The Visualization Pipeline

CS 5630 / 6630
Fall 2009

The Application Visualization System: A Computational Environment for Scientific Visualization

Craig Upson, Thomas Faulhaber, Jr., David Kamins, David Laidlaw, David Schlegel, Jeffrey Vroom, Robert Gurwitz, Andries van Dam

Stellar Computer

IEEE Computer Graphics & Applications

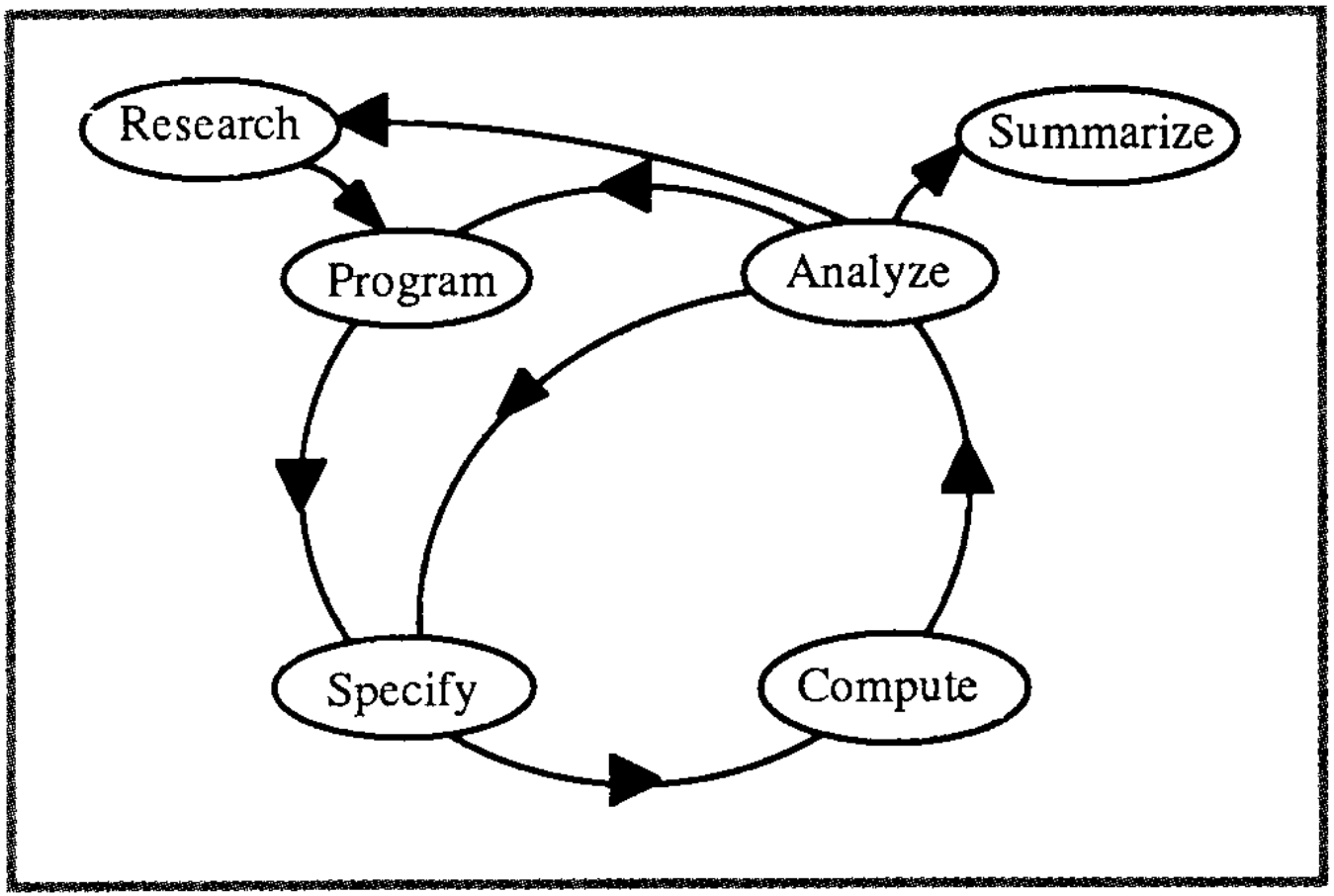


Figure 1. Computational cycle.

