VTK 使用心得

VTK source directory Structure

InfoVis

-----classes for information visualization

Views

-----classes for viewing data including filters, visualization, interaction and selection

Common

-----core classes

Filtering

-----classes related to data processing in the visualization pipeline

GenericFiltering

-----an adaptor framework to interface VTK to external simulation packages

GeoVis

-----view, sources and other objects useful in terrain visualization

Graphics

-----filters that process 3D data

GUISupport

-----classes for using VTK with the MFC and Qt user interface packages

Hybrid

-----complex classes that depend on classes in multiple other directories

Imaging

-----image processing filters

IO

-----classes for reading and writing data

Parallel

-----classes used to render

Utilities

-----supporting software like expat, png, jpeg, tiff and zlib

VolumeRendering

-----classes used for volume rendering

Widgets

-----3D widget classes

Wrapping

-----support for Tcl, Python, and Java wrapping.

Examples

-----examples, grouped by topic

CMake

-----configuration files for cross-platform building

Application software: ParaView

Except creating VTK application using the Tcl ( in this case, pre-compiled binaries may be available for the windows platform), you will have to compile and link the source code to produce libraries and executables.

VTK = visualization pipeline + rendering engine

Visualization pipeline is used to acquire or create data, process that data, and either write the results to a file or pass the results to the rendering engine for display

Rendering engine is responsible for creating a visual representation of the data

Actors: serves to group rendering attributes such as surface properties(e.g., ambient, diffuse, and specular color), representation(e.g., surface or wireframe), texture maps, and a geometric definition(a mapper)

Mappers: geometric definition using analytic primitives such as points, lines, polygons and triangle strips, the mapper terminates the visualization pipeline and serves as the bridge between the visualization subsystem and the graphics subsystem

Coordinate systems:

Display: x-y pixel values in the rendering window, the original is the lower-left corner

View: x-y-z(-1,1) values in camera coordinates(z is depth)

Operation performed on image data in VTK: image processing + geometry extraction + direct rendering