Geometry Toolbox Manual

Chapter 1

Introduction

Chapter 1 - Introduction

A. Introduction

The Geometry Toolbox is the result of an ongoing collaboration between Wes Bethel at Lawerence Berkeley Labs (LBL) and the Khoros Group at Khoral Research (KRI). The focus of this collaboration has been to add support for the creation and transport of geometry data vis a vis visualization algorithms and geometry data services, and in developing a geometry renderer. The Geometry Toolbox is the first step toward taking Khoros into the exciting field of geometry and volume visualization.

A.1. What is Geometry?

According to Webster's, **geometry** is *"The mathematics of the properties, measurements and relationships of points, lines, angles, surfaces, and solids ... a surface shape."*. In terms of data visualization, geometry refers to structures which specify shapes in three dimensions. There are also numerous attributes which may be associated with geometry which do not have any direct bearing on the spatial description of a structure, namely colors, surface characteristics, text strings such as names, and so forth.

A.2. Using Geometry for Visualization

The following describes in general how geometry is used for visualization. First, geometry is stored in a data object which is passed between processing modules just as images or other data is passed between data processing routines. The destination of the geometry data objects is an xvroutine named **RenderMonster**.

The following example illustrates this principle. Non-geometry data is read into a network. Visualization modules from the Geometry toolbox are then used to create geometry based on information in the non-geometry data. The geometry, when viewed, will hopefully allow a scientist to gain insight into the data or the process that creates the data. The geometry is turned into an image by a process called **rendering**. All geometry produced in a workspace which is to be viewed must be passed into RenderMonster for rendering.

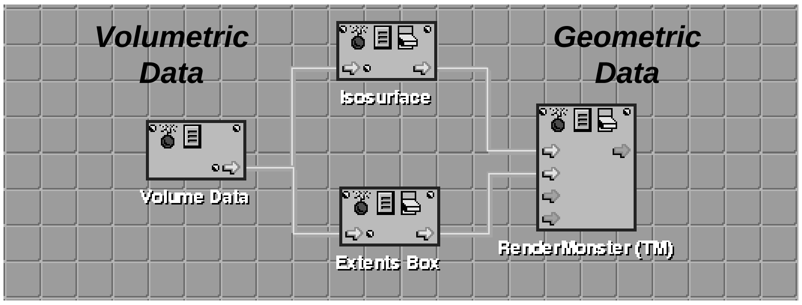


Figure 1: Illustration of a simple geometry workspace. Geometry is generated from an input volumetric data set. The RenderMonster program renders the geometry data into an image.

Geometry is generally created as the result of various visualization algorithms. Extensive geometry modeling and editing tools are not provided, as that is not the focus of this effort. The intent is to provide an environment for research and development in the visualization of scientific data, in contrast to the construction of complex geometry models (better suited for a CAD system). Geometry is intended to be used as a means to visualize data. Isosurfaces, for example, are constructed using triangles to represent functions of a constant value within a volumetric dataset. *Geometry is simply a convenient means for expressing the results of various visualization algorithms.*

Users may write custom modules for visualizing data, ie., modules that produce geometry from non-geometry data. Khoros provides a geometry services library to facilitate the storage and retrieval of geometric data. Please consult *Chapter 3 "Geometry Data Services"* in *Program Services - Volume II* for more information.

B. Programs in the Geometry Toolbox

The table below provides a complete list of the operators that exist within the Geometry Toolbox. In the sections that follow, the different programs are explained briefly. More detailed information about each program is included in the next chapter.