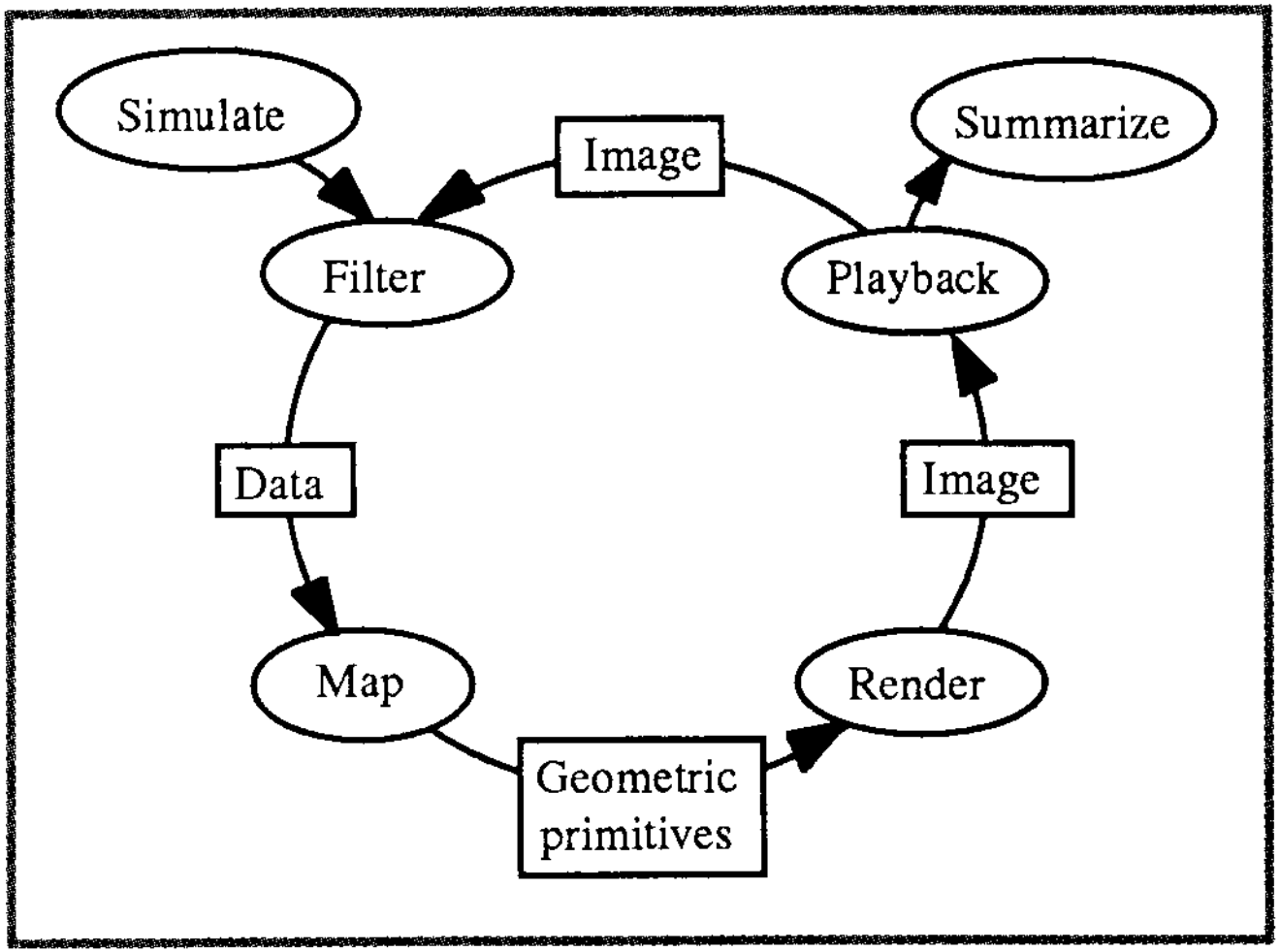


Figure 2. Analysis cycle.

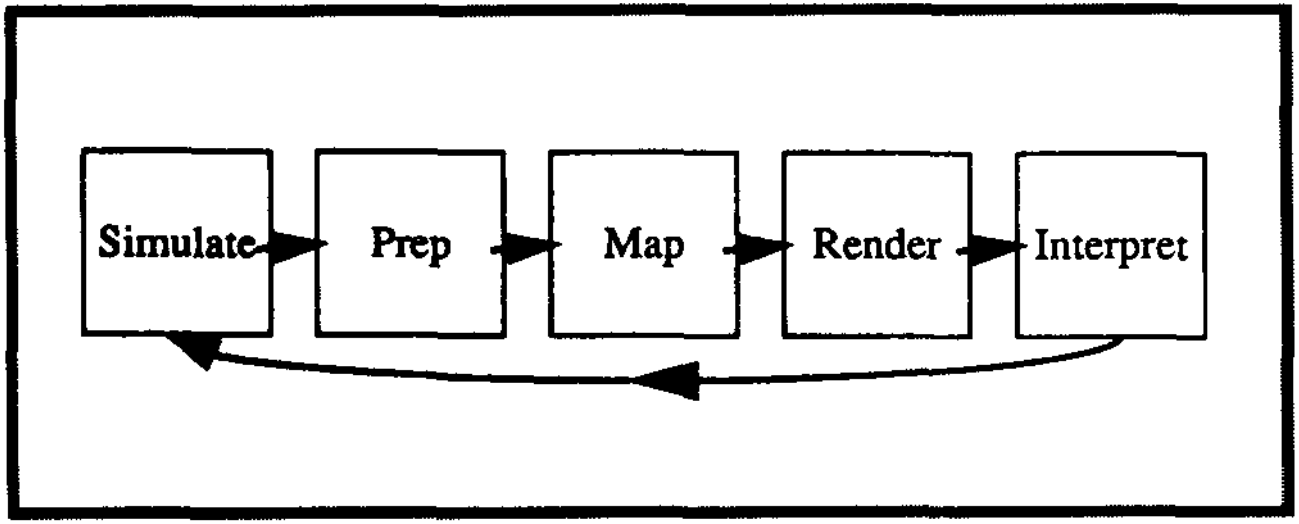


Figure 1. The visualization pipeline.

