计算机辅助手术讲座（5）

Image Guided Surgery (5)

VTK and ITK

顾 力栩 (***Lixu Gu***)

上海交通大学 Med-X研究院

2009.12

What is VTK

Introducing VTK

- The Visualization ToolKit (VTK) is an open source, freely available software system for 3D computer graphics, image processing, and visualization used by thousands of researchers and developers around the world.
- VTK consists of a C++ class library, and several interpreted interface layers including Tcl/Tk, Java, and Python. Professional support and products for VTK are provided by Kitware, Inc.

Introducing VTK

- VTK supports a wide variety of visualization algorithms including:
  - scalar, vector, tensor, texture, and volumetric methods;
  - advanced modeling techniques such as implicit modeling, polygon reduction, mesh smoothing, cutting, contouring, etc.

Introducing VTK

- dozens of imaging algorithms have been directly integrated to allow the user to mix 2D imaging / 3D graphics algorithms and data.
- The design and implementation of the library has been strongly influenced by object-oriented principles.
- VTK has been installed and tested on nearly every Unix-based platform, PCs (Windows 98/ME/NT/2000/XP), and Mac OSX Jaguar or later.

Introducing VTK

- The graphics model in VTK is at a higher level of abstraction than rendering libraries like OpenGL. This means it is much easier to create useful graphics and visualization applications.
- In VTK applications can be written directly in C++, Tcl, Java, or Python. In fact, using the interpreted languages Tcl or Python with Tk, and even Java with its GUI class libraries, it is possible to build useful applications really, really fast.

Introducing VTK

- **Technical Overview: Software**
  - Over 700 C++ classes
  - 350,000+ lines of C++ code (110,000 executable lines)
  - Designed using the approach of Rumbaugh et al. (Object-Oriented Modelling and Design from Prentice-Hall)
  - 215,000+ lines of automatically generated Tcl wrapper code (similar counts for Python and Java)

Introducing VTK

- **Technical Overview: Software**
  - In-line documentation (both in-code and man pages)
  - Easy to understand C++ code (honest!)
  - Designed to be extensible
  - Lots of examples, applications, test cases, and data
  - Supports portable multithreading and distributed memory for parallel algorithms

Introducing VTK

- **Technical Overview: Interaction and GUI**
  - Integrates seamlessly with a variety of windowing systems including Qt, FLTK, wxWindows, Tcl/Tk, Python/Tk, Java, X11, Motif, Windows, Cocoa and CARBON.
  - Supports a variety of interaction styles including trackball and joystick modes for cameras and actors. Interaction styles can be customized and easily added.

Introducing VTK

- Implements a command/observer event handling mechanism. Objects can watch other objects for a particular event and invoke callbacks as appropriate. Events can be prioritized and aborted for complex event handling. VTK classes define a large palette of events that are invoked throughout the system.
- Includes an extensive set of 3D widgets including point, line, plane, implicit plane, box, sphere, scalar bar, image plane, and spline widgets.

Introducing VTK

- **Technical Overview: 3D Graphics**
  - Surface Rendering
  - Volume Rendering
  - Rendering Primitives
  - Interactive Viewer/Renderer "3D Widgets" for interacting with data

Introducing VTK

- **Technical Overview: Visualization**
  - Data Types:
    - polygonal data (points, lines, polygons, triangle strips)
    - images and volumes (i.e., structured point datasets)
    - structured grids (e.g., finite difference grids)
    - unstructured grids (e.g, finite element meshes)
    - unstructured points
    - rectilinear grids
  - Cell Types:
    - line, poly-line, triangle, triangle strip, pixel, quadrilateral, polygon, voxel, etc.

Introducing VTK

- **Technical Overview: Imaging**
  - Features
    - Uses cached, streaming pipeline so that you can operate on gigantic datasets (i.e., deals with pieces of data). This is done completely transparently.
    - Most imaging filters are multi-threaded for parallel execution
    - Fully integrated with 3D graphics/visualization pipeline

Introducing VTK

- Filter types (a quick summary)
  - diffusion filters
  - Butterworth, low-pass, high-pass filters
  - dilation, erosion, skeleton
  - convolution
  - difference, arithmetic, magnitude, gradient, mean
  - distance
  - FFT, Fourier, Gaussian, Sobel
  - Histogram, threshold
  - Permutation (置换）, conversion （变换）, padding （填充）

What's Cool About VTK

- It's free (although the books help!)
- Easy to create graphics/visualization applications
- C++ source code - you have a lot of control
- Easy to derive new classes
- Can prototype or build applications using "interpretive" languages Tcl, Python, and Java
- User interface can be created fast with Tk or Java GUI class libraries

What's Cool About VTK

- Can learn about graphics / visualization / imaging
- Supports an extensive palette of 3D widgets
- Platform/rendering library independent
- Lots of advanced and very useful algorithms
- Integrated software
- You can convert data into pictures
- Object-oriented
- Heavily tested in real-world applications
- Large user base provides decent support