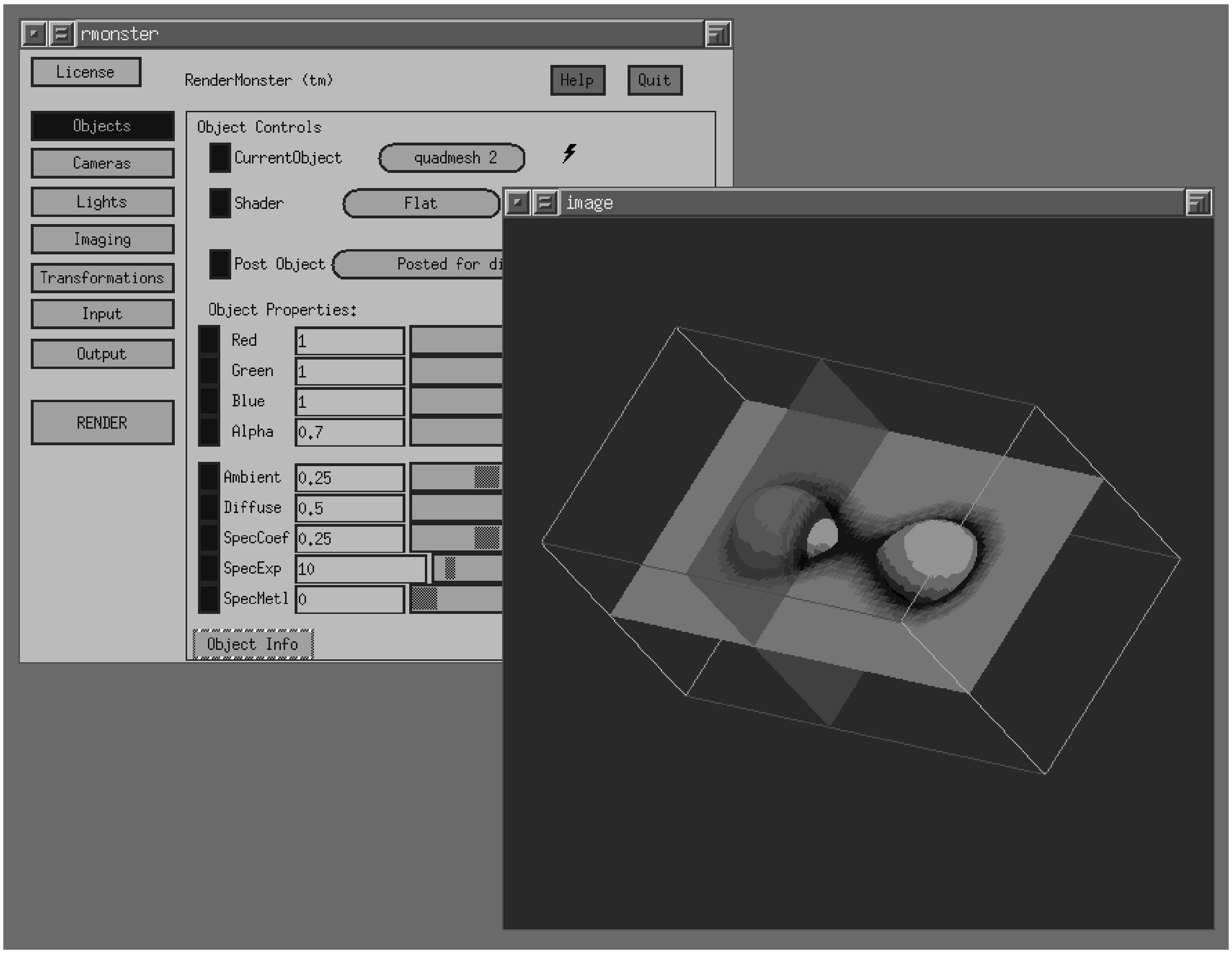
Toolbox GUI Operators: geometry

Category Subcategory	Operator	Description	Executable
Geometry	Color Legend	Construct Geometry Representing a Color Legend	`color_legend`
Annotations	Extents Box	Generate Geometry Representing Extents Box Around Data	`extents_box`
Geometry	Import Facet Data	Import FACET data	`kimportfacet`
Data to Geometry	Isosurface	Create Geometry Representing a 3D Contour within Data	`isosurface`
	Octmesh	Create an Octmesh Geometry Object from 3D Data	`octmesh`
	Plot Color	Create Polylines from ASCII XYZ data and a Vis Cmap	`plot_color`
	Plot XYZ	Create Polylines from ASCII XYZ Data	`plot_xyz`
	Quadmesh	Create a Quadmesh Geometry Object from 2D Data	`quadmesh`
	Spheres	Produce Spheres Whose Radii are Determined from Data	`sphere_maker`
Geometry Information	Print Geometry	Print ASCII Report of Geometry Input	`kprgeom`
Geometry	Grid 2D Data	Compute z=f(x,y) over a user-defined grid given scatter data.	`kgridder_2d`
Region Operators	Ortho Slice	Extract (N-1)D Region from (N)D Object	`kslicer`
Geometry Texture Operators	Apply Texture	Assign a Texture to a Geometry Object	`kapplytexture`
Geometry Visualization	RenderMonster (TM)	Interactive Geometry Visualization	`rmonster`
Geometry	Create Vismap	Generate Colormap: Create a Colormap for Mapping Float Data	`kmakecmap`
Visualization Colormap	Map Vismap	Map Data: Map Value Data Through a Visualization Colormap	`kgmapdata`

