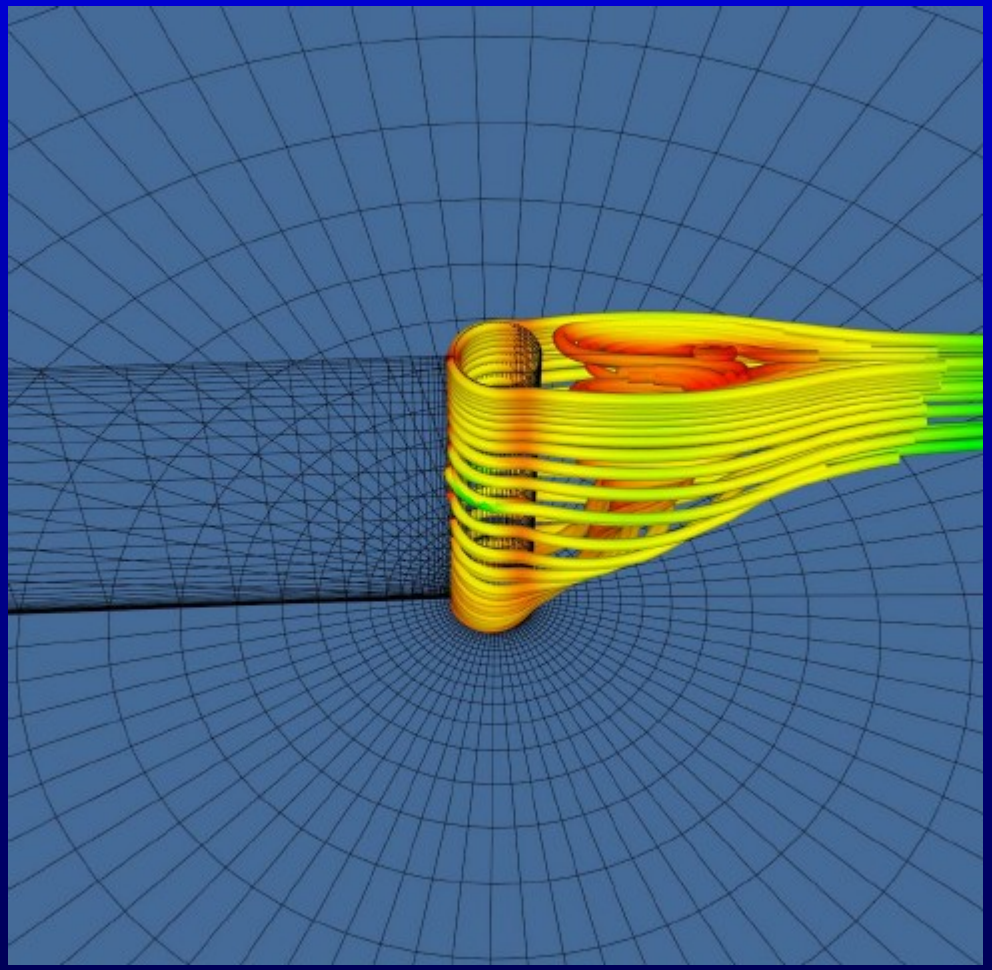

SIGGRAPH
COURSE 14
Object-Oriented
Geometric
Modeling &
Rendering
SIGGRAPH
PANEL
Tool Kits:
A Product
of their
Environments
Volumetric
Modeling


WELCOME
SIGGRAPH
PANEL
Tool Kits:
A Product
of their
Environments
Volumetric
Modeling