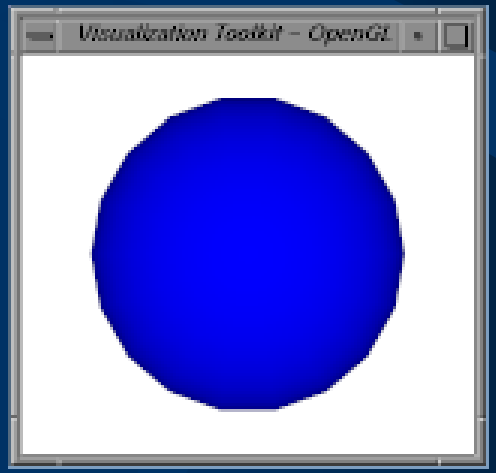
What's UnCool About VTK

- Not a super-fast graphics engine...VTK uses C++ dynamic binding and a device independent graphics model.
- C++ source code (so use Tcl, Python, or Java)
- Very large...not a toy...you'll need a decent system to use it effectively

Getting Start

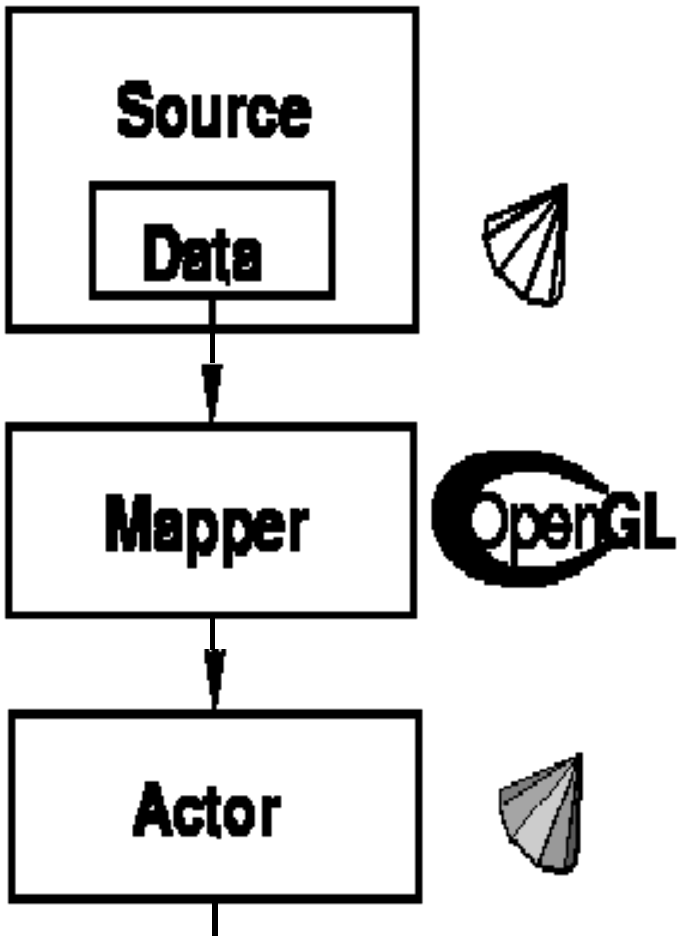
Get The Package Running

- Download Software:
  - From VTK page: http://www.vtk.org
- Install the software in your own computer
- Exercise:
  - Creating a Sphere:



The VTK Pipeline

- You write a VTK program by creating *objects* and joining them together
- Every VTK program has at lease one *pipeline*
- Cone Pipeline Diagram



Either reads the data from a
file or creates the data from
scratch.

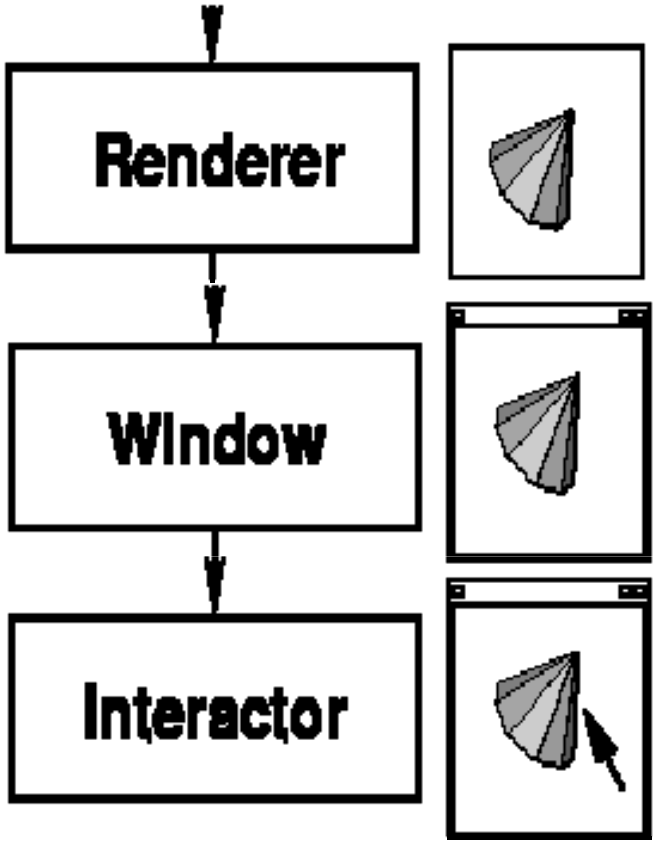
```
from vtkpython import *
cone = vtkConeSource()
cone.SetResolution(10)
```

Moves the data from VTK into OpenGL.

```
coneMapper = vtkPolyDataMapper()
coneMapper.SetInput(cone.GetOutput(
))
```

For setting colors, surface properties, and the position
of the object.

```
coneActor = vtkActor()
coneActor.SetMapper(coneMapper
```



The rectangle of the computer screen that VTK draws into.

```
ren = vtkRenderer()
ren.AddActor(coneActor)
```

The window, including title bar and decorations.