A Dataflow Toolkit for Visualization

D. Scott Dyer
Ohio Supercomputer Center

IEEE Computer Graphics & Applications

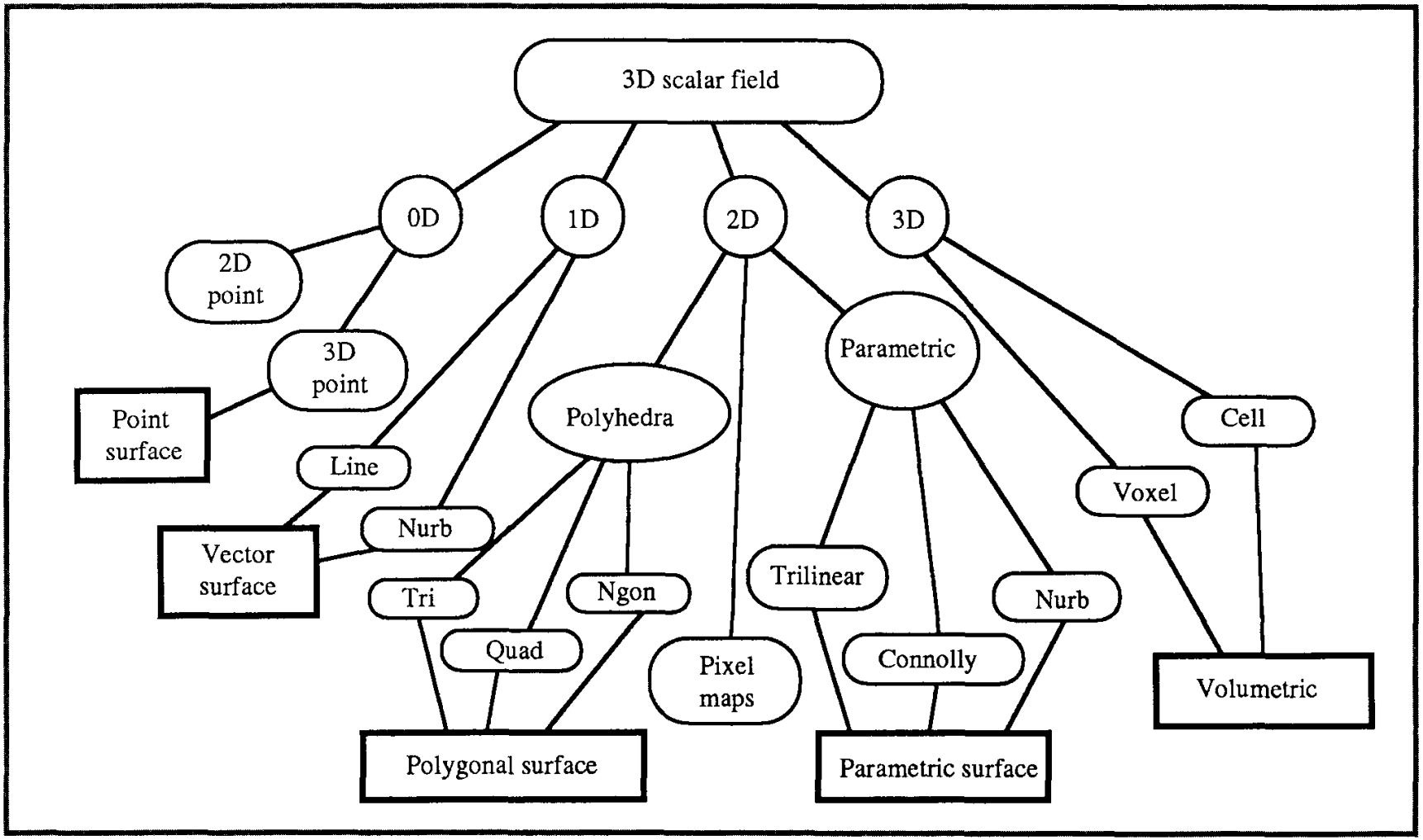


Figure 3. Mapping approaches for 3D scalar fields.

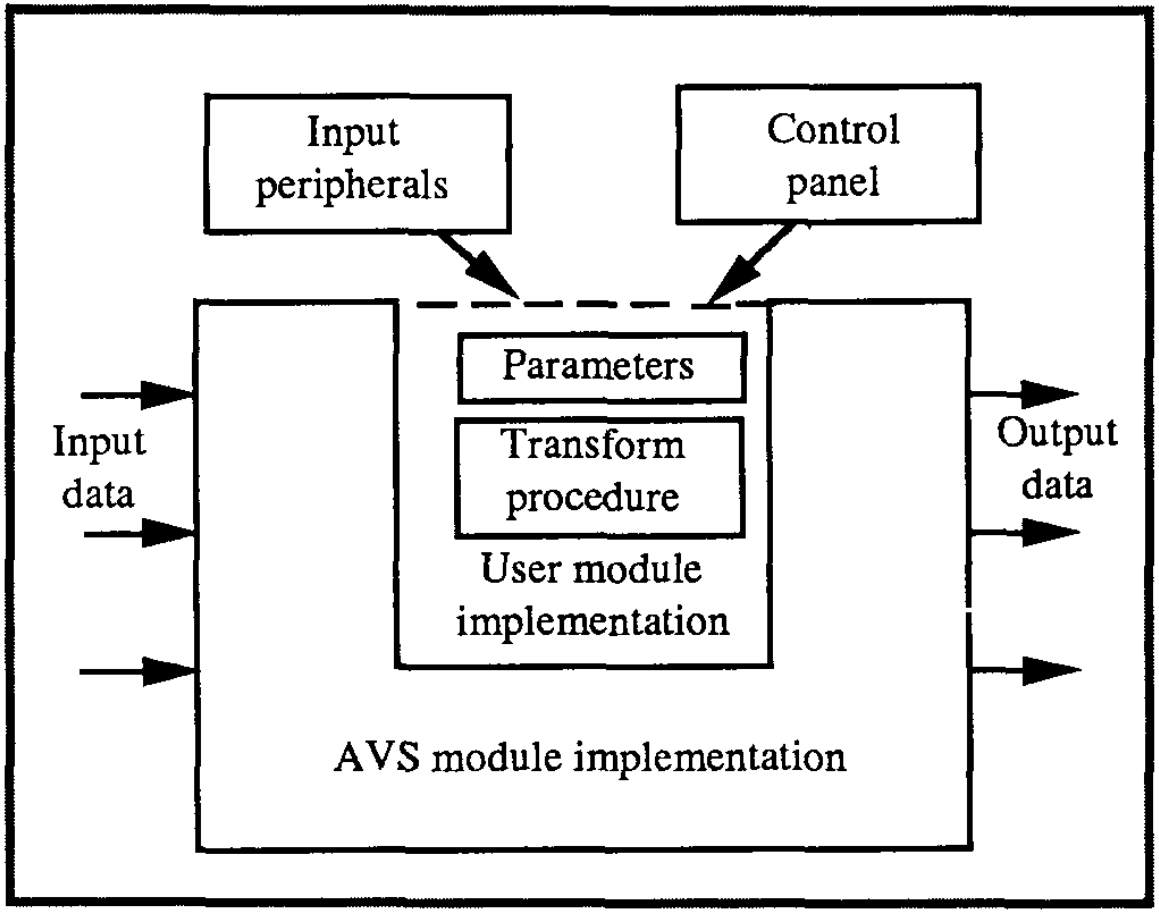


Figure 4. Conceptual model of a module.