Baseball Visualization 1989



Stream Polygons - 1991



1991

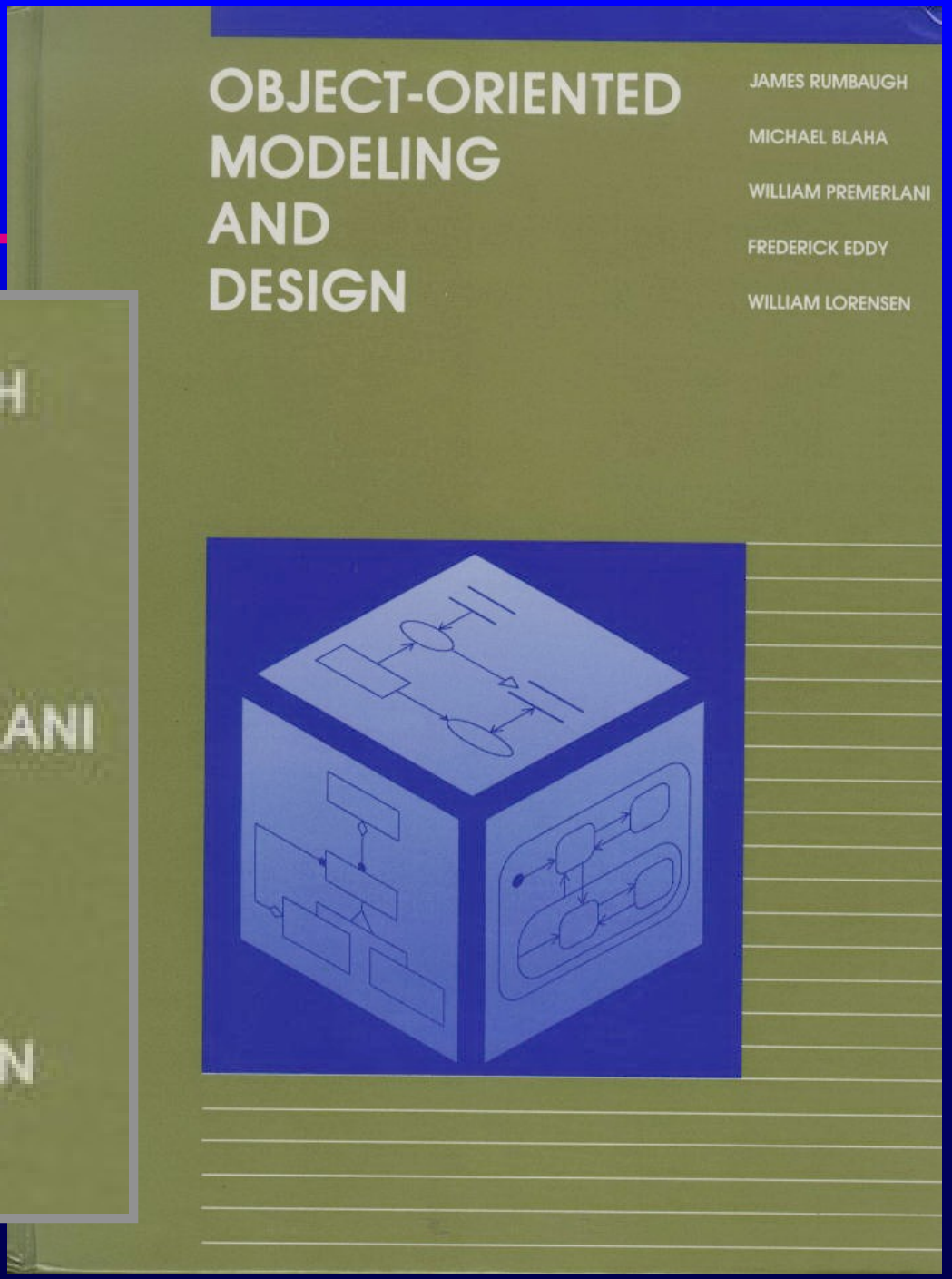
JAMES RUMBAUGH

MICHAEL BLAHA

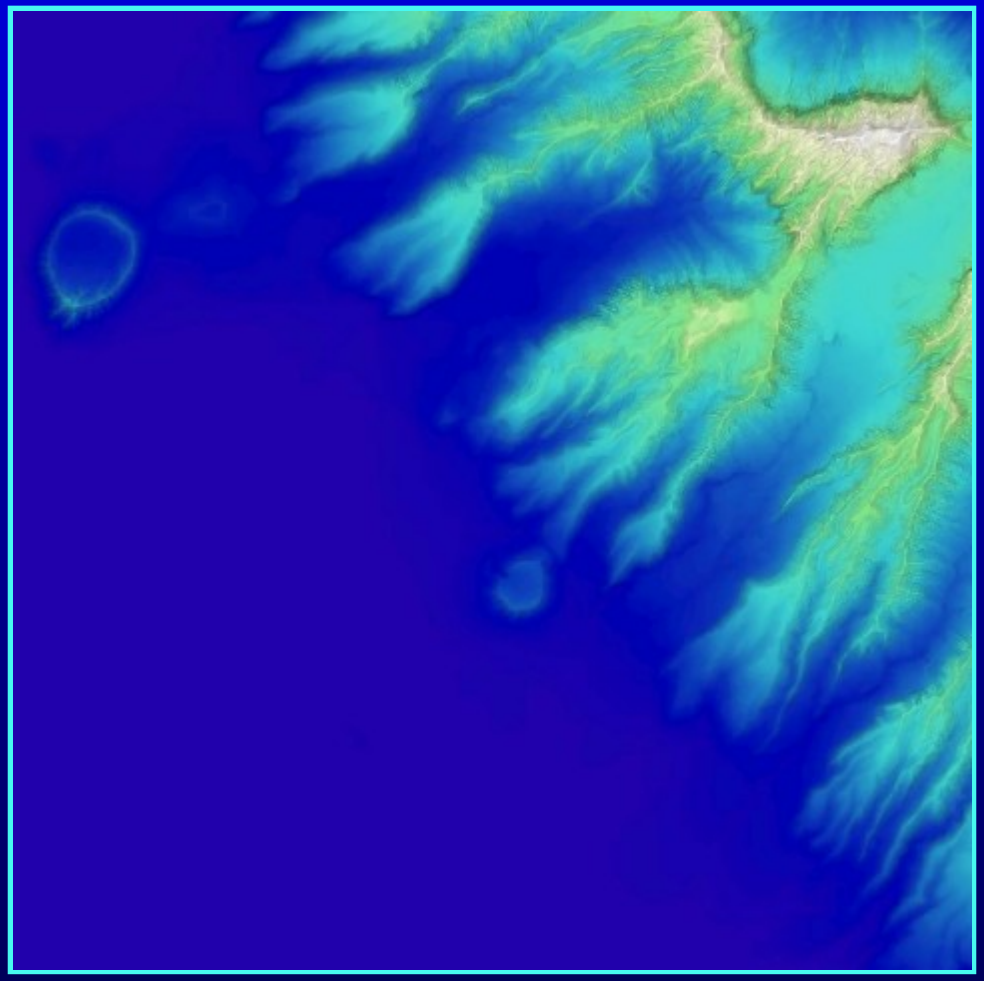
WILLIAM PREMERLANI

FREDERICK EDDY

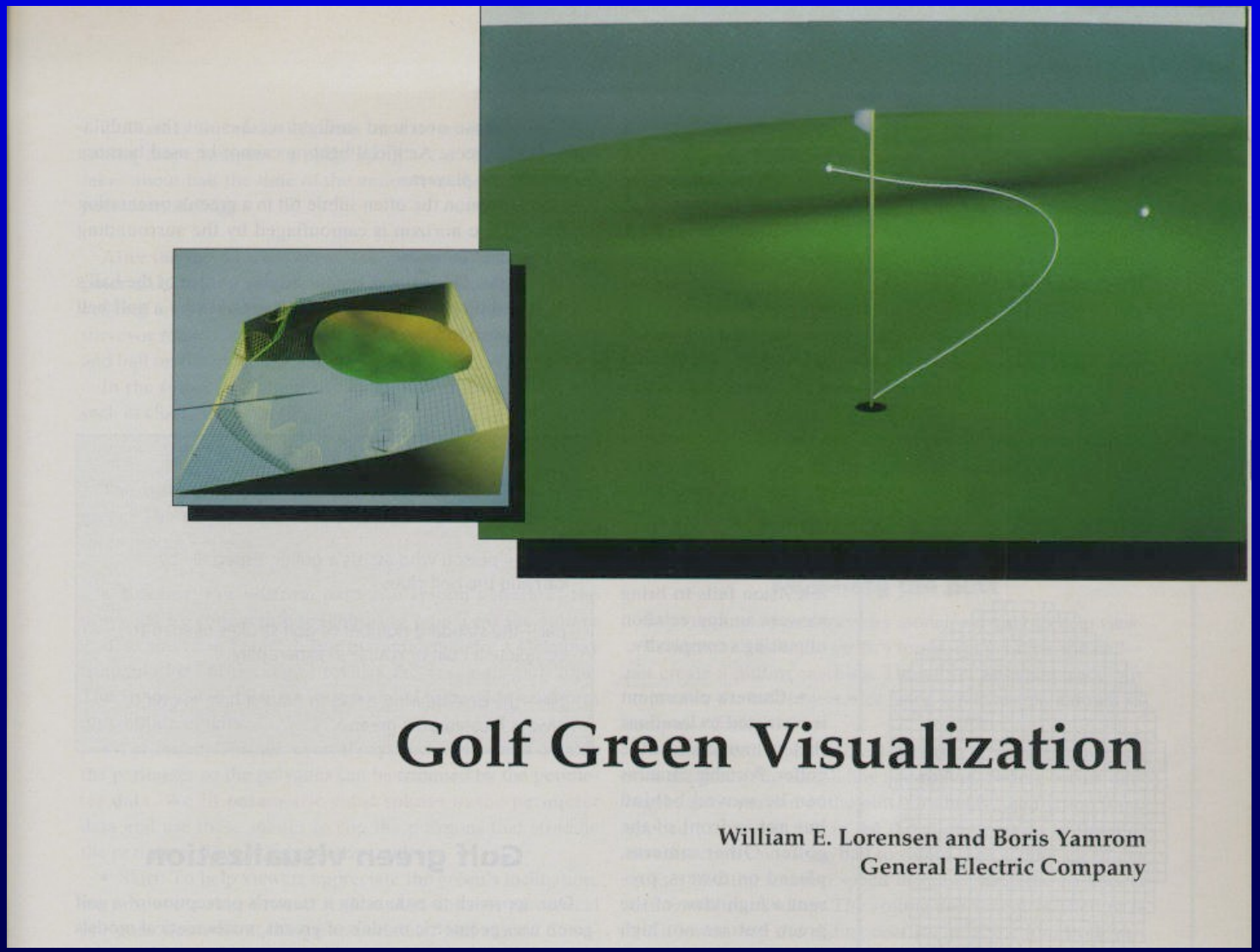
WILLIAM LORENSEN



Triangle Decimation - 1992



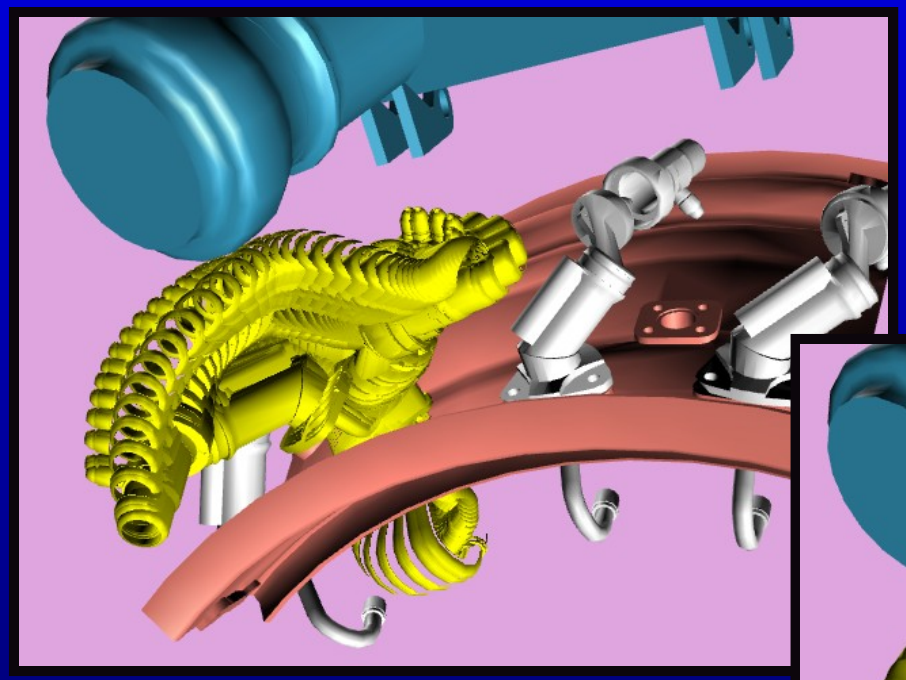
IEEE CG&A 1992



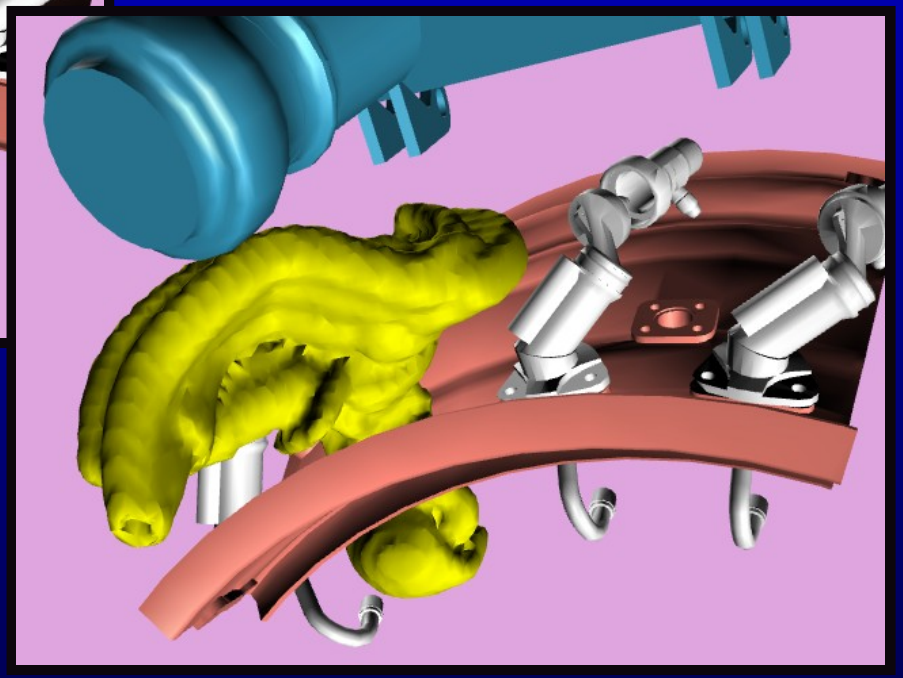
Golf Green Visualization

William E. Lorensen and Boris Yamrom
General Electric Company

Swept Surfaces 1993