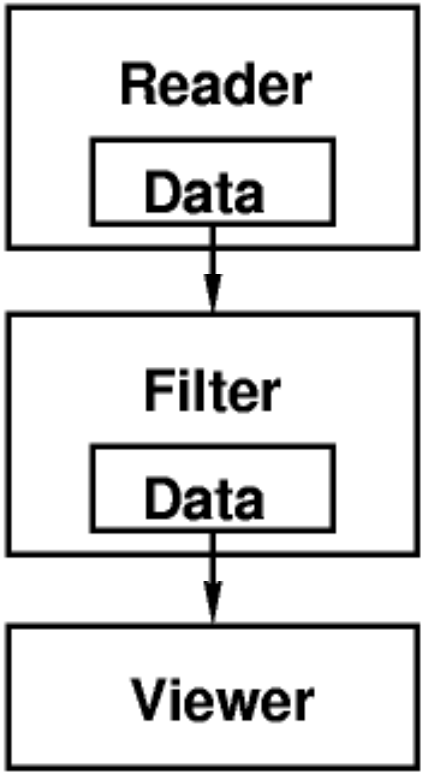
```
renWin = vtkRenderWindow()
renWin.SetWindowName("Cone")
renWin.SetSize(300,300)
renWin.AddRenderer(ren)
```

Allows the mouse to be used to interact wth the data.

```
iren = vtkRenderWindowInteractor()
iren.SetRenderWindow(renWin)
iren.Initialize()
iren.Start()
```

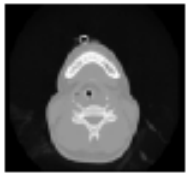
VTK Pipeline

- Image Pipeline Diagram



Reads the data from a file.

```
from vtkpython import *
reader = vtkBMPReader()
reader.SetDataSpacing(1.0, 1.0, 2.0)
reader.SetDataOrigin(0.0, 0.0, 0.0)
reader.SetFileName("image.bmp")
```



Processes the image (enlarge the image, in this case).

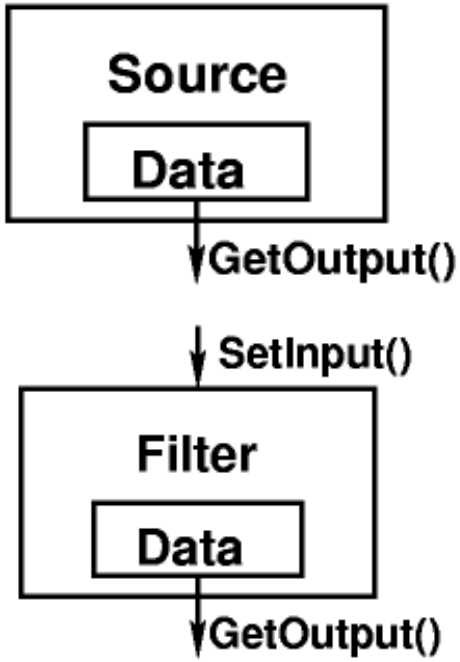
```
resize = vtkImageReslice()
resize.SetInput(reader.GetOutput())
resize.SetOutputSpacing(0.25, 0.25,
2.0)
resize.SetInterpolationModeToCubic()
```

Viewer (has an ImageMapper, Actor2D, and ImageWindow inside).

```
viewer = vtkImageViewer()
viewer.SetInput(resize.GetOutput())
viewer.SetColorWindow(255)
viewer.SetColorLevel(127.5)
viewer.Render()
```

VTK Pipeline

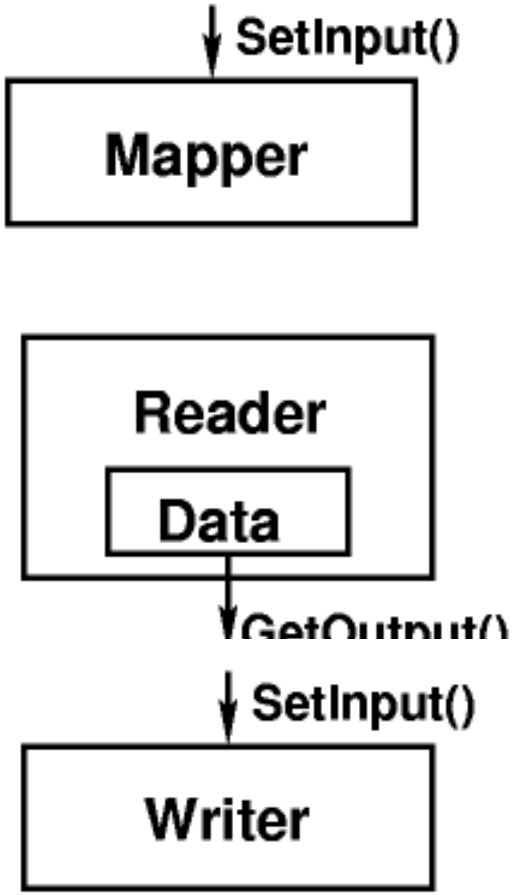
- You write a VTK program by creating ***objects*** and joining them together
- The primary pipeline object types are as follows



vtkConeSource::SetHeight(5.0)
vtkSphereSource::SetRadius(2.0)

vtkImageMapToColors::SetLookupTable(table)
vtkMarchingCubes::SetValue(0,1043)
vtkTransformPolyDataFilter::SetTransform(transform)

VTK Pipeline



All Mappers have a Render() method.
The vtkImageMapper has two important methods:
vtkImageMapper::SetColorWindow(255.5)
vtkImageMapper::SetColorLevel(127.5)

All Readers have a SetFileName("filename") method

All Writers have a SetFileName("filename") method and a Write() method.