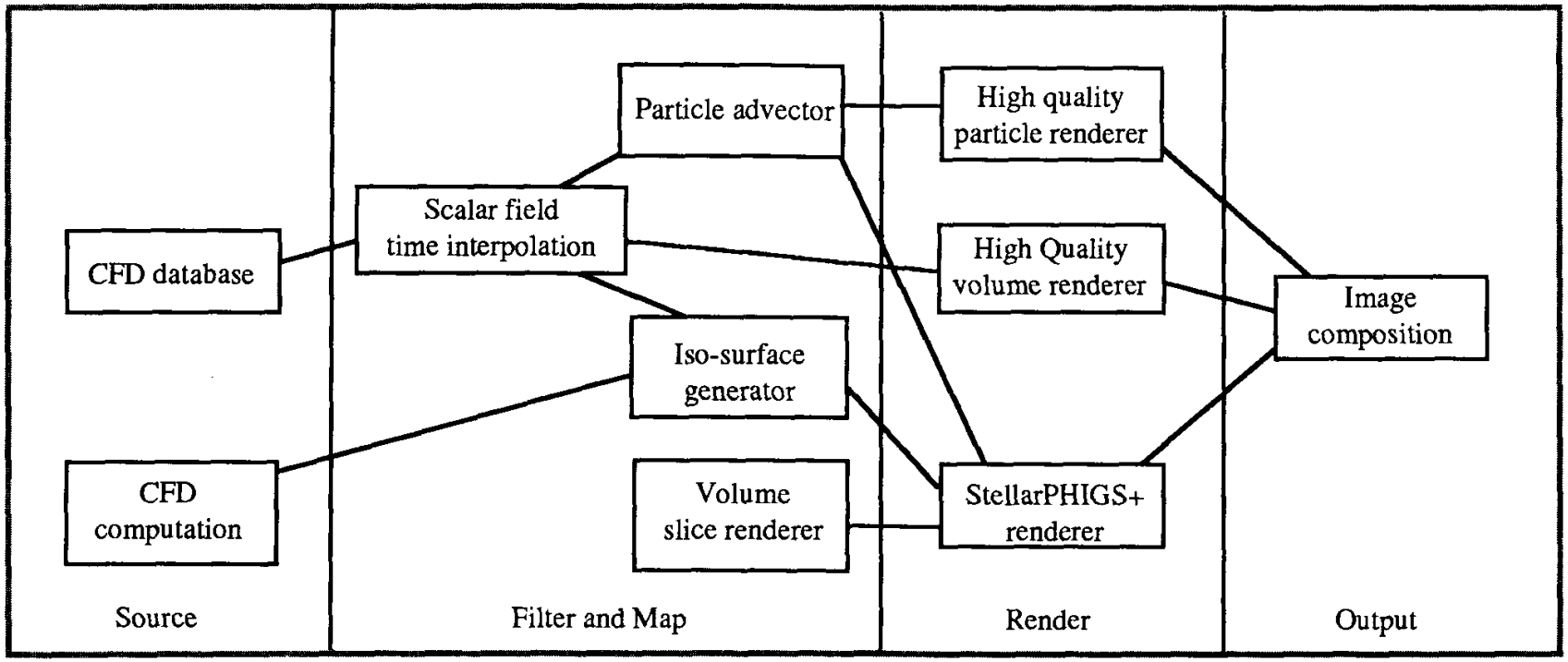


Figure 9. Example of a computational flow network.

Computer Graphics, Volume 22, Number 4, August 1988

ConMan: A Visual Programming Language for Interactive Graphics

Paul E. Haeberli

Silicon Graphics, Inc.
Mountain View, CA 94043

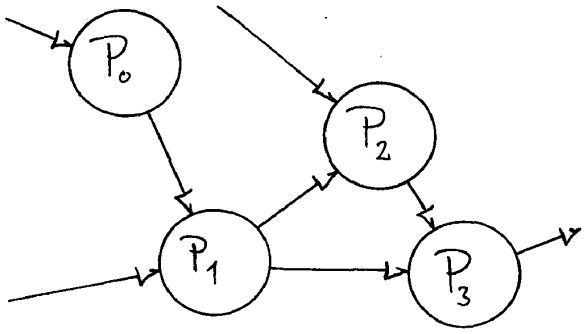


Figure 1. A directed graph representation.

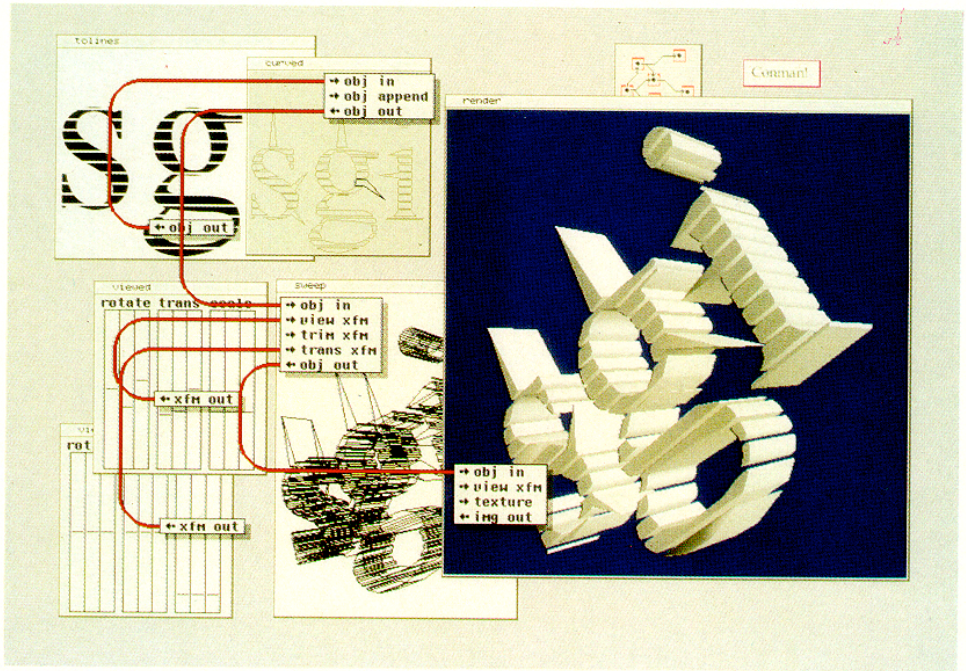


Figure 8. Extracting geometry from an image to make an extruded logo.