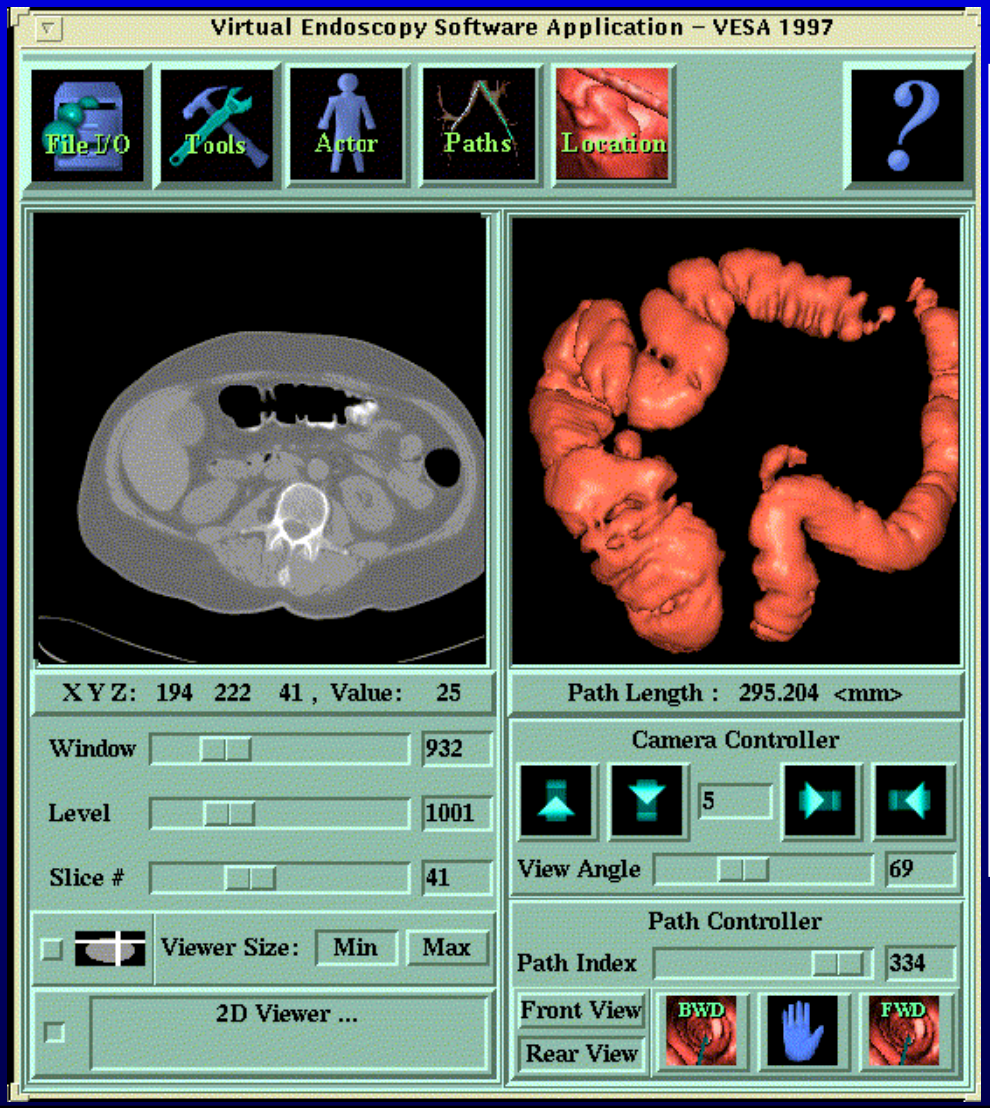


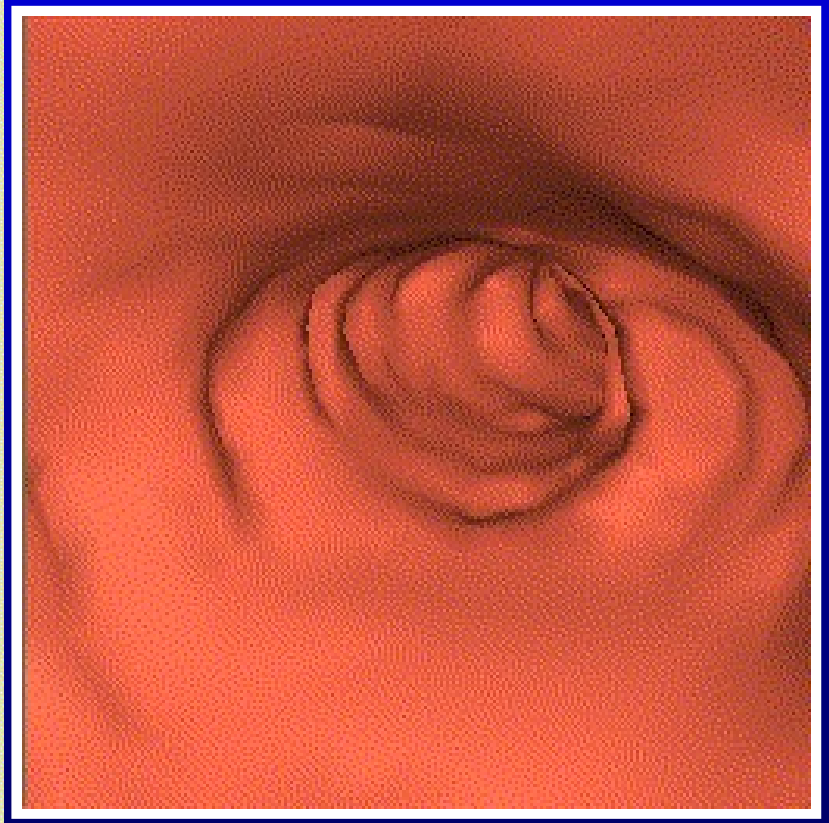
Removal Path



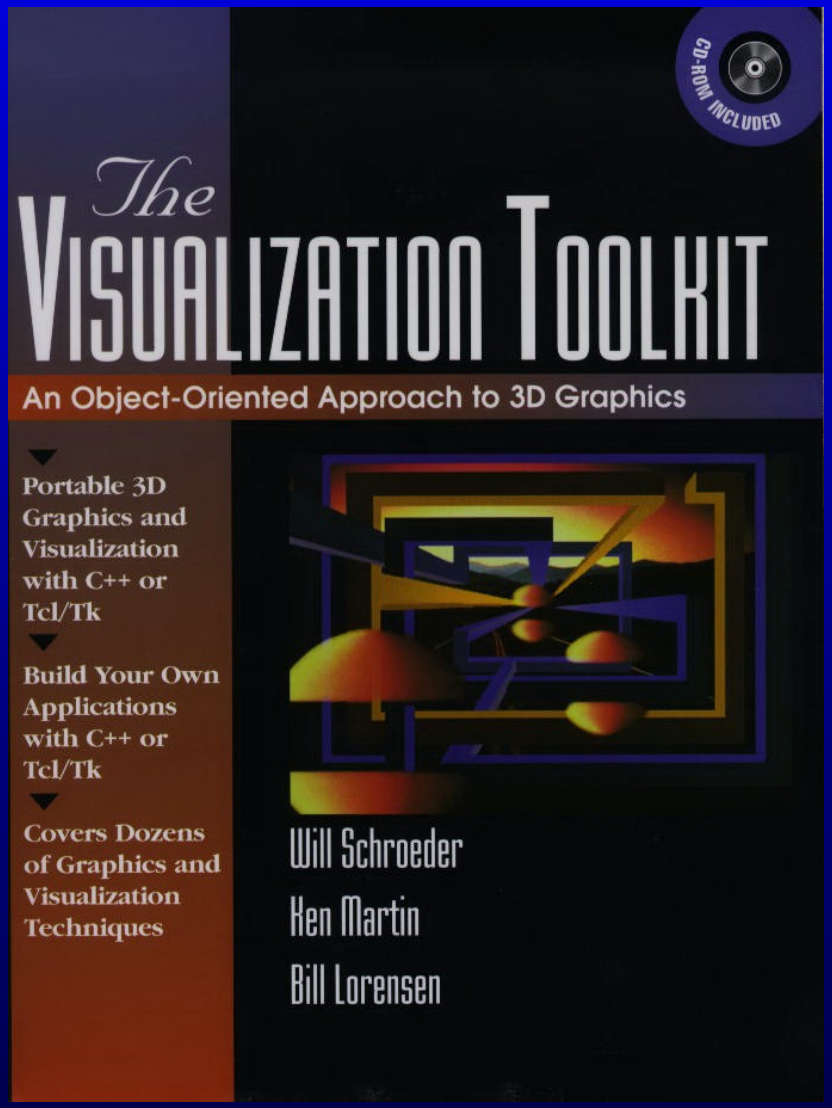
Swept Surface

Virtual Endoscopy 1994

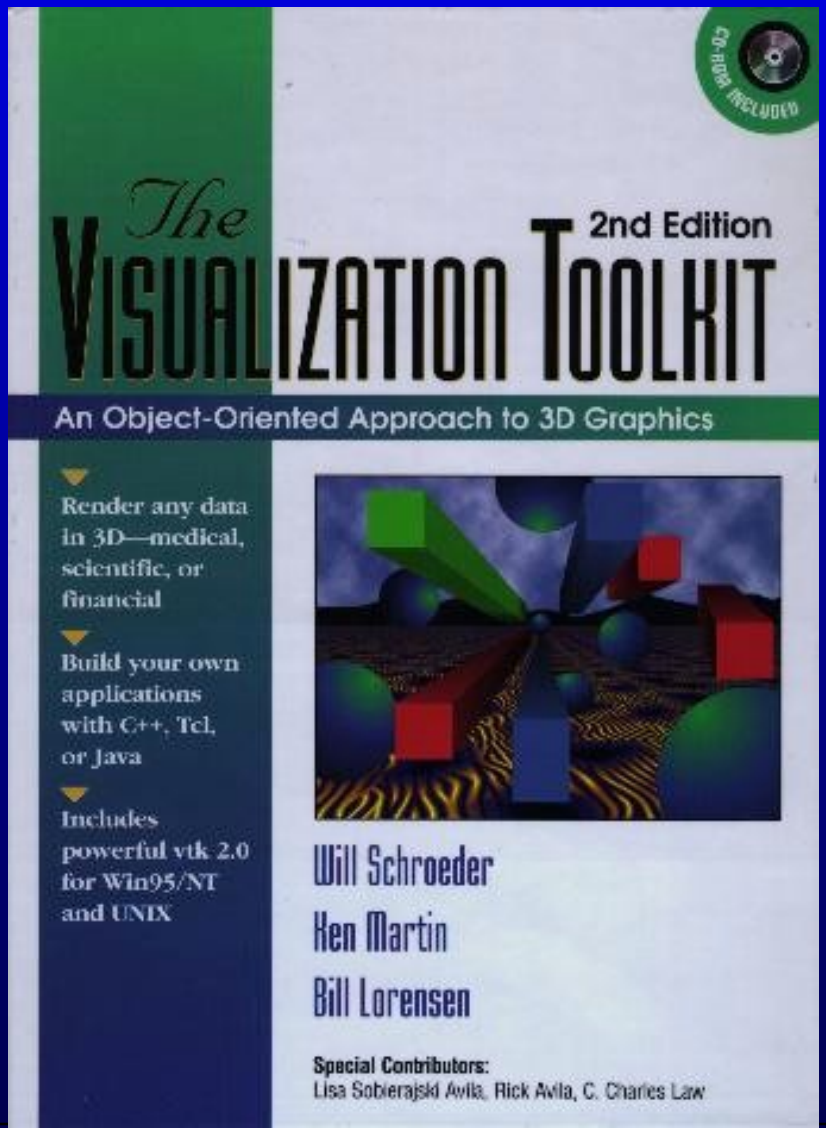




vtk1.0 1995

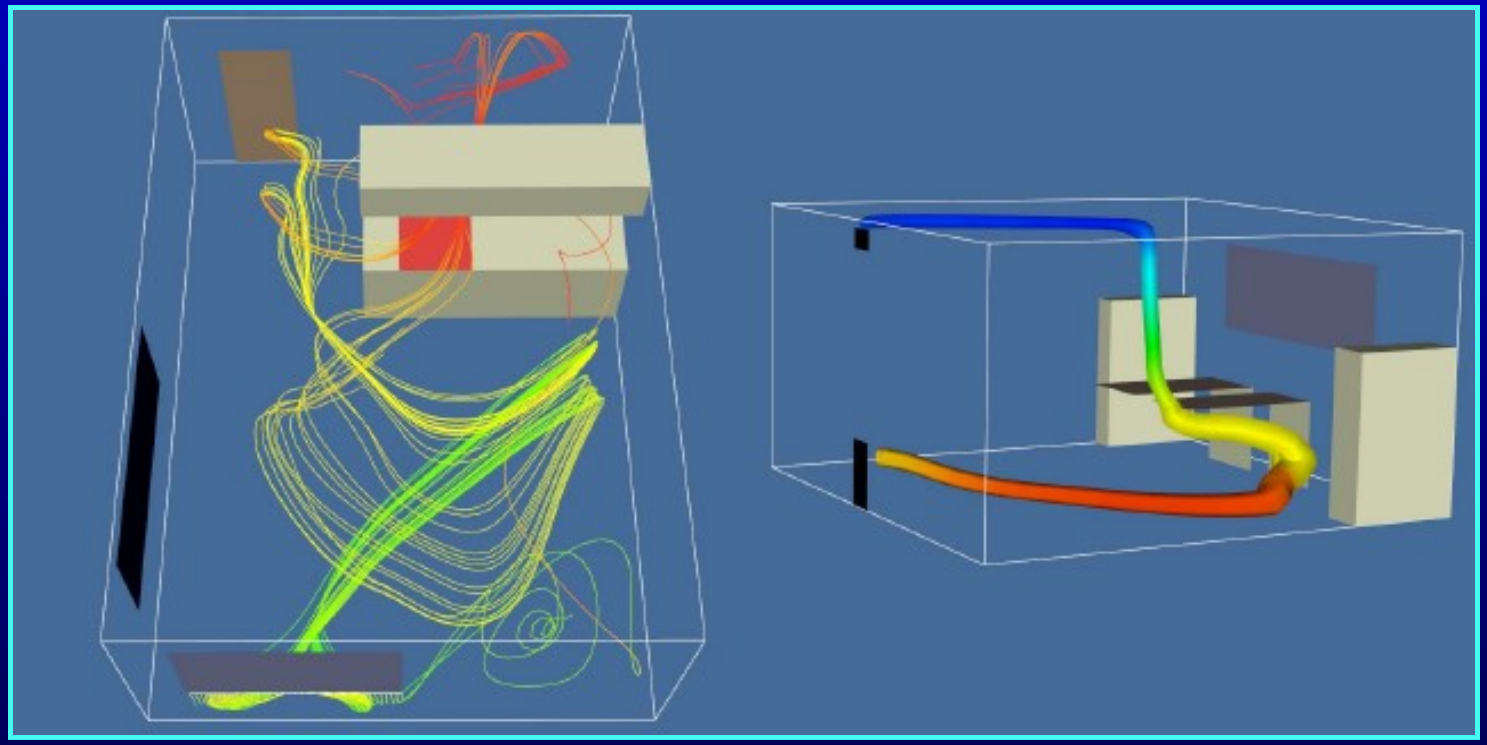


vtk2.0 1997

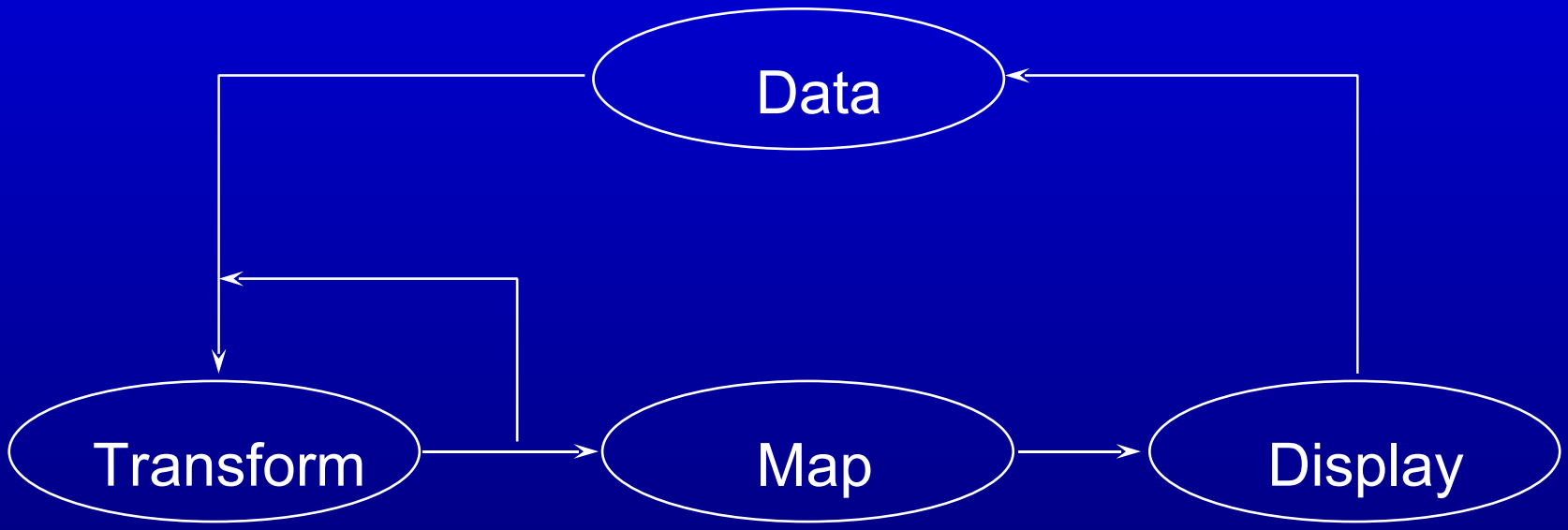


Visualization

The transformation of data into images



Visualization



Object-Oriented Visualization

- Goals
  - Reusability
  - Portability
    - »Operating System
    - »Graphics API
    - »User interface