Pipeline Execution

- The pipeline doesn't compute anything until:
  - the Write() method of the Writer object at the end of the pipeline is called
  - the Render() method of a Window associated with the pipeline is called
  - you drag the mouse in an Interactor associated with the pipeline

Pipeline Execution

- A VTK Filter can never "force" any changes to occur further down the pipeline. It has to wait until it is requested to Update, then it can only do the following:
  - request an Update of its Input data
  - create new data (the Output) from the Input data

Pipeline Execution

- Data does not "flow" along the pipeline. Each filter has its own copy of data, which it creates from its Input data.
- If you have one Source and two Filters, you will have three copies of the data (yup, VTK can be a memory hog). You can force a filter to delete its data once the next filter is done with it.

What is ITK

What is ITK?

- ITK is an open-source software toolkit for performing registration and segmentation
- ITK is designed for medical application, although it is capable of processing other types
- ITK is implemented in C++. In addition, an automated wrapping process will help it support other programming languages such as Tcl, Java, and Python in the future.

What are ITK's origins?

- NLM (the US National Library of Medicine) and other six principle organizations
- Three commercial partners: GE Corporate R&D, Kitware, and Insightful
- Three academic partners University of North Carolina (UNC), University of Tennessee (UT), and University of Pennsylvania (UPenn)

The Design Features

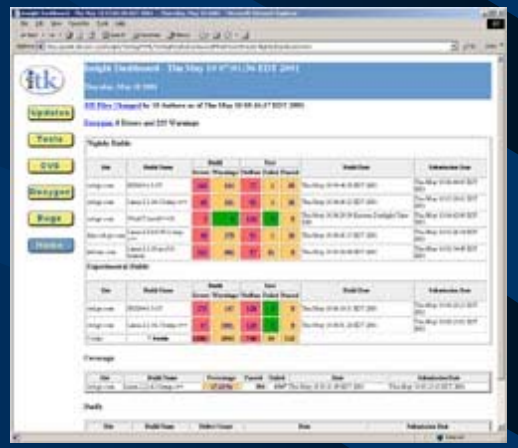
- ITK provides data representations in general form for images (arbitrary dimension) and (unstructured) meshes.
- ITK does not address visualization or graphical user interface as well as tools for file interface.
- ITK supports Multi-threaded parallel processing

The Implementation Features

- ITK is cross-platform (Unix and Windows).
- ITK is implemented using generic programming principles. Heavily templated C++ code challenges the MSVC, Sun, gcc, and SGI compilers.
- "Smart pointers": maintain a reference count to objects and auto-delete objects when they out of scope.
- Cmake: uses configure and make on Unix and generates projects and workspaces in Windows.

ITK Implementation Statues

- Itk project is at its start point. Most of its classes are under developing and testing stage. But most released Classes are stable.
- Today's testing dashboard:
  - There still have lots of errors and warnings when build and test it
  - Most errors and warnings are occurred at the example parts. Basic classes are relatively stable.



What does ITK contain?

- ITK seeks to provide a class hierarchy that directly supports specific segmentation and registration taxonomies and yet does not limit the exploration of novel methods.
- Registration algorithm:
  - Metrics:
    - Mutual Information
    - Landmark Distance
  - Transformations
    - Affine, Rigid, or Projective
    - Kernel-based (e.g., Elastic-Body Splines and Thin-Plate Splines)
  - Optimization Algorithms
    - Conjugate Gradient
    - Gradient Descent

What does ITK contain?