Building Block Shaders

Gregory D. Abram and Turner Whitted

Numerical Design Ltd.
Chapel Hill, NC

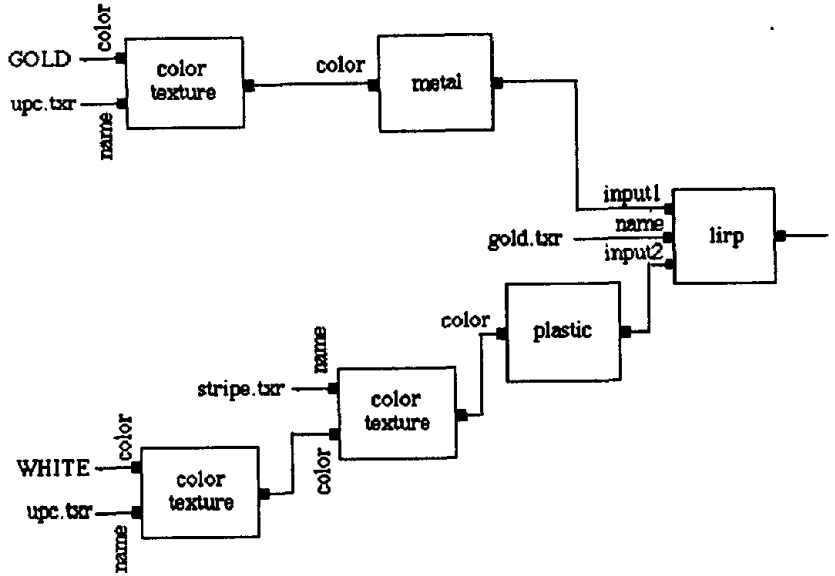
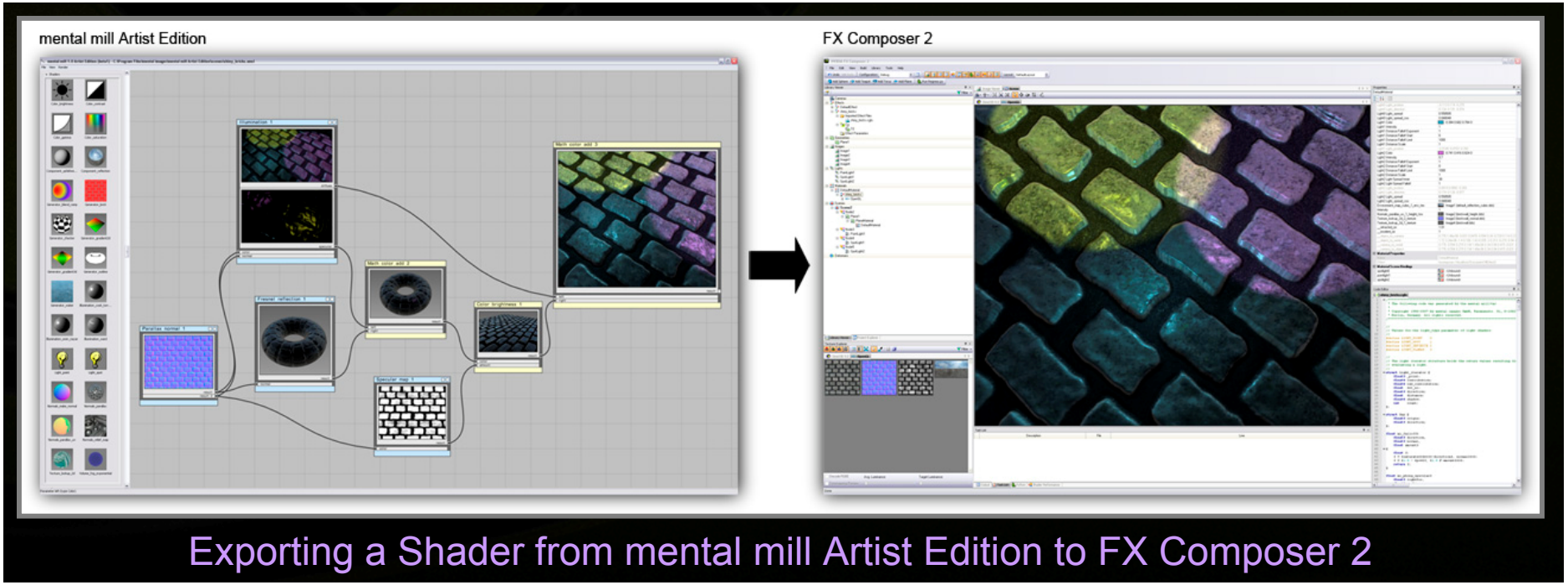


Figure 9. The Cola-Cola can shader.

Figure 8. Two Cola-Cola cans.