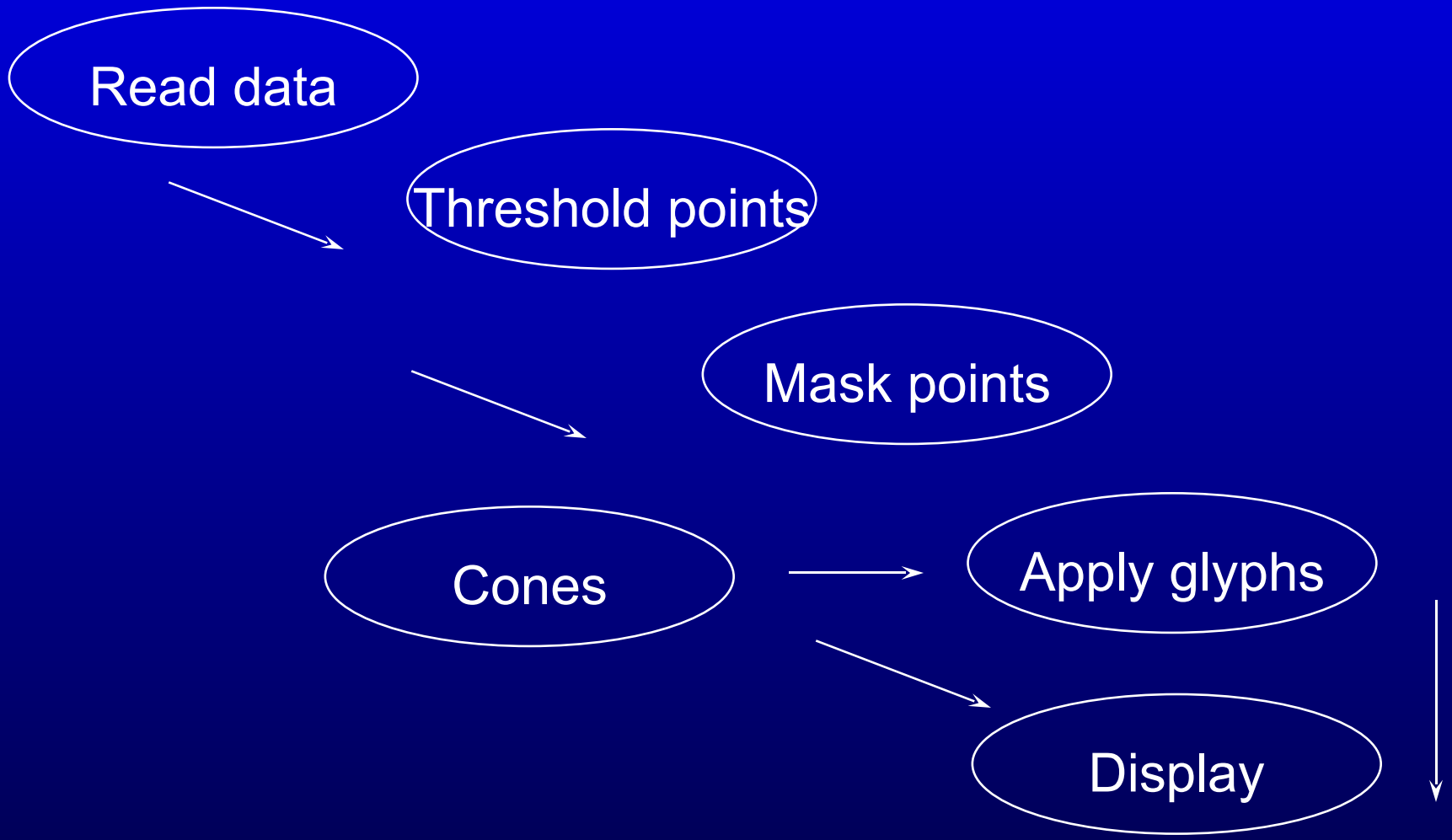
Object-Oriented Visualization

- Goals
  - Longevity
  - Simplicity

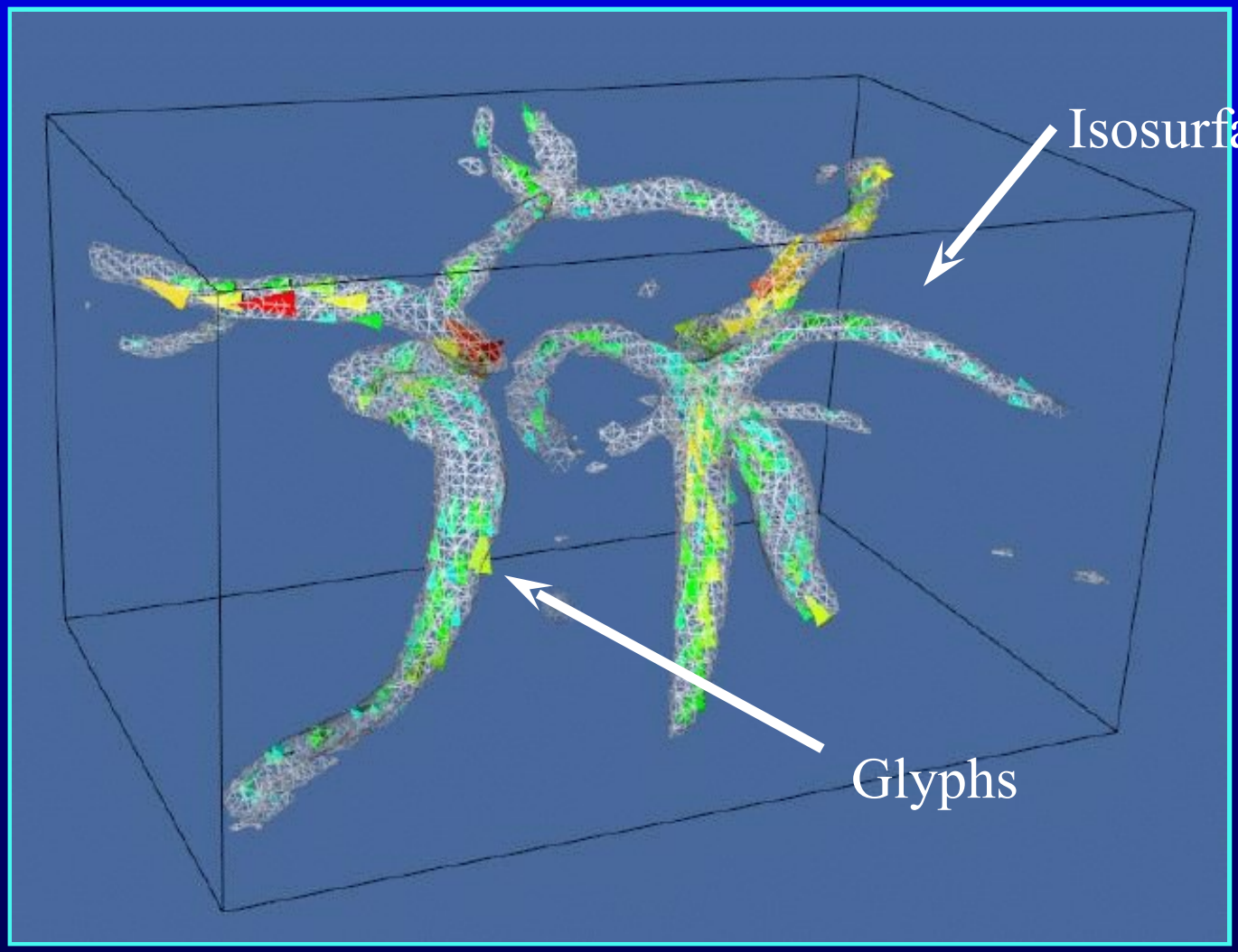
Motivation

- Visualization is still evolving
- New techniques introduced yearly
- Multiple algorithms often used