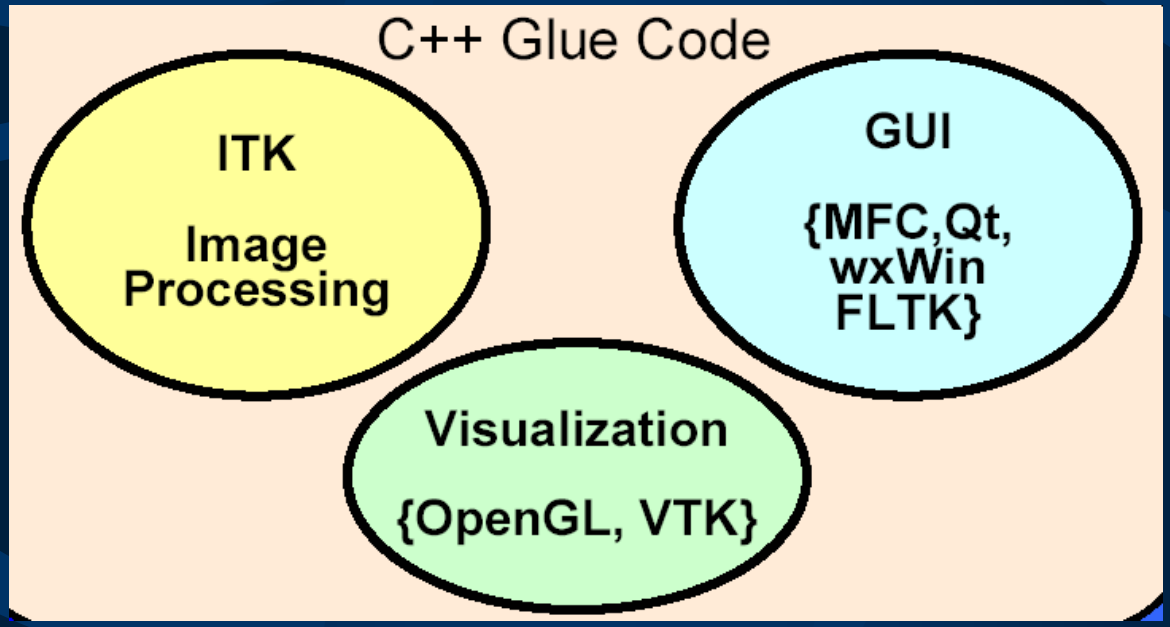
- Segmentation Algorithms:
  - Deformable mesh
  - Balloon force filter
  - Region growing
  - Watershed
- Image Processing Algorithms:
  - Contrast Enhancement (Power-law Adaptive Histogram Equalization)
  - Morphology Image filters

What's the effect to us?

- Use well established code source from multiple institutions.
- Reduce our efforts in code developing
- Establish a foundation for our research.
- Create a repository of fundamental algorithms.
- Develop a platform for advanced product development.
- Create conventions for our work.

What ITK Has Not?

- No Graphic User Interface
- No Visualization



What Do I need?

C++ Compiler

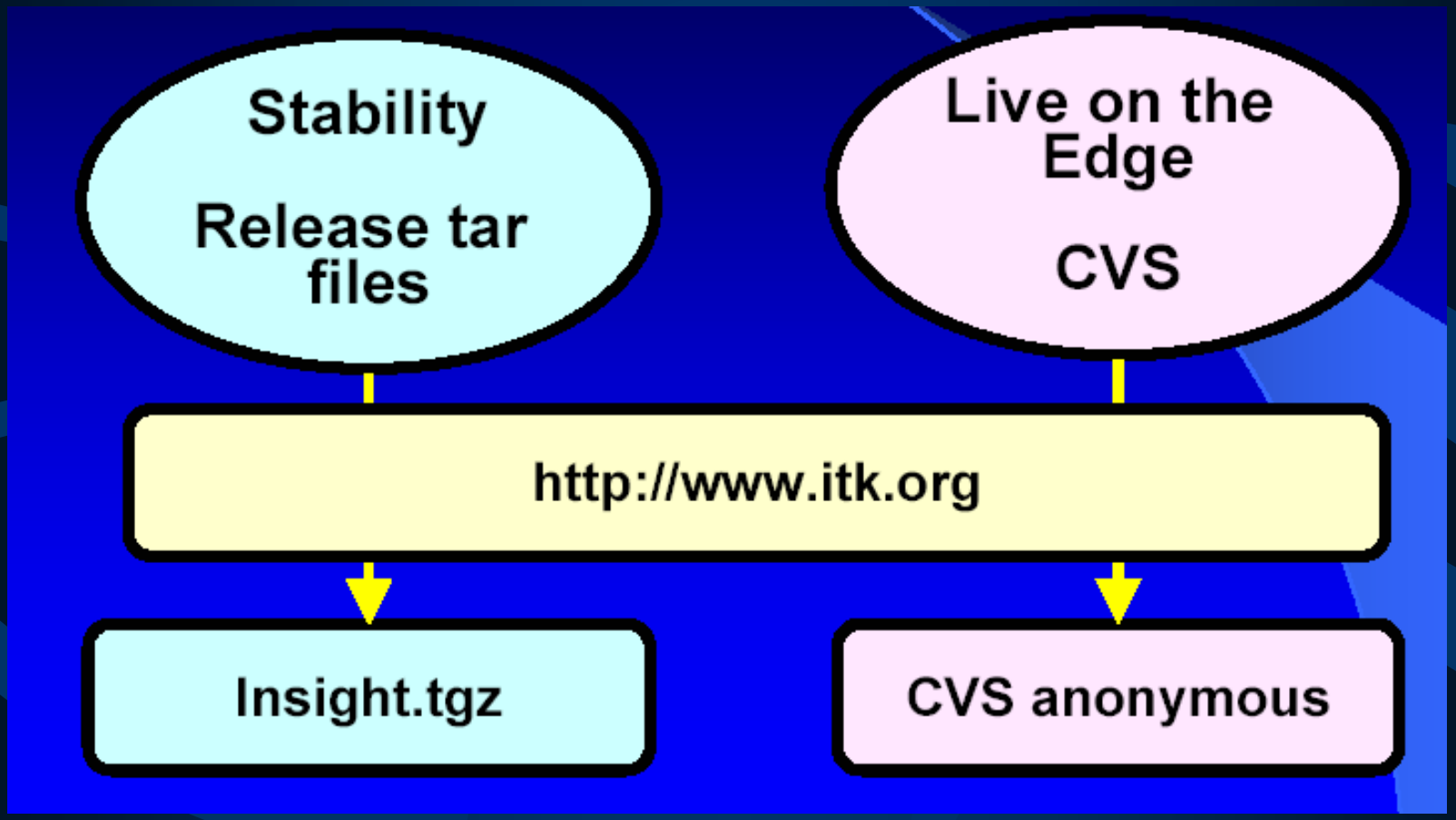
gcc 2.95 – 3.2
Visual C++ 6.0
Visual C++ 7.0
VC++ 7 2003
Intel 5.0
IRIX CC
Borland 5.0
Mac - gcc