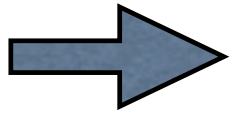


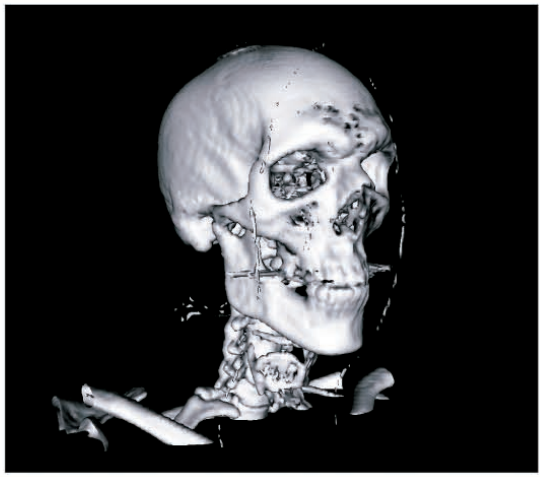
Exporting a Shader from mental mill Artist Edition to FX Composer 2

Visualizing with VTK: A Tutorial

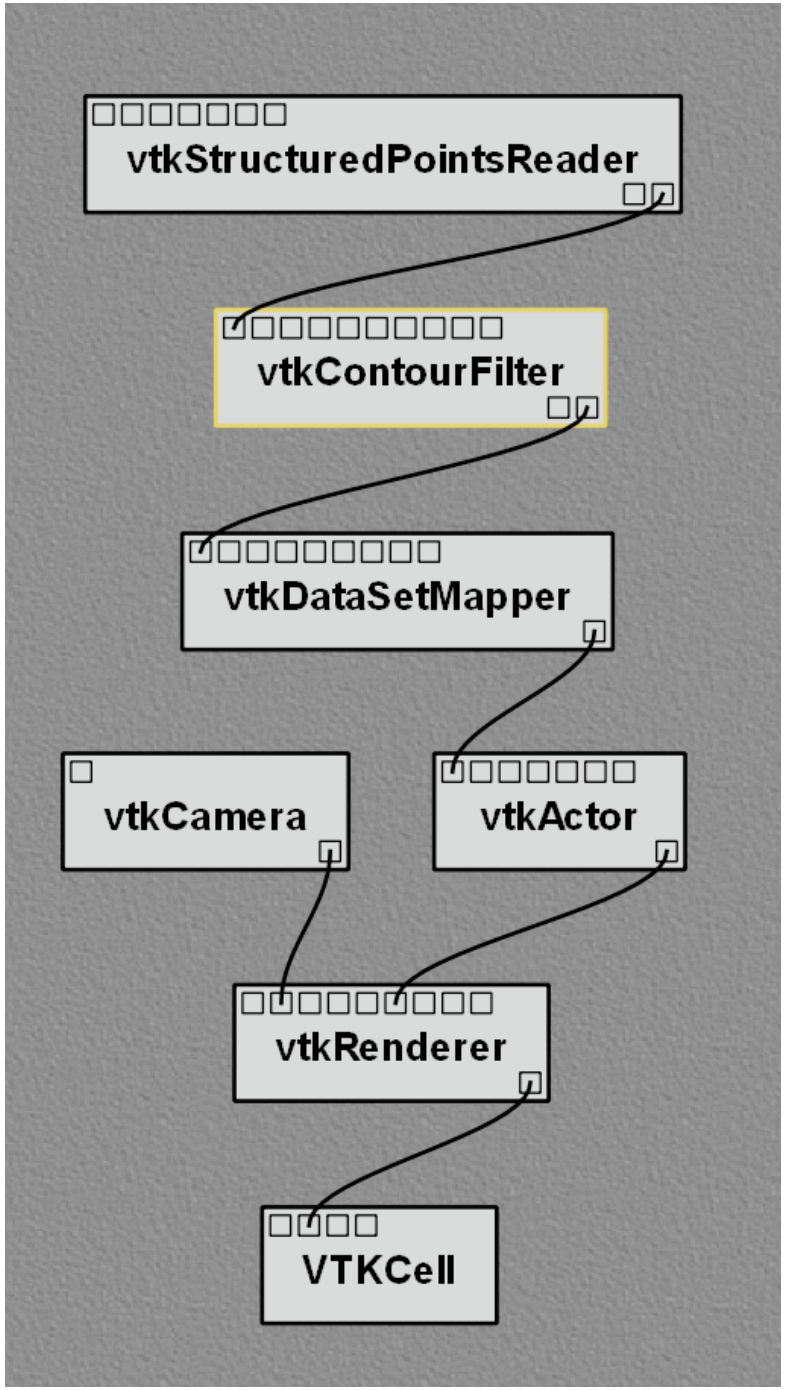
William J. Schroeder, Lisa S. Avila, and William Hoffman
Kitware

```
import vtk

data = vtk.vtkStructuredPointsReader()
data.SetFileName(../examples/data/head.120.vtk)

contour = vtk.vtkContourFilter()
contour.SetInput(data.GetOutput())
contour.SetValue(0, 67)

mapper = vtk.vtkPolyDataMapper()
mapper.SetInput(contour.GetOutput())
mapper.ScalarVisibilityOff()

actor = vtk.vtkActor()
actor.SetMapper(mapper)

cam = vtk.vtkCamera()
cam.SetViewUp(0,0,-1)
cam.SetPosition(745,-453,369)
cam.SetFocalPoint(135,135,150)
cam.ComputeViewPlaneNormal()

ren = vtk.vtkRenderer()
ren.AddActor(actor)
ren.SetActiveCamera(cam)
ren.ResetCamera()

renwin = vtk.vtkRenderWindow()
renwin.AddRenderer(ren)

style = vtk.vtkInteractorStyleTrackballCamera()
iren = vtk.vtkRenderWindowInteractor()
iren.SetRenderWindow(renwin)
iren.SetInteractorStyle(style)
iren.Initialize()
iren.Start()
```





```
import vtk

data = vtk.vtkStructuredPointsReader()
data.SetFileName(../examples/data/head.120.vtk)

contour = vtk.vtkContourFilter()
contour.SetInput(data.GetOutput())
contour.SetValue(0, 67)

mapper = vtk.vtkPolyDataMapper()
mapper.SetInput(contour.GetOutput())
mapper.ScalarVisibilityOff()

actor = vtk.vtkActor()
actor.SetMapper(mapper)

cam = vtk.vtkCamera()
cam.SetViewUp(0,0,-1)
cam.SetPosition(745,-453,369)
cam.SetFocalPoint(135,135,150)
cam.ComputeViewPlaneNormal()

ren = vtk.vtkRenderer()
ren.AddActor(actor)
ren.SetActiveCamera(cam)
ren.ResetCamera()

renwin = vtk.vtkRenderWindow()
renwin.AddRenderer(ren)

style = vtk.vtkInteractorStyleTrackballCamera()
iren = vtk.vtkRenderWindowInteractor()
iren.SetRenderWindow(renwin)
iren.SetInteractorStyle(style)
iren.Initialize()
iren.Start()
```