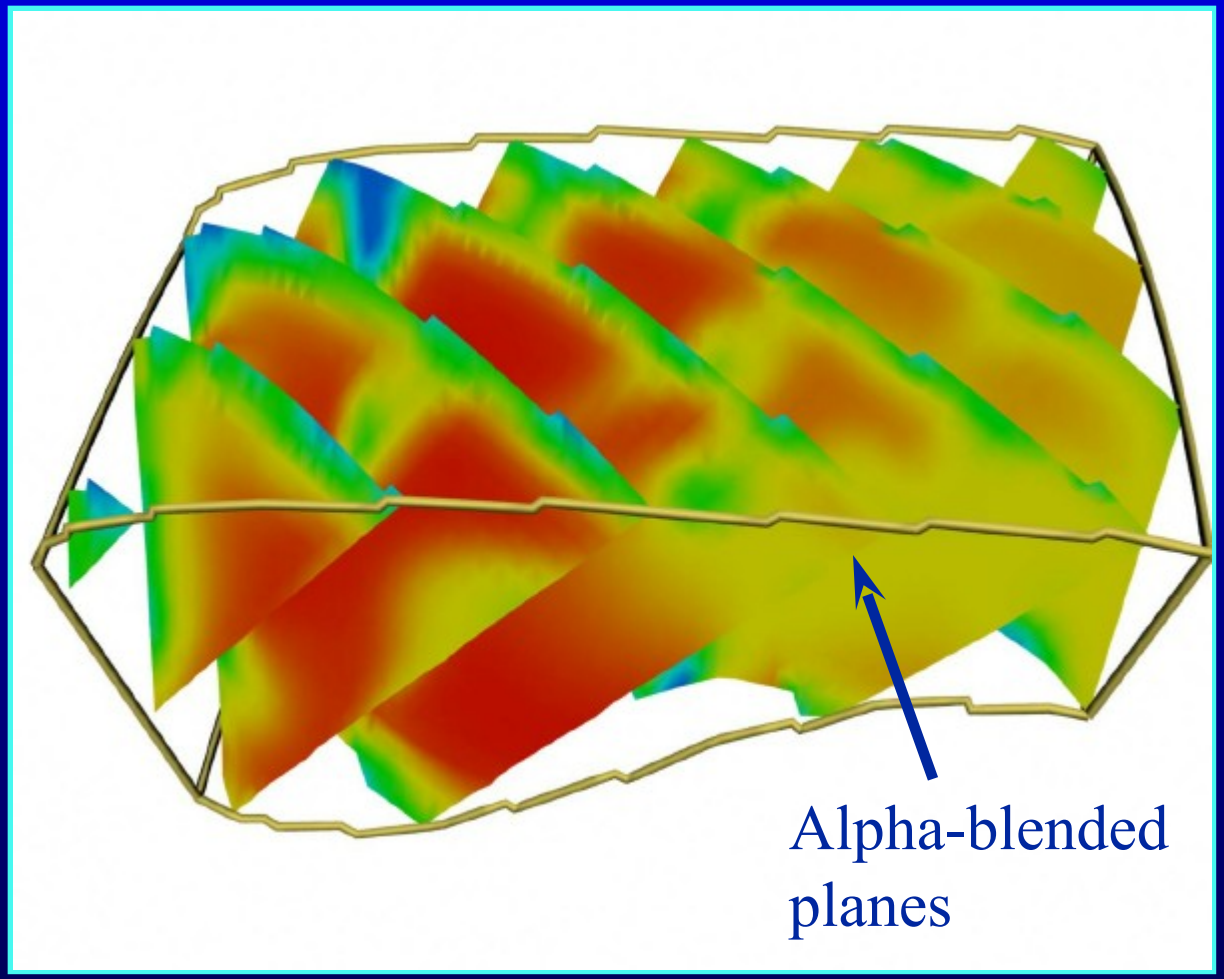
Anatomy of a Visualization



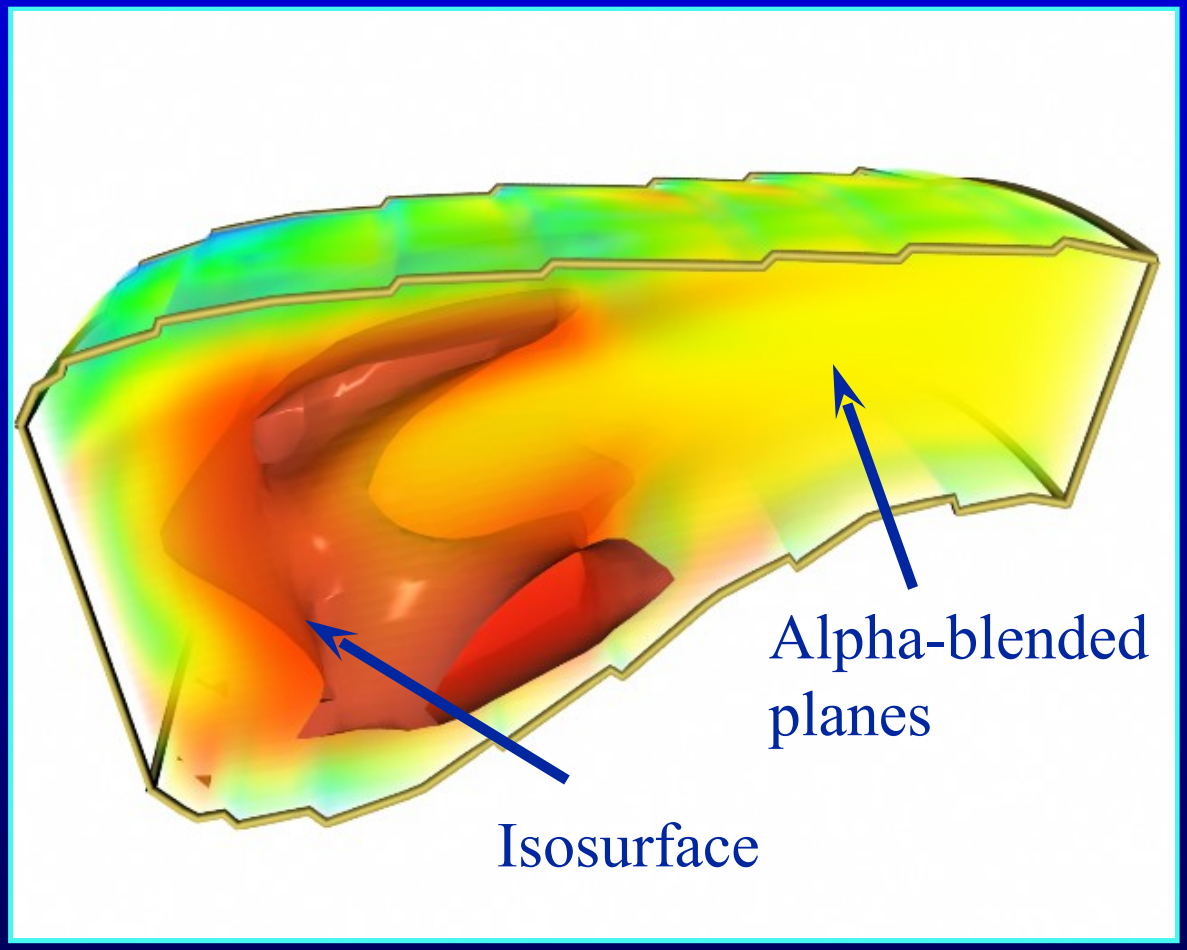
Glyphs and Isosurfaces



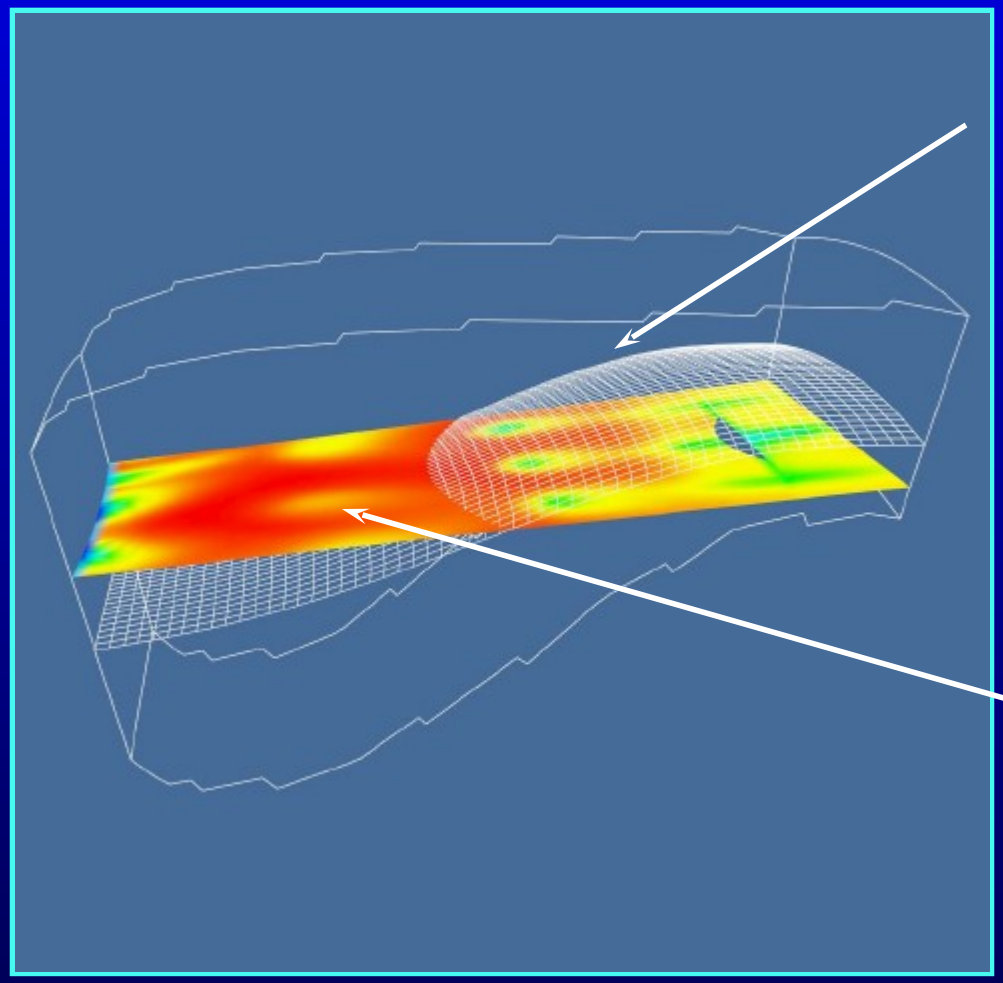
Slicing



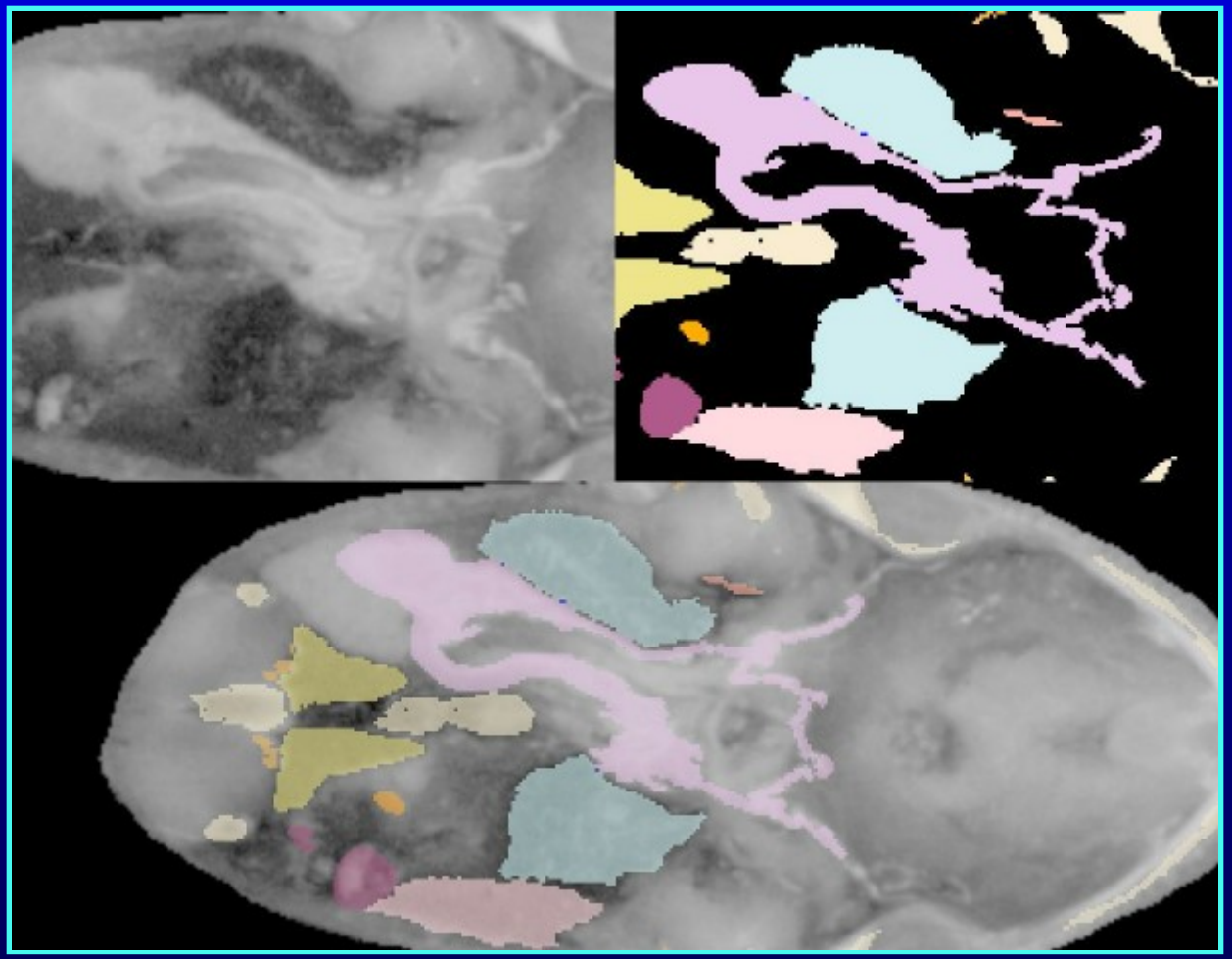
Volume Rendering using Alpha Planes



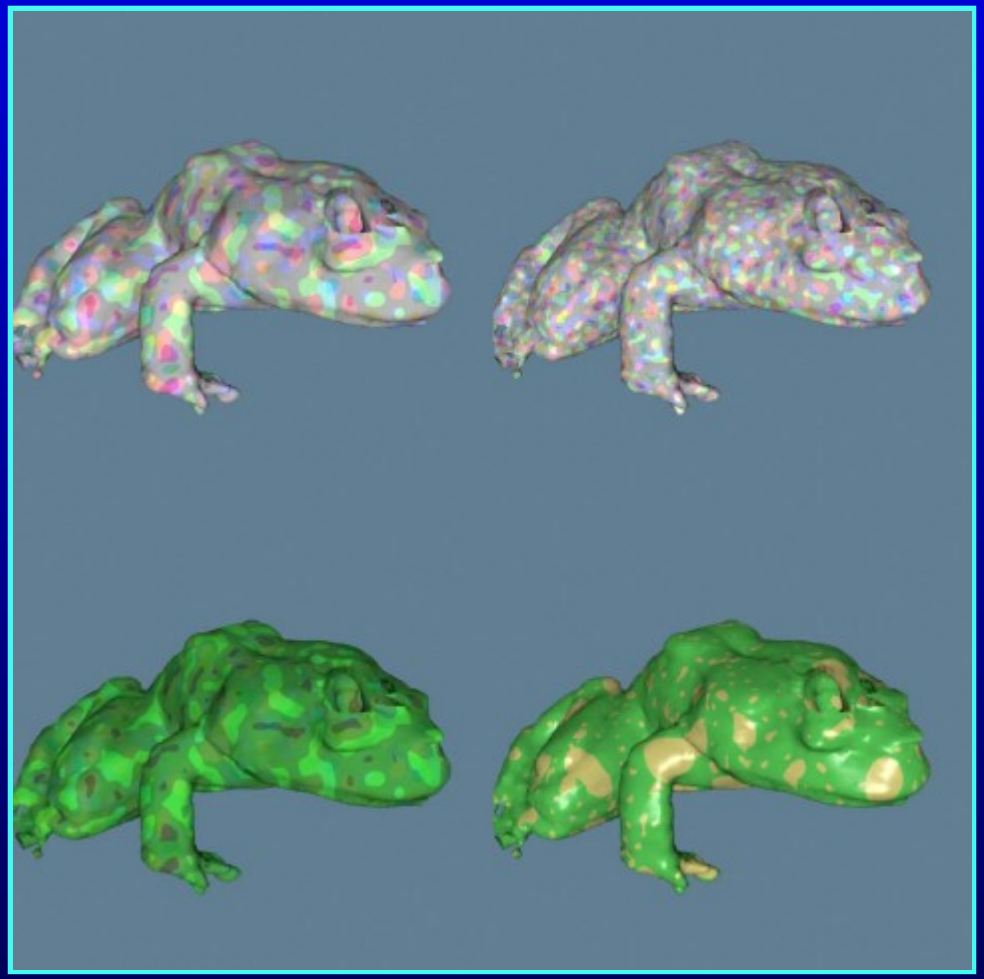
Data Probing



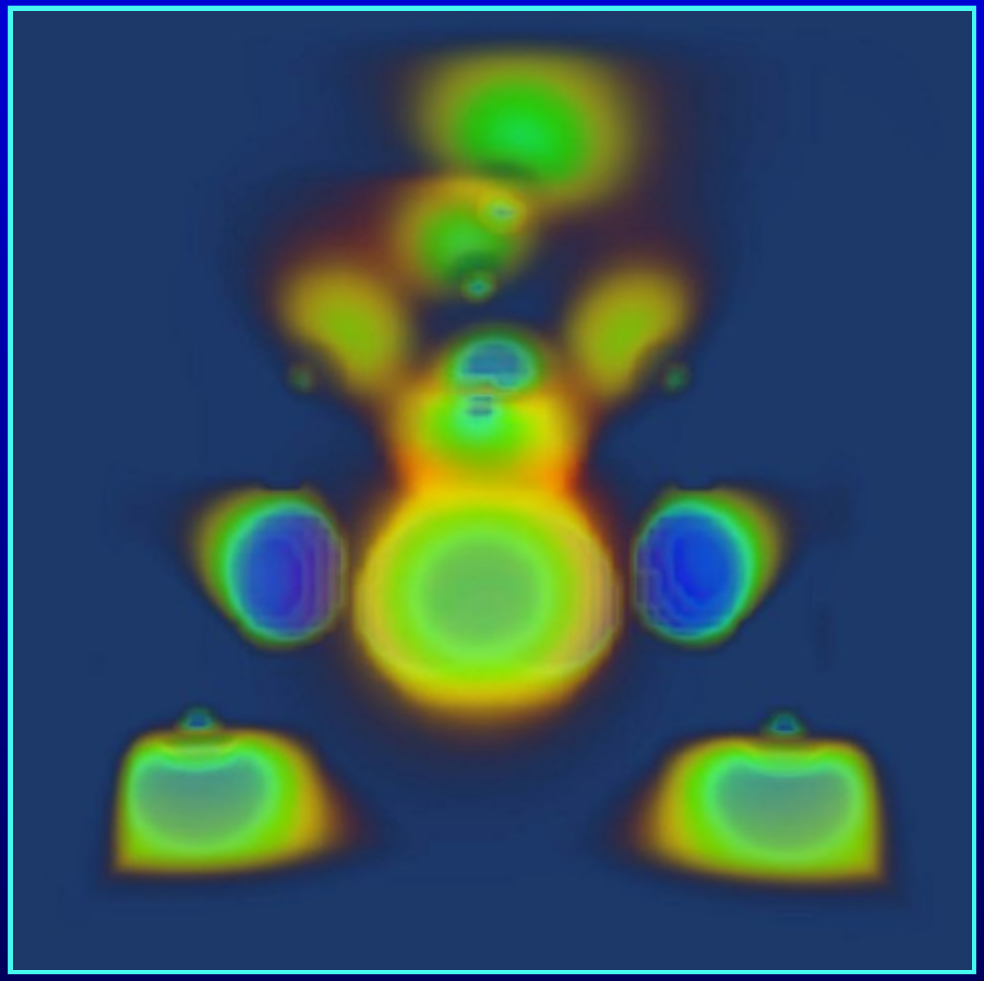
Texture Mapping



High Quality Software Rendering



Volume Rendering



Why Object-Oriented?

- Visualization is a complex task
  - OO can deal with complexity
- Easy to map application domain to implementation domain
  - great fit with graphics
- OO promotes modular systems

Why Object-Oriented?