C. RenderMonster

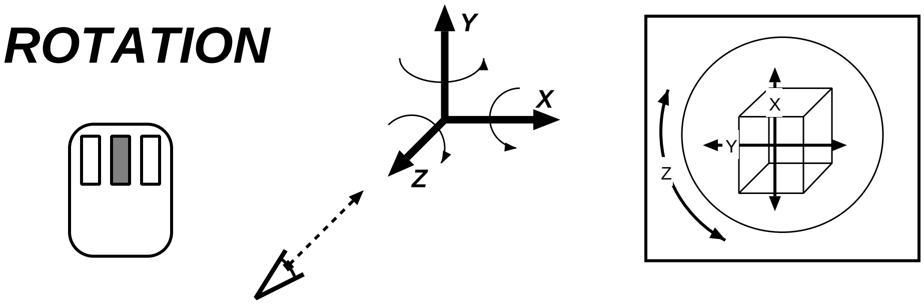
The RenderMonster is the Khoros software geometry renderer. Implemented as an Xvroutine, it may be included as part of a Cantata workspace or run from the command line. The RenderMonster accepts input in the form of geometry created by the visualization modules in the Geometry Toolbox, and produces images, displaying them, and interacting with the user.

When RenderMonster comes up, it will pop up two windows. One window is the control panel for the RenderMonster, the second is the image display window. The image display window will initially be the background color, that is, no geometry is initially rendered. Press on the "render" button on the control panel and any input geometry will be rendererd.

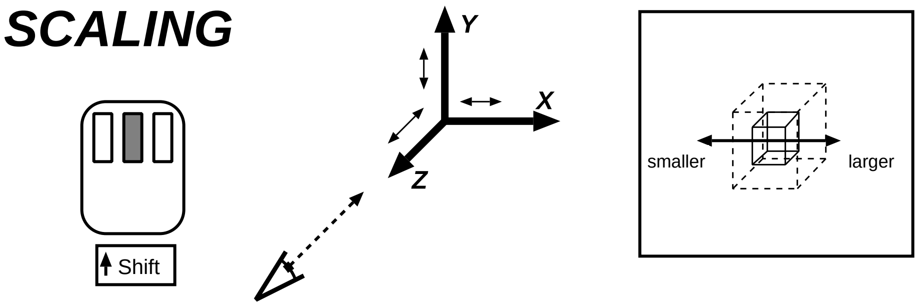


Direct interaction with the rendered image is available with the mouse. Pressing different mouse buttons on the image and moving the mouse will perform different transformations. When a transformation is initiated by pressing on a mouse button, you will see a bounding box representing the extents of the geometry. Moving the mouse will transform this bounding box. When the desired transformation has been performed, releasing the mouse button will initiate a rendering at the new transformation.