VTK Graphical Model

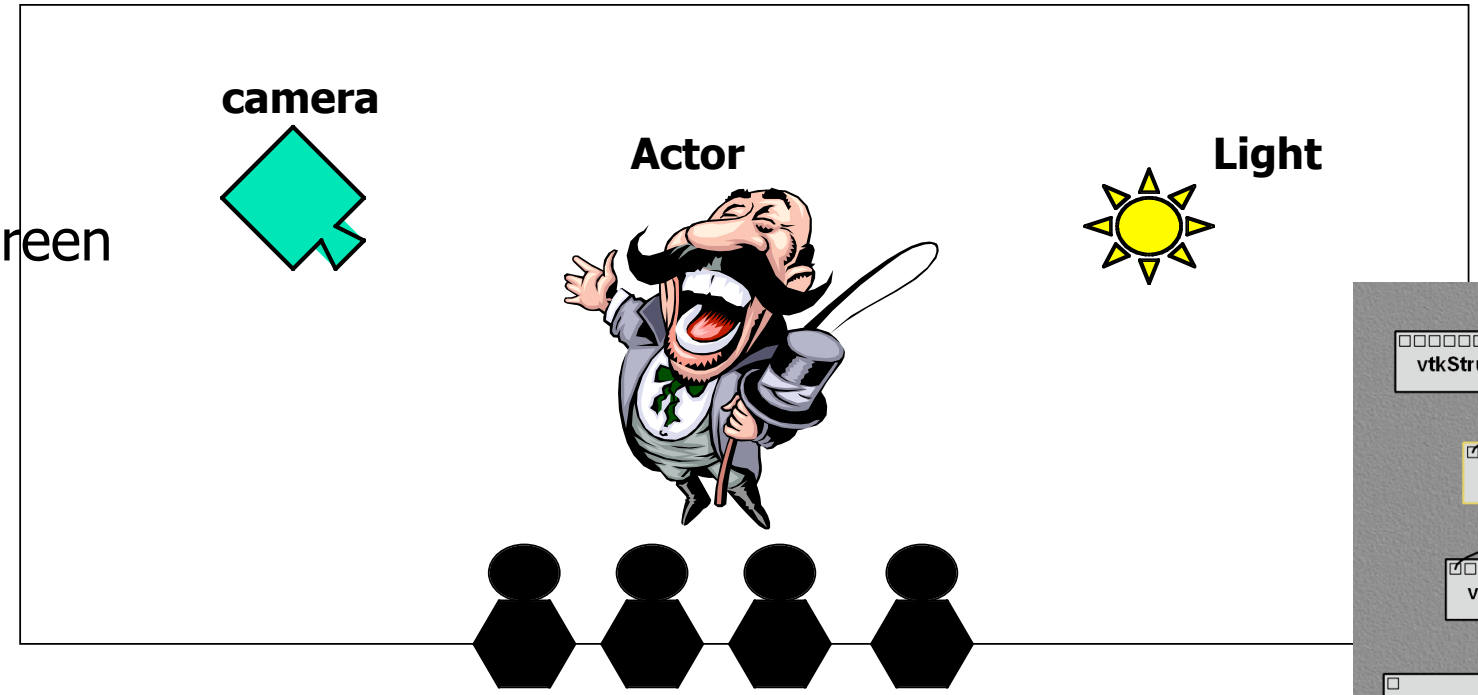
- Render Windows: The object which manages a window on the display device.
- Renderers: The object which coordinates the lights, cameras, and actors of the scene and draws them into the render window.
- Props: The objects added to the renderers to create a scene. The props are the things that you see in the scene.
- Mappers: The object that refer to an input data object and knows how to transform and render it.
- Properties: The object that contains rendering parameters such as color and material properties.

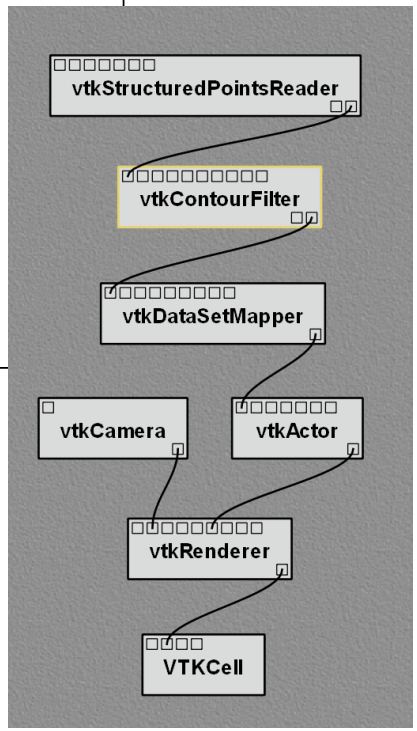
VTK Object Types

- Process Objects: The sources, filters, and mapper algorithms that manipulate the data.
- Data Objects: The datasets that define the dataflow through the network.