- OO Technology is mature
- OO Technology is being taught in college
  - There are several texts available
- OO Technology is accepted by industry

Systems versus Toolkits

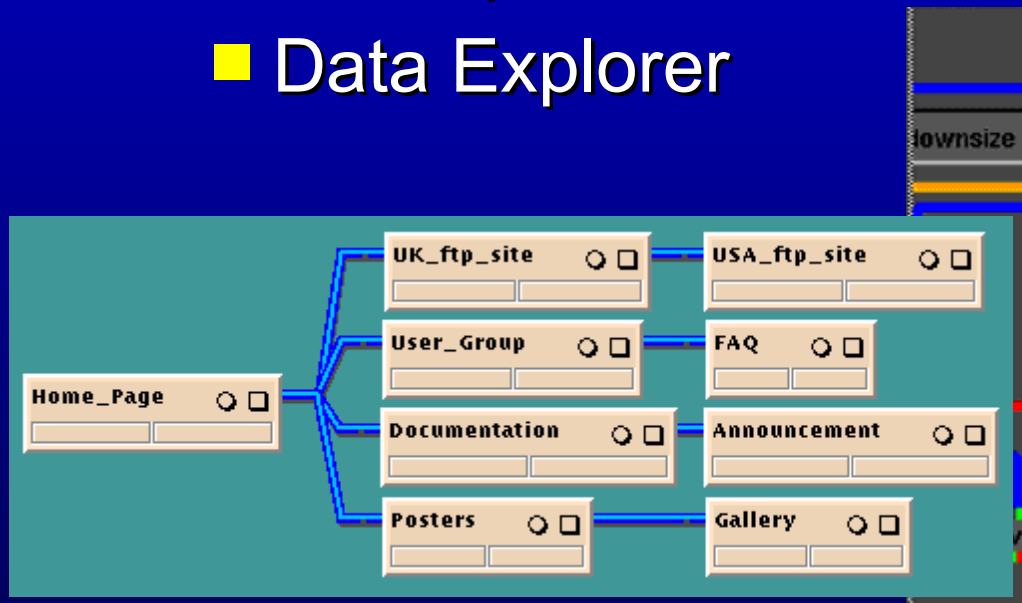
- Systems
  - Self-contained
  - Often turn-key
  - Great reuse
  - Integrated user interface
  - All or nothing

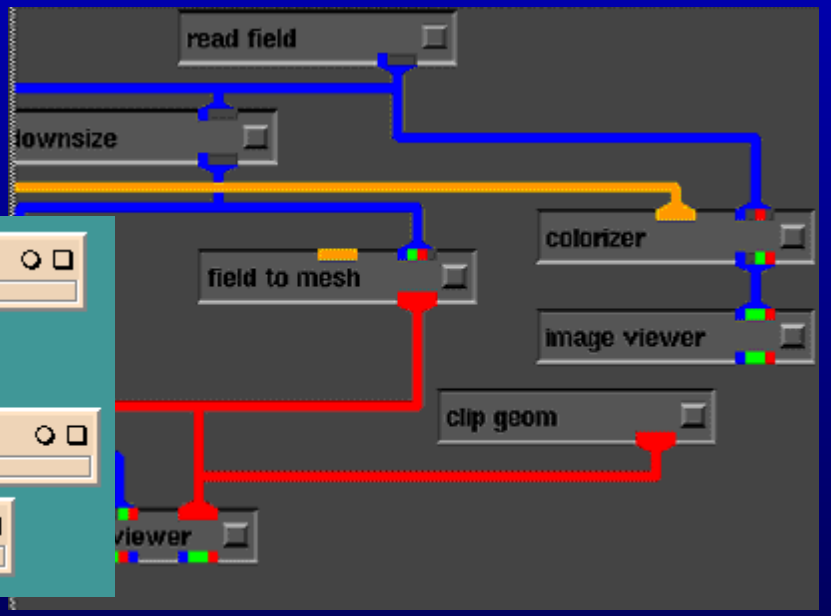
Systems versus Toolkits

- Toolkits
  - More than a library
  - Includes an architecture
  - Use only what you need
  - Independent of user interface

Systems

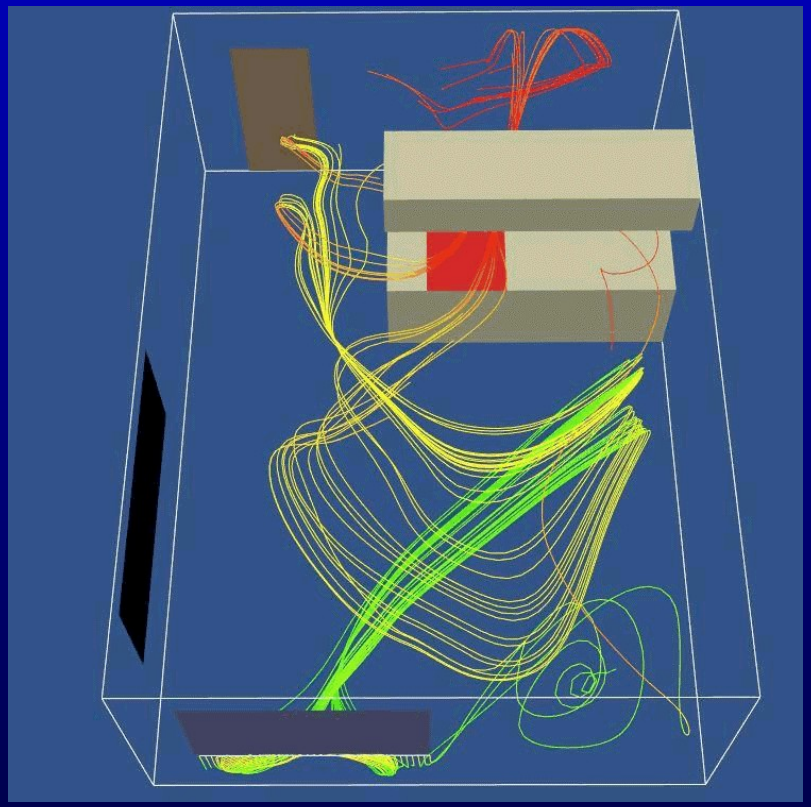
- Examples
  - AVS
  - Iris Explorer
  - Data Explorer





Toolkits

- Examples
  - The Visualization Toolkit
  - Inventor
  - ISG's IAP



LYMB

- An object-oriented system
  - **L**orensen, **Y**amrom, **M**cLachlan, **B**arry
  - A methodology for implementing OO concepts in C
  - An interpreter for implementing object interaction
  - Object interaction via run-time message passing

LYMB's History

- Started in 1984 as an animation system
  - OSCAR - Object-oriented SCene AnimatoR
- Initial system had 25 classes for animation and rendering
- Current system has over 600 classes
- After a short time, we realized that we had much more than an animation system

LYMB Applications

- Decimation
  - triangle reduction
- Visage
  - scientific visualization
- Golf
  - golf green and putting visualization
- Product Vision
  - design for maintainability
- Dozens of small custom interfaces

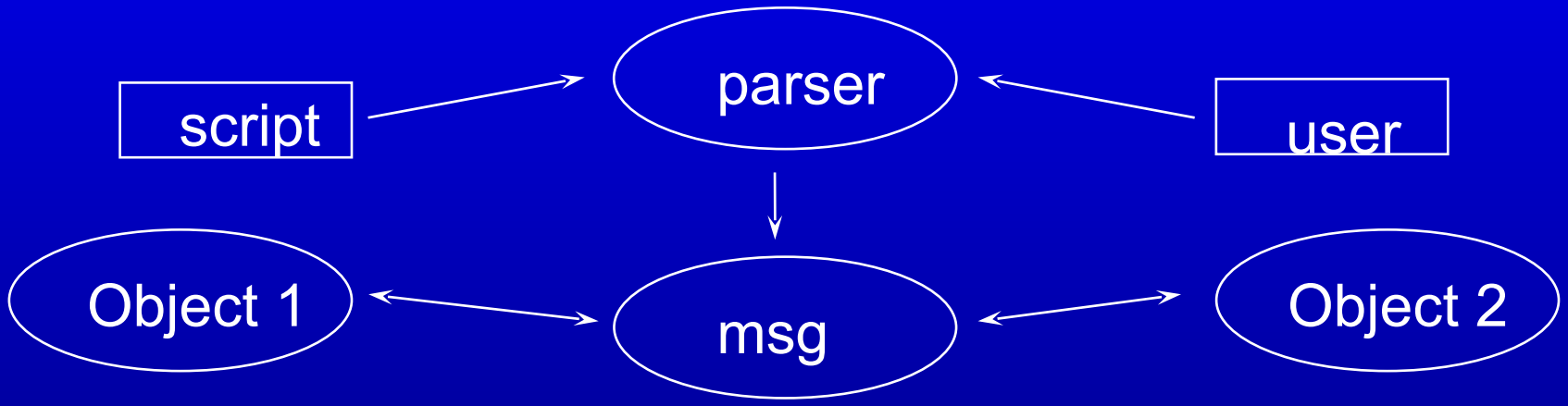
OSCAR Classes

- System core
  - message passing, argument handling
  - parser
  - collection, scalars, vectors
- Rendering classes
  - actors, cameras, lights
  - renderers
- Animation classes
  - scenes, cues
  - keyframes

LYMB Classes

- Visualization
  - marching cubes
  - decimation
  - stream polygons
- User Interface
  - Xlib
  - Motif

LYMB Architecture



- Efficient (object implemented in C)
- Rapid application development (interpreter)
- Objects and users interact via uniform message passing protocol
- Portable
  - C, Unix, X, Motif, Graphics Standards

Graphics Example (simple)

```
ply_modeller new: bunnyModel
    filename=`bunny.ply';
actor new: bunny
    modeller=bunnyModel;
renderer new: aren
    actors=bunny;
aren render!;
```

Graphics Example (interaction)

```
<motif_renderer.meta
obj_modeller new: beet_obj
    filename=`beethoven.obj';
actor new: beethoven modeller=beet_obj
    property=brass;
property new: brass diffuse_color=(.2,.8,.4)
   specular=.4 specular_power=30;
motif_renderer new: aren actors=[actor instances?]
   render!;
motif start!;
```

1992 LYMB Recognition

In Recognition of User Achievement
Computerworld's Object Application Award

Presented to GE Corporate Research and Development in recognition of outstanding custom application development using object technology in the category of
"Best implementation of a reusable development environment for company deployment".

Bill Laberis
Editor-in-Chief, Computerworld

COMPUTERWORLD
The Newspaper of IS

Christopher M. Stone
Chairman, President & CEO, Object Management Group

OMG
OBJECT MANAGEMENT GROUP

LYMB: The Good

- Simple Concepts
- Started Small
- High reuse rate
- Portable (Unix only)
- Easy to use
- High acceptance
- Uniform methodology

LYMB: and The Bad

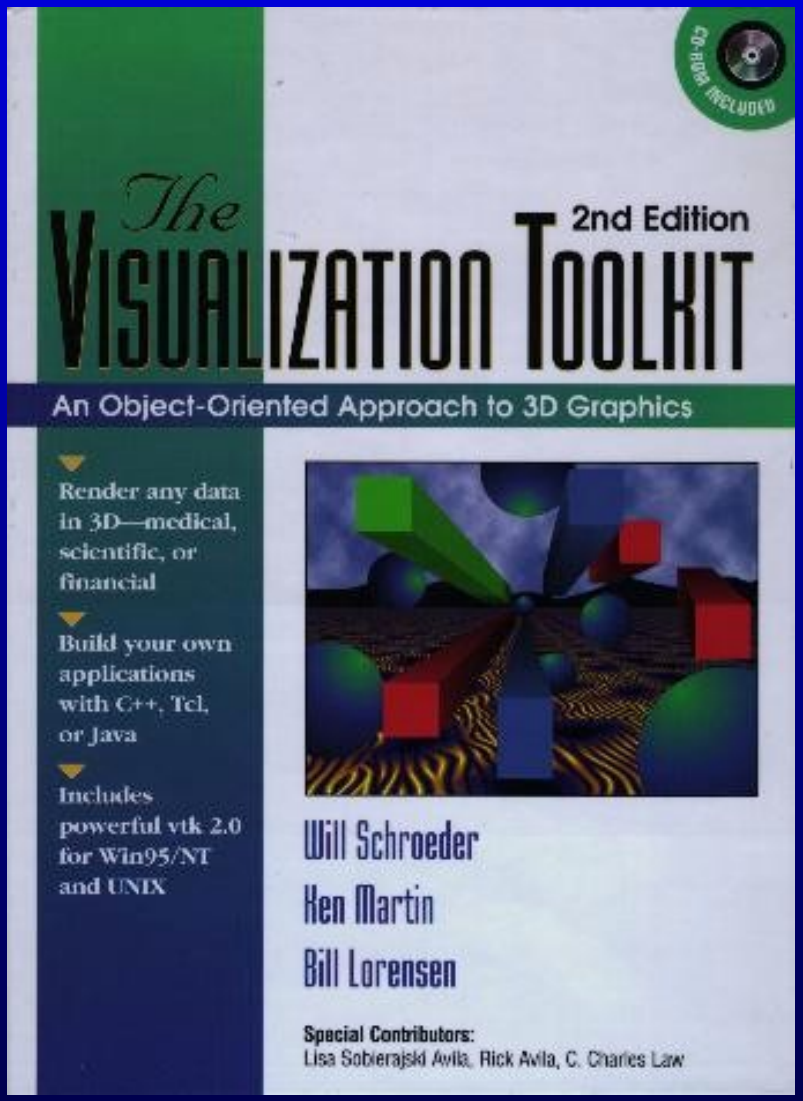
- Complex concepts
- Creeping features

- Weak documentation
- Big learning curve
- All or nothing
- Proprietary

The Visualization Toolkit

- Started as an example implementation for a text book
- Implemented in C++
  - runs on Unix workstations and PC's
- Many concepts from LYMB
  - similar graphics abstractions
  - visualization pipeline
  - more flexible data model
- No interpreter (initially)