CMake

www.cmake.org

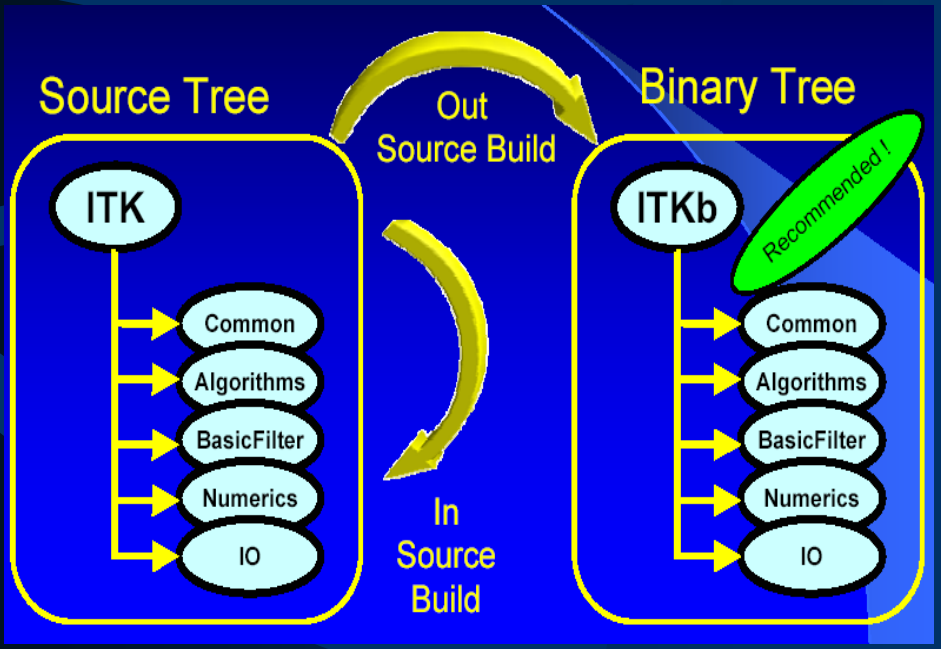
Get Start

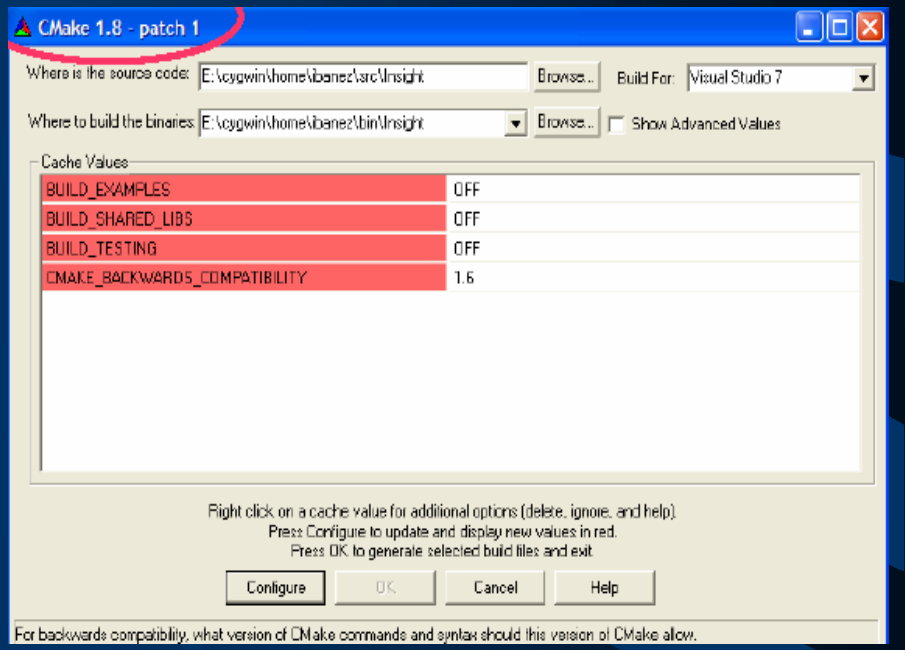
- Setp 1: Download ITK and Cmake



Get Start

- Step 2: Configure ITK



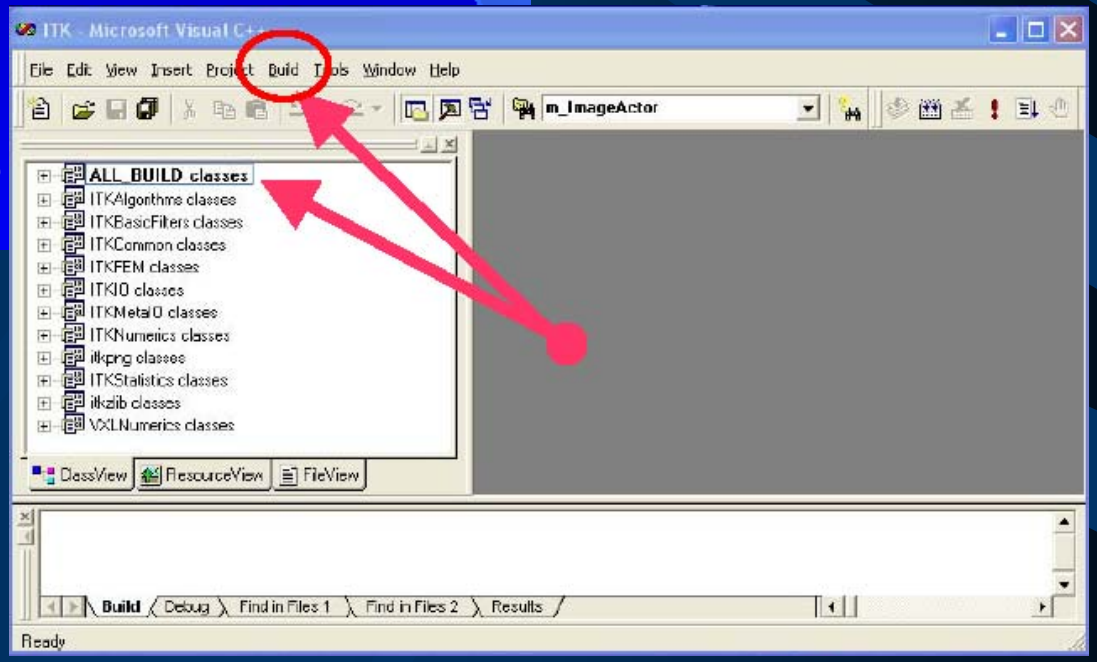


Get Start

- Step 3: Build ITK

  - Open ITK.dsw in the Binary Directory
  - Select ALL_BUILD project
  - Build it

    …It will take about 15 minutes …



Get Start

- Step 4: Build and verify Object

- Most of ITK classes are C++ Templates
- Basic libraries are small

  they only contain non-templated classes
- Basic libraries are built in about 15 min

The following libraries should be there

- ITKCommon
- ITKBasicFilters
- ITKAlgorithms
- ITKNumerics
- ITKFEM
- ITKIO
- ITKStatistics
- ITKMetaIO
- itkpng
- itkzlib

Get Start

- Start your own project:
  - Create a clean new directory
  - Write a CmakeLists.txt file
  - Write a simple .cxx file
  - Configure with Cmake