The Graphics Model

The purpose is to render the geometry (volume) on the screen





Slide from Dr. Chuck Hansen

User interaction

- vtkRenderWindowInteractor – allow the user to interact with the graphics objects
- Try the following keypresses:

w: wireframe mode

s: surface mode

r: reset the transformation

3: toggle stereo

button 3: zoom; botton 2: pan; button1: rotate;

c/o: camera mode or object mode

j/t: joy stick or tracer ball mode

e: exit

Slide from Dr. Chuck Hansen

Dataflow Programming with VisTrails

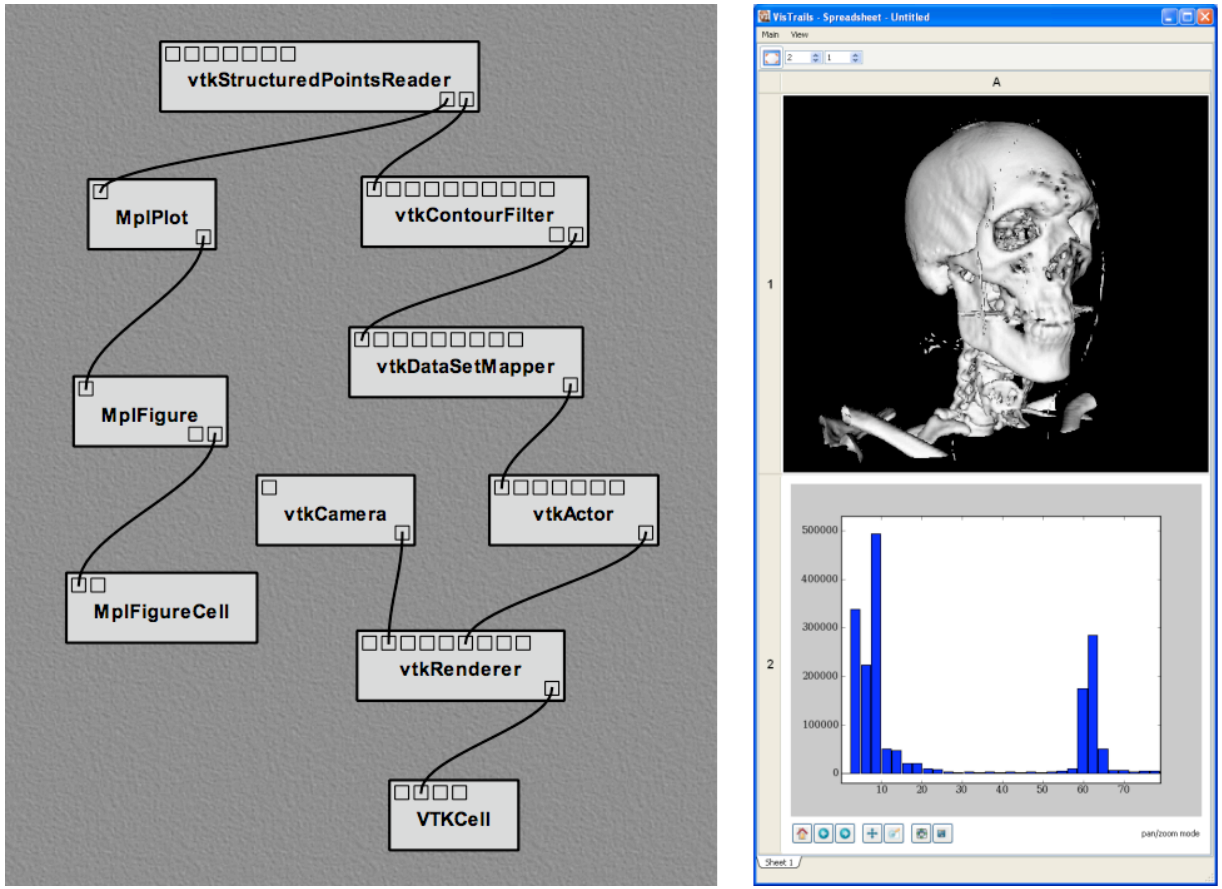
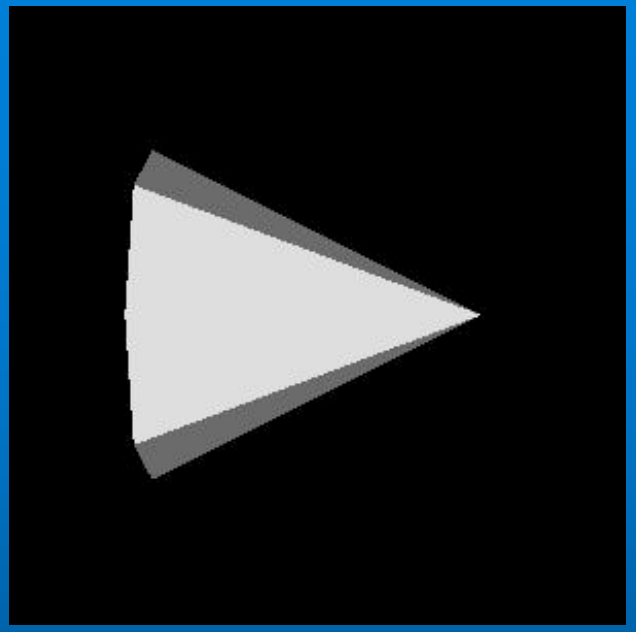


Figure 3.3: Multiple libraries are combined to create two seperate visualizations of the same data. The histogram uses Matplotlib.

Cone.tcl



```
catch {load vtktcl}
# user interface command widget
source ../../examplesTcl/vtkInt.tcl

# create a rendering window and renderer
vtkRenderer ren1
vtkRenderWindow renWin
    renWin AddRenderer ren1
vtkRenderWindowInteractor iren
    iren SetRenderWindow renWin

# create an actor and give it cone geometry
vtkConeSource cone
  cone SetResolution 8
vtkPolyDataMapper coneMapper
  coneMapper SetInput [cone GetOutput]
vtkActor coneActor
  coneActor SetMapper coneMapper

# assign our actor to the renderer
ren1 AddActor coneActor

# enable user interface interactor
iren SetUserMethod {wm deiconify .vtkInteract}
iren Initialize

# prevent the tk window from showing up then start the event loop
wm withdraw .
```

Slide by Dr. Penny Rheingans