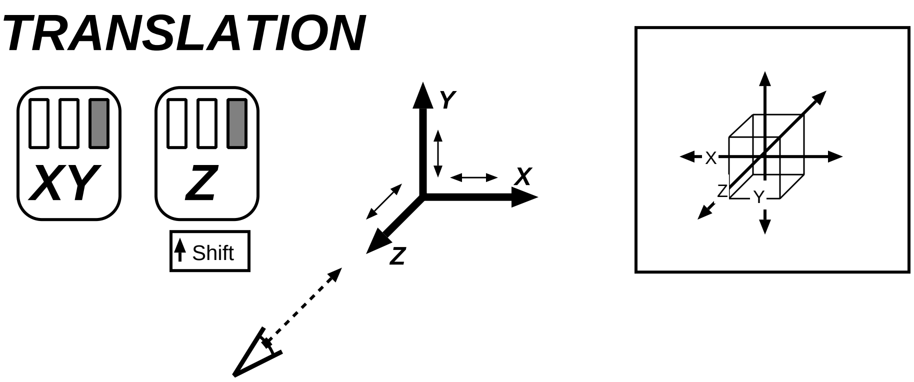
Rotations - Pressing the middle mouse button on the rendered image and moving the mouse will allow you to rotate the geometry. When you press down on the middle mouse button, you will see a circle drawn on the image. Moving the mouse up and down within this circle will perform an X rotation. Moving the mouse left and right within this circle will perform a Y rotation. Moving the mouse around the outside of the circle will perform a Z rotation.



Scaling - Pressing the middle mouse button while holding down the shift key on the rendered image and moving the mouse will allow you to scale the geometry. Moving the mouse to the left will scale downward and moving the mouse up and to the left will scale upward.

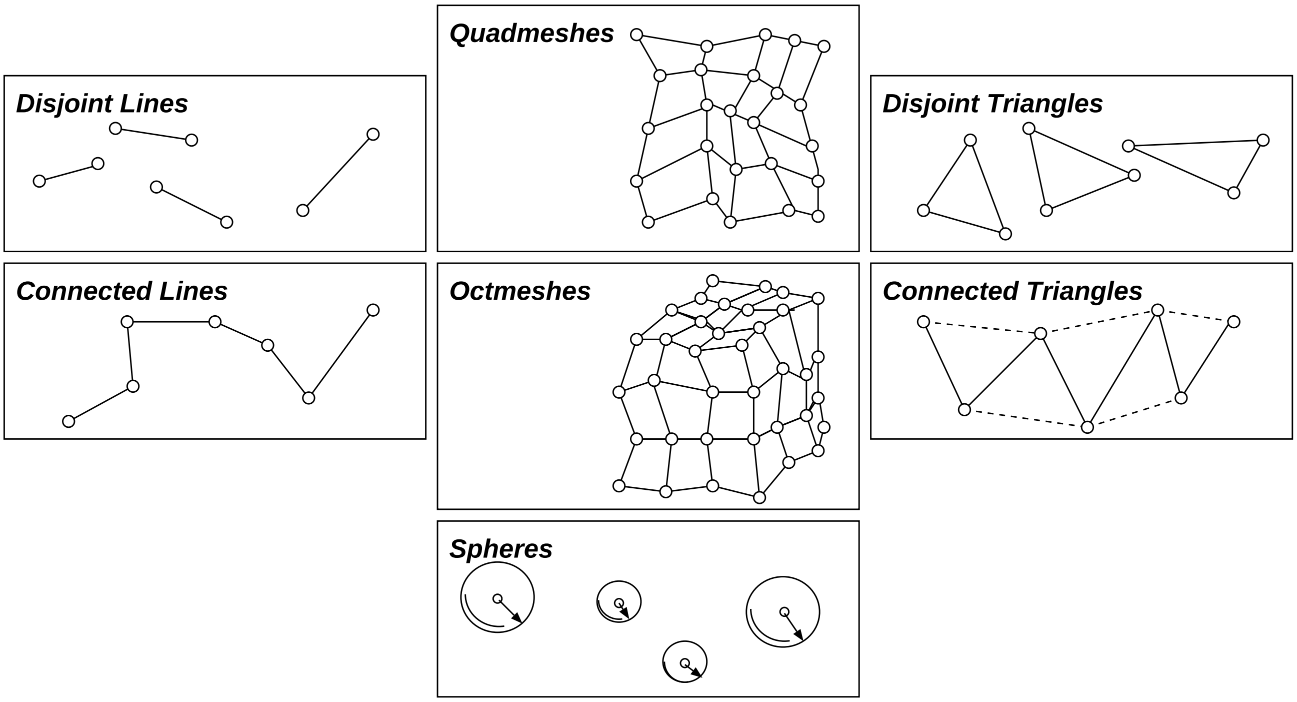


Translation - Pressing the right mouse button the rendered image and moving the mouse will allow you to translate the geometry in the X and Y directions. Translations in the Z direction is accomplished by pressing the right mouse button and the shift key.