  - Build
  - Run

```
PROJECT( myProject )

FIND_PACKAGE ( ITK )
IF ( ITK_FOUND )
      INCLUDE( ${USE_ITK_FILE} )
ENDIF( ITK_FOUND )

ADD_EXECUTABLE( myProject   myProject.cxx )

TARGET_LINK_LIBRARIES ( myProject  ITKCommon   ITKIO)
```

Get Start

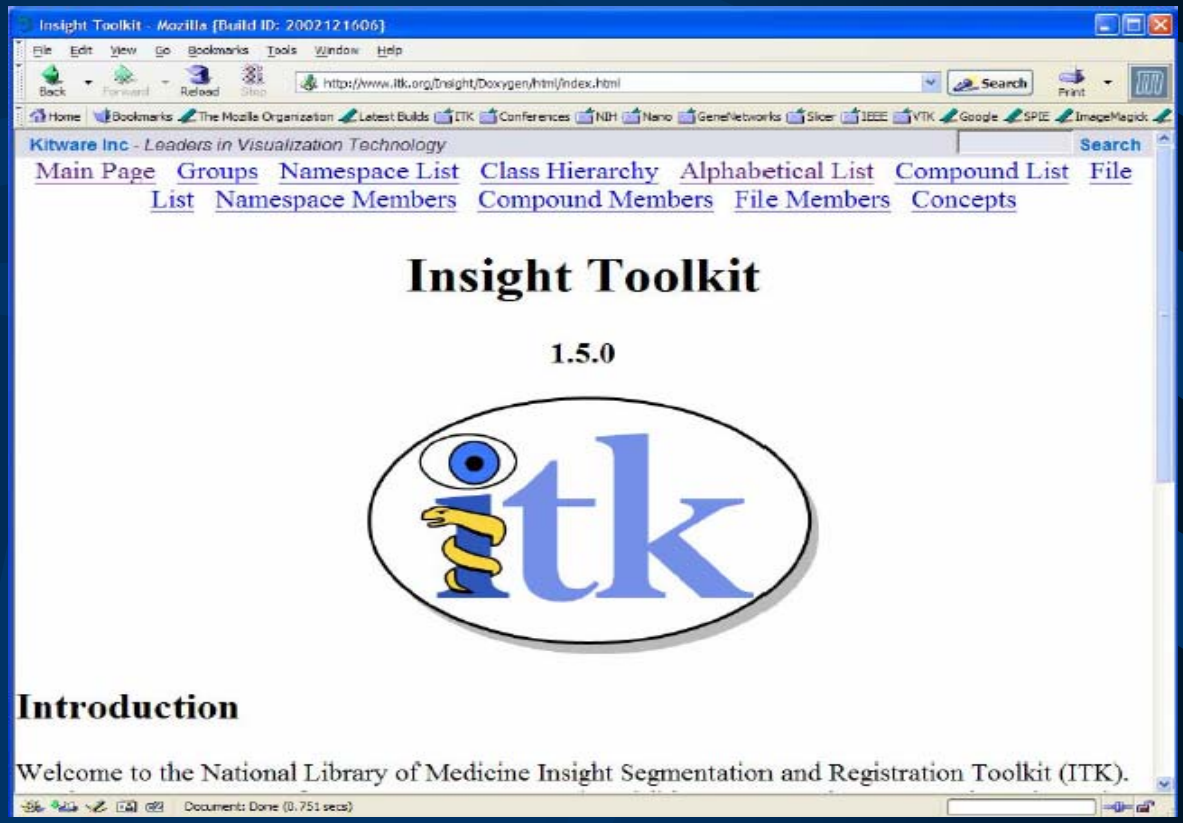
- Writing myProject.cxx:

```
#include "itkImage.h"
#include "itkImageFileReader.h"
#include "itkGradientMagnitudeImageFilter.h"

int main( int argc, char **argv ) {
  typedef itk::Image<unsigned short,2>          ImageType;
  typedef itk::ImageFileReader<ImageType>       ReaderType;
  typedef itk::GradientMagnitudeImageFilter<
                    ImageType,ImageType>        FilterType;

  ReaderType::Pointer reader = ReaderType::New();
  FilterType::Pointer filter = FilterType::New();

  reader->SetFileName( argv[1] );
  filter->SetInput( reader->GetOutput() );
  filter->Update();
  return 0;
}
```