The Visualization Toolkit 2.0

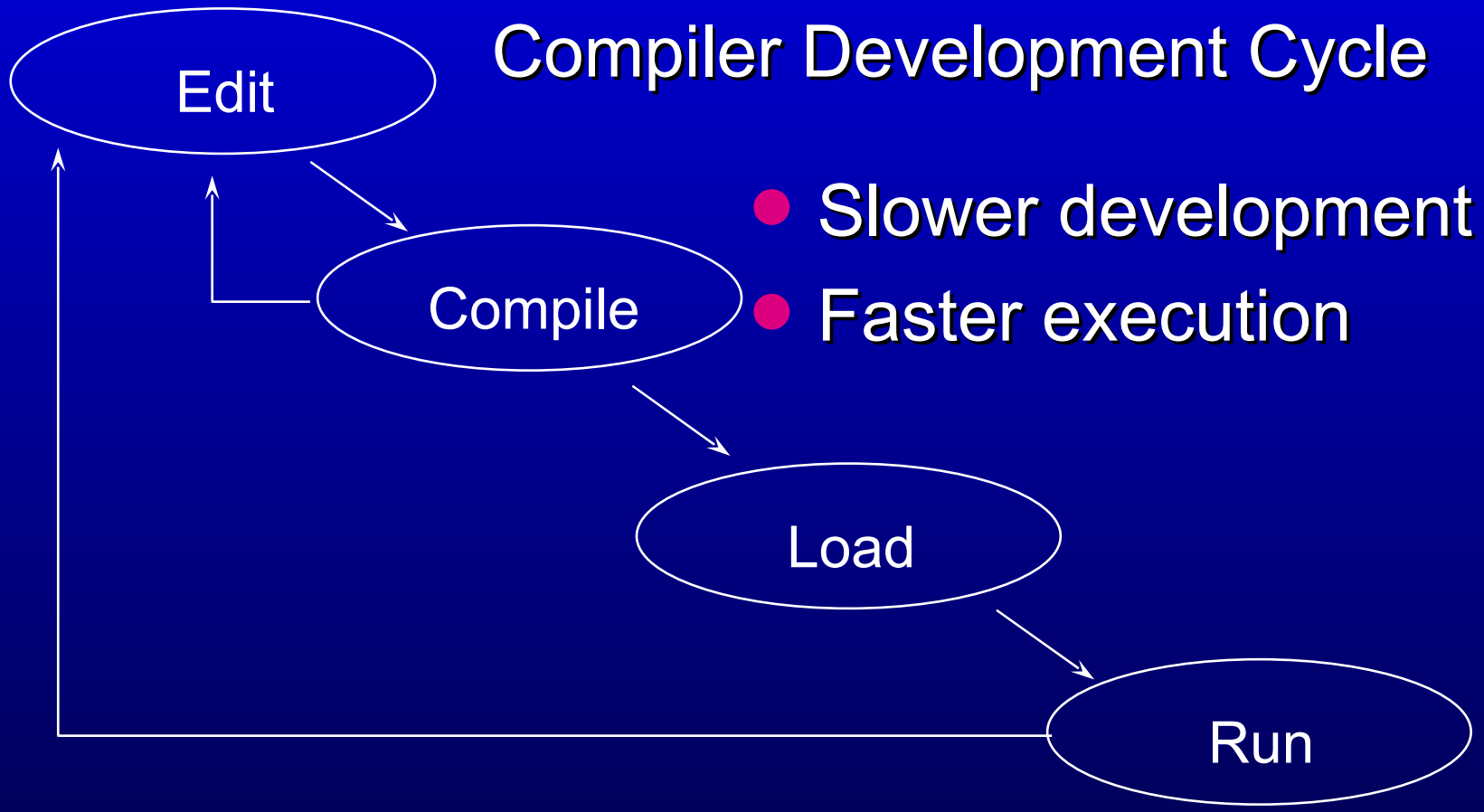


- vtk1.0 Plus
  - Volume Rendering
  - Image Processing
- Includes CD-ROM
  - Unix/PC Source code
  - Documentation
  - PC executable
  - Examples

The Visualization Toolkit

- Interpreters added through automatically generated wrappers
  - tcl
  - java, java beans
  - python...
- All documentation contained within code
  - makes for easy man page, html, etc.... generation
- Source code available via Internet

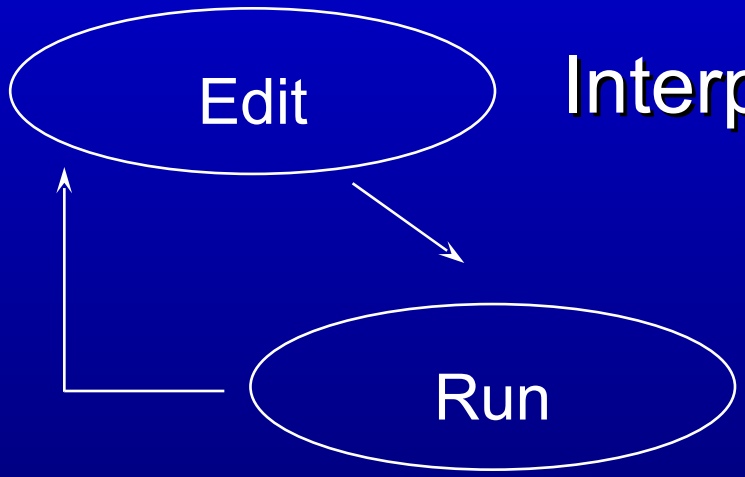
Compiled versus Interpreted



Compiler Development Cycle

- Slower development
- Faster execution

Compiled versus Interpreted

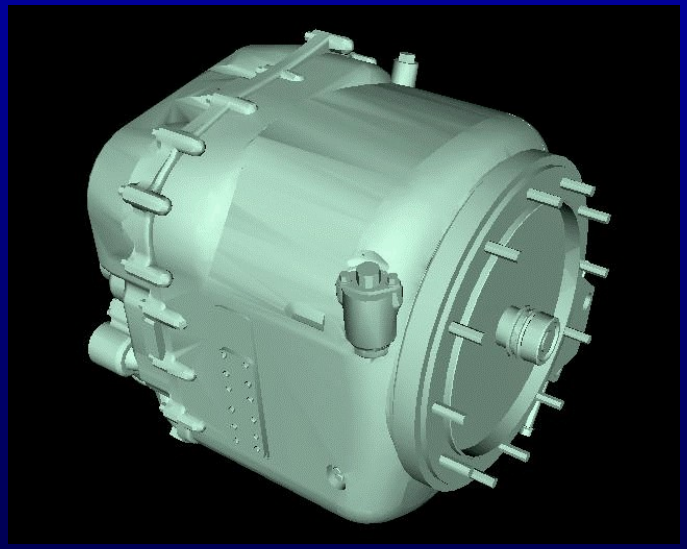


Interpreter Development Cycle

- Faster development
- Slower execution

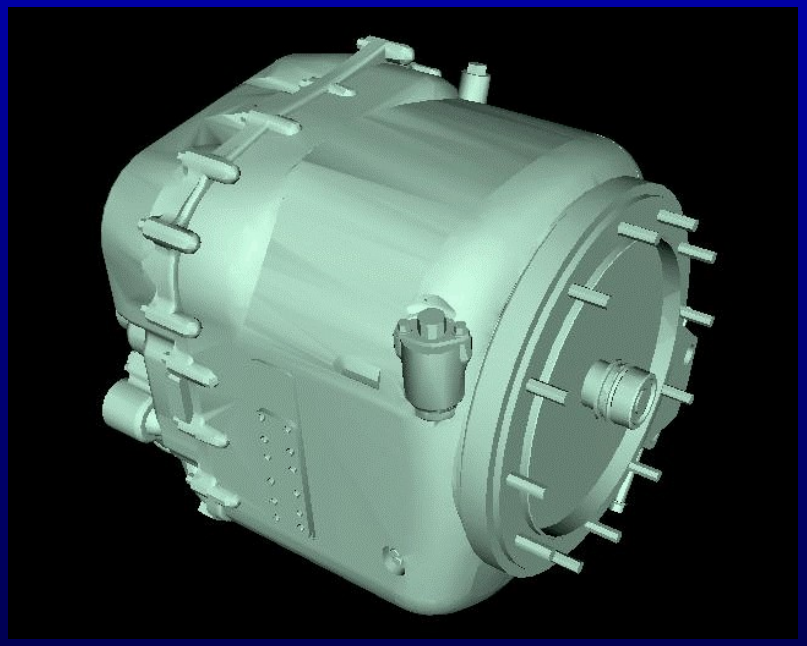
VTK - C++ Example

```
vtkRenderer *aren = vtkRenderer::New();
vtkRenderWindow *renWIn = vtkRenderWindow::New();
  renWin->AddRenderer( aren);
vtkRenderWindowInteractor iren = vtkRenderWindowInteractor::New();
  iren->SetRenderWindow(renWin);
vtkSTLReader stl = vtkSTLReader::New();
  stl->SetFileName (“cad.stl”);
vtkPolyDataNormals normals = vtkPolyDataNormals::New();
  normals->SetInput (stl.GetOutput ());
  normals->SetFeatureAngle (30);
vtkPolyDataMapper mapper = vtkPolyDataMapper::New();
  mapper->SetInput (normals.GetOutput ());
vtkActor actor1 = vtkActor::New(0;
  actor1->SetMapper (mapper);
  actor1->GetProperty () ->SetColor (.8, 1, .9);
 aren->AddActors(&actor1);
 renWin->Render ();
 iren->Start ();
```



VTK - Tcl Example

```
vtkRenderer aren
vtkRenderWindow renWin
  renWin AddRenderer aren
vtkRenderWindowInteractor iren
  iren SetRenderWindow renWin
vtkSTLReader stl
  stl SetFileName “cad.stl”
vtkPolyDataNormals normals
  normals SetInput [stl.GetOutput]
  normals SetFeatureAngle  30
vtkPolyDataMapper mapper
  mapper SetInput [normals GetOutput]
vtkActor actor1
  actor1 SetMapper mapper
  [actor1 GetProperty] SetColor .8 1 .9
aren AddActors  actor1
renWin Render
iren Start
```



A System Application: LYMB

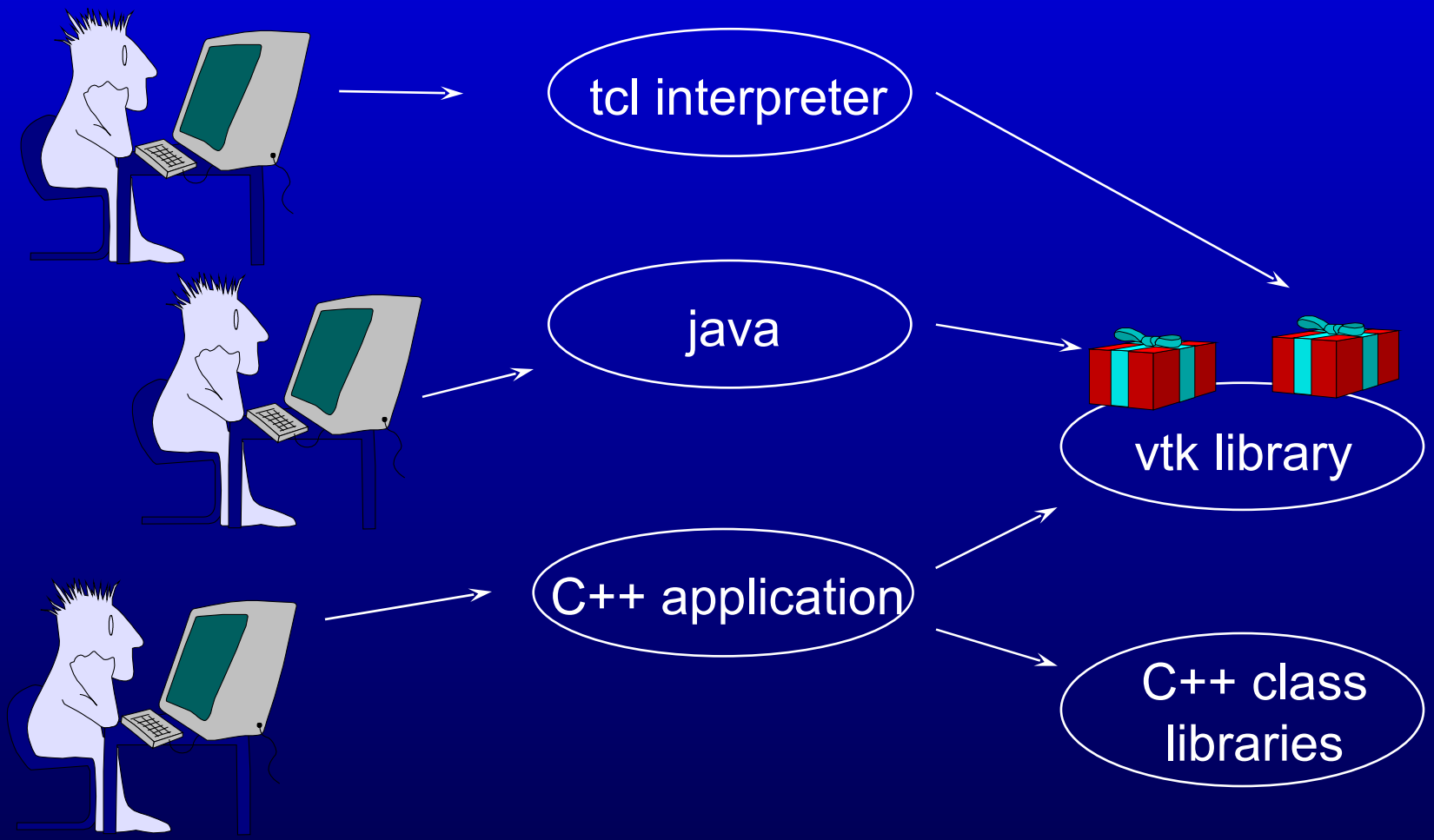
parser

msg

motif lib

lymb objects

A Toolkit Application: VTK



Lessons Learned

- Object-oriented is good
  - if you enforce a methodology
- Interpreters are good
  - but don’t invent your own
- Abstractions are good
  - they protect against future changes beyond your control

Lessons Learned

- C++ is mature
- Isolate the user interface
- Keep it simple!
- Watch those features!
- Proprietary is bad!!!