Supported Geometry

The figure below illustrates the different geometry primitives which can be rendered directly by the Render-Monster.



In future releases, texture mapping will be supported, along support for rendering text, directed points, and cylinder primitives.

D. Visualization Colormap

The geometry toolbox provides a number of *visualization colormap* modules. A visualization colormap is different from a garden variety colormap in one important way. The visualization colormap uses a part of the location segment to indicate how colors are extracted from the map. In normal colormaps, a *pixel* value is used as an index into the colormap, producing some new value(s). Using this scheme, it is not possible to have perform mapping functions on floating point or large integer data.

The visualization colormap uses values in the polymorphic location segment which effectively "bracket" entries in the map. In other words, the user may define a range of values which map to a particular map entry, thus producing non-linear and discontinuous mapping functions.

At present, this module supports only a linear range which gets spread evenly over the entire colormap. The range is implemented using floating point values. There is one location segment entry per map entry, which defines the minimum value corresponding to that entry. All values greater than or equal to a particular bracket value, but less than the next one, will define the range for a given map entry.

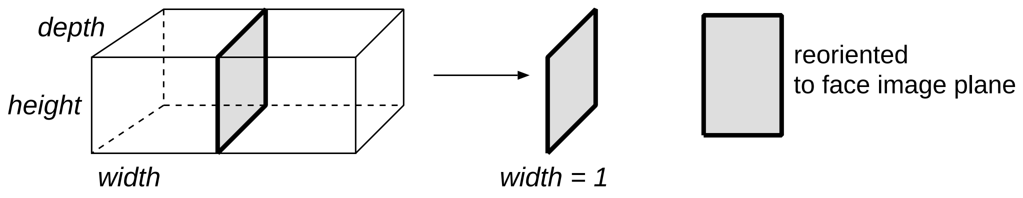
The color information in the colormap may be specified in either RGB or HSV space, each of which has it's own advantages and disadvantages. The alpha, or opacity, channel is optionally specified separately from the color information.

This module is highly useful for creating maps for floating point numbers. Many of the visualization modules in the Geometry toolbox make use of visualization colormaps.

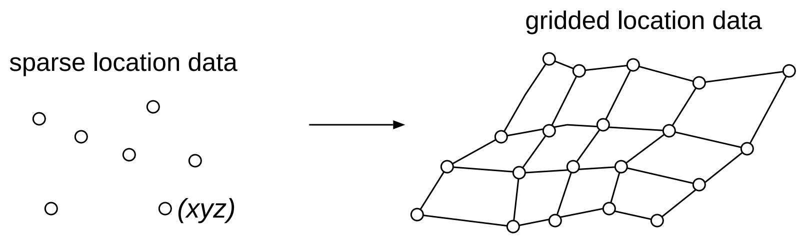
E. Region Operators

The region operators in the geometry toolbox are useful programs for manipulating volumetric data to make it more suitable for volumetric processing or geometry processing.

An orthogonal slicer allows the easy extraction of (N-1)D regions from ND datasets.



A gridder routine allows the construction of regular grids from sparse location points. A Fourier Analysis method is used to construct the grid. Given the input values, the frequency components of the estimated function are computed. The values at each output grid point are computed using this function.



F. Data to Geometry

A number of different routines are provided for inputing a set of data (either in the form ASCII values or a polymorphic data object) and producing geometry data. The isosurface routine, given a volumetric data set, generates geometry data which represents a contour surface in three dimensions. A quadmesh and octmesh routine map two and three dimensional data sets though a visualization colormap and provide colored mesh data which is suitable for rendering. A routine which generates polylines given a series of xyz points as well as a routine which imports General Dynamic ACAD facet files are also available.

G. Annotations

Currently the Geometry Toolbox provides an extents box routine, which generates polylines representing the extent of the location data in a volumetric data set, and a color legend routine, which generates a color legend from a visualization colormap.

H. Information

The geometry printing routine provided under the information category generates an ASCII printout detailing the contents of a data object containing geometry data.

This page left intentionally blank

Table of Contents

This page left intentionally blank