Get Start

- How to find what you need

  http://www.itk.org/Doxygen/html/Index.html



Threshold In VTK/ITK

- VTK:
  - Fixed thresholding (double thresholds acceptable)
    - vtkImageThreshold: output → image
    - vtkThreshold: output → unstructured grid
    - vtkThresholdPoint: output → polygonal data
- ITK:
  - Fixed thresholding (double threshold acceptable)
    - itkBinaryThresholdImageFilter: output → image
    - itkBinaryThresholdImageFunction: output → Boolean
  - Optimal thresholding:
    - itkOtsuThresholdImageCalculator: output → image

Filters in VTK/ITK

- VTK:
  - vtkImageConvolve()
  - vtkImageGaussianSmooth()
  - vtkImageMedian3D()
  - vtkImageLogic()
  - vtkImageMathematics()
- ITK
  - itk::MeanImageFilter
  - itk::MedianImageFilter
  - itk::DiscreteGaussianImageFilter

Projects

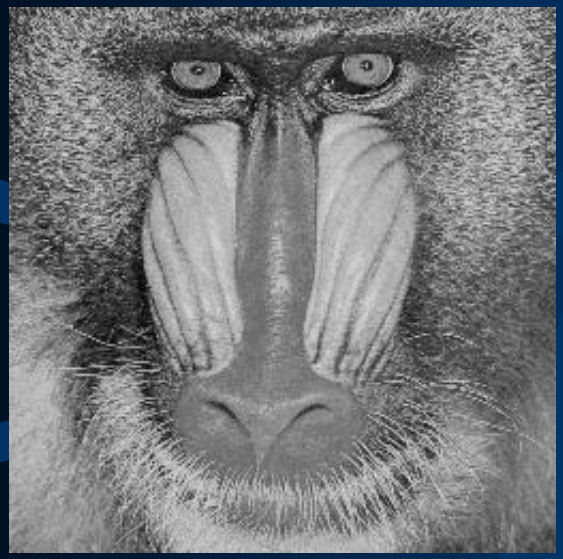
Project -1

- Histogram and threshold:
  - Requirement:
    - Program to realize Histogram analysis and threshold operation
    - Design UI and function buttons
    - Threshold operation can be manual or automatic (Otsu and Entropy)
  - Choose your favorite language

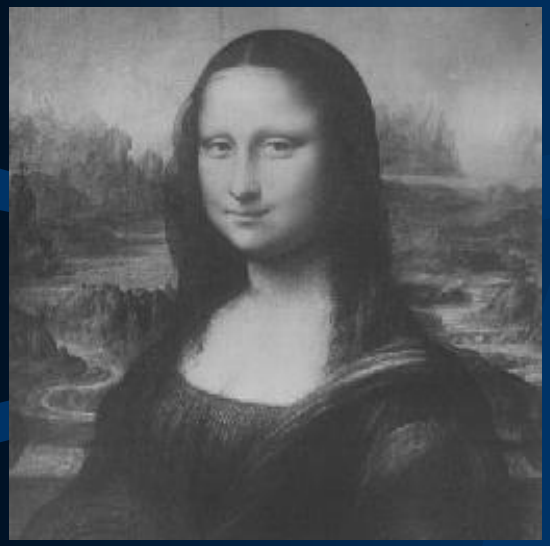
Project - 2

- Convolution and Image Filters
  - Requirement:
    - Program to realize the convolution operation and one of the next filters
      - ✓ Roberts operator; Prewitt operator; Sobel operator;
      - ✓ Gaussian filter and Median filter
    - Design proper UI and result display
    - The edge detection and noise reduction

Classic Image Samples



Classic Image Samples



Discussion