Visualization Today

- The Visualization Community is no longer in control
  - Technology drivers have changed
  - Customers expectations are high
- but…we do have lots of software experience
  - OO is a proven technology
  - We have a large installed base
  - We know our application domain

External Forces

- Internet
- Wintel
- Standards
- Language Wars

Internet

- The right information, to the right person, at the right time…
- In the future,
  - Most applications will be Internet-ready
  - Finance market will solve security problem
- But,
  - Performance remains an issue

Wintel

- Microsoft OS's and API's dominate
- Intel processors dominate
- Scientific Visualization is a small player compared to
  - Entertainment
  - Word Processing
- What can we leverage???

Standards

- OpenGL
  - Available on Unix and PC's
  - Cheap accelerator boards
  - Impacts graphics and volume rendering
- Java
  - Write once, run anywhere (???)
  - Performance

Language Wars

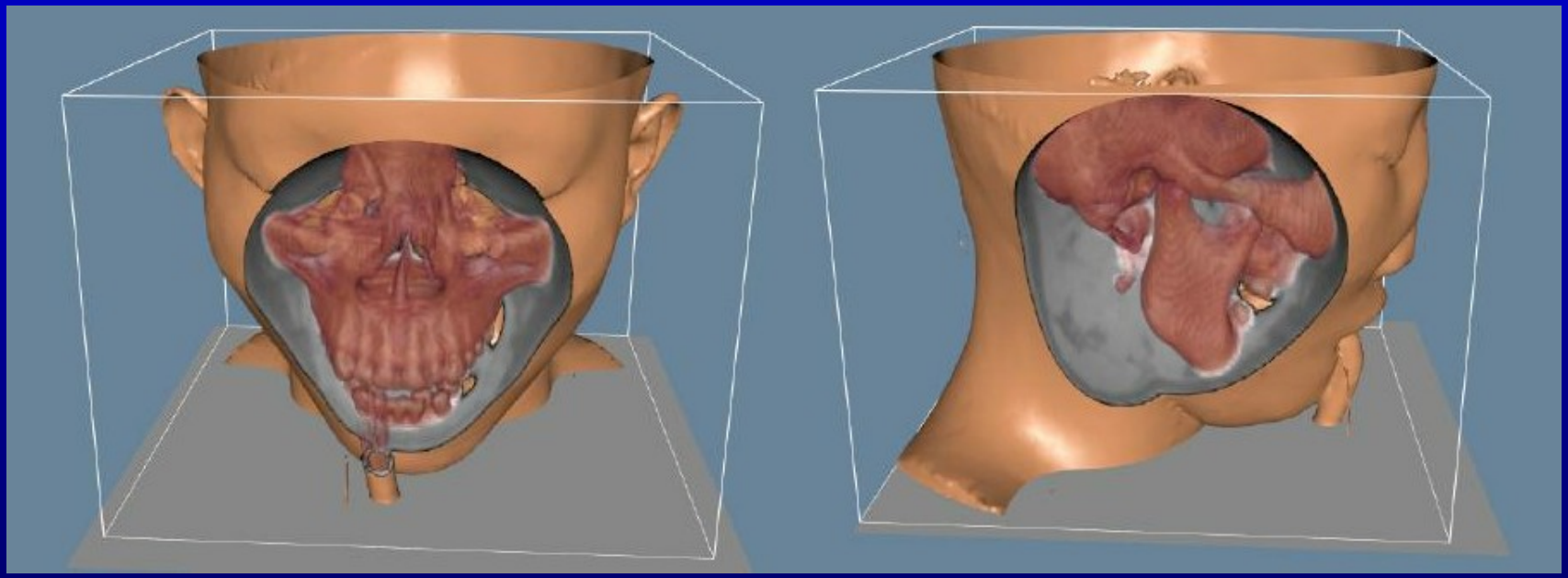
- Java Wars
  - Sun vs Microsoft
- Java vs C++
  - Portability vs Performance
- Java3D
  - Sun vs the world

We need strategies to protect our software investment

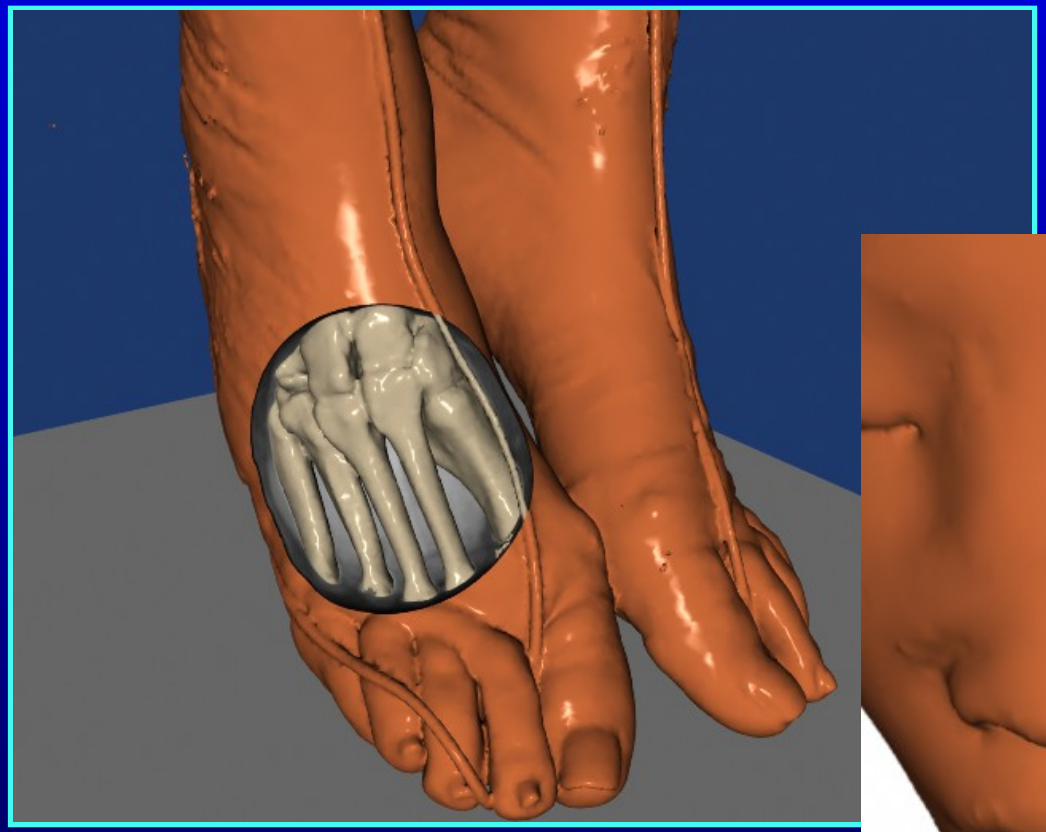
Visualization 1998 and Beyond

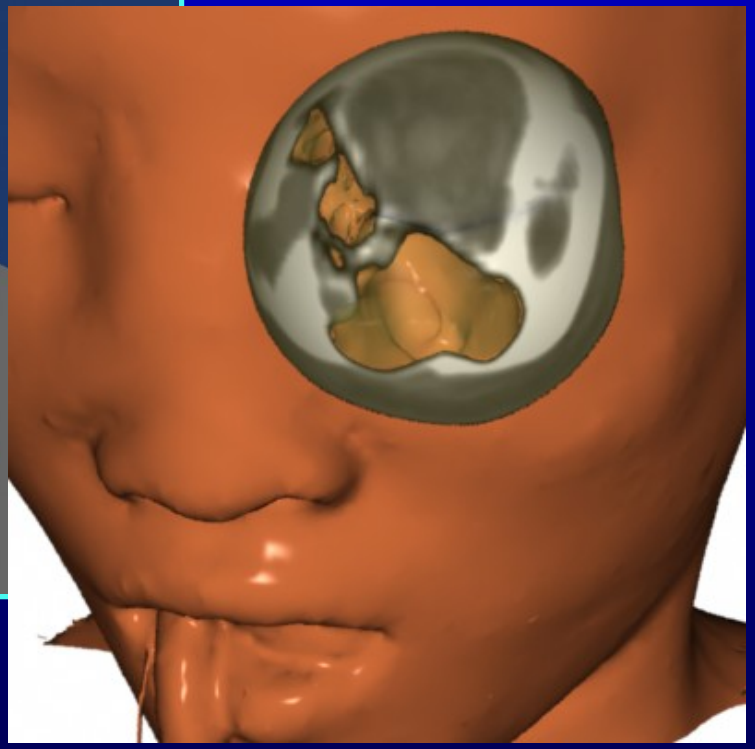
- Hybrid surface / volume rendering
- Visualization components
  - Even higher, richer abstractions
- Information visualization
  - More abstract information
  - Space/Time, Uncertainty
- Embedded Visualization
  - Vis is just a piece of the puzzle

Surface and Volume Rendering



Tissue Lens





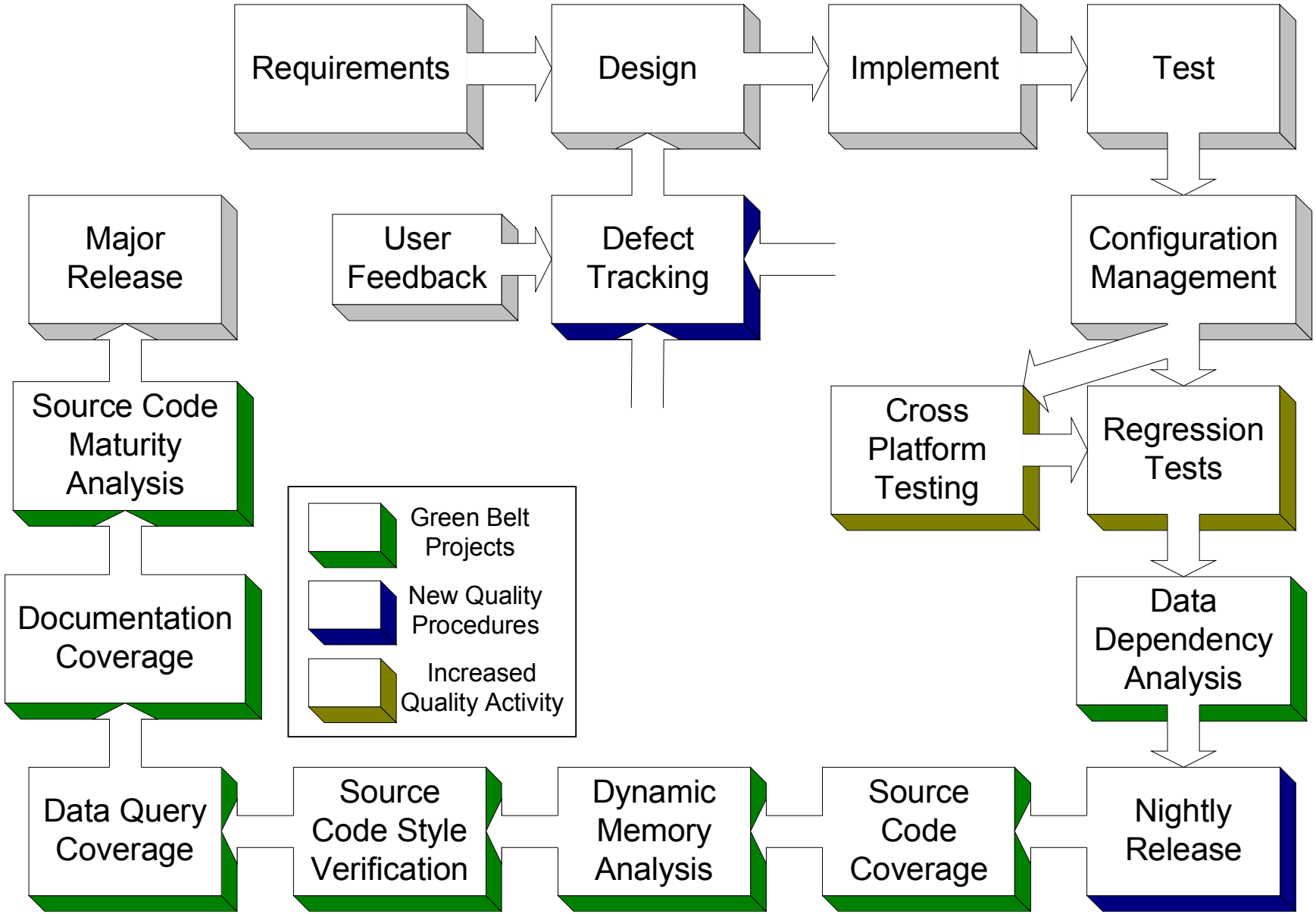
Multi-Modality Fusion

Spatio-Temporal Visualization

User Support and Software Quality

- Support
  - Mailing List
  - Bug Tracking
  - Nightly Releases
- Quality
  - Regression Testing
  - Cross Platform Builds
  - Coverage Testing

Visualization Toolkit (VTK) Software Process



14 Years of OO Visualization

- Revolutionary changes in hardware
- Mature methodologies and languages

- But the drivers have changed
- And, systems are getting more complex, multi-disciplinary