Interactive 3D-visualization in R

Oleg Nenadić, Daniel Adler
Institut für Statistik und Ökonometrie, Universität Göttingen
Platz der Göttinger Sieben 5
37073 Göttingen, Germany.
email: onenadi@uni-goettingen.de

Introduction

A shortcoming of many statistical software packages is the lack of sophisticated methods for three–dimensional visualization. Extending the visualization facilities from two to three dimensions, allthough seemingly trivial, raises many conceptual and computational issues. Due to the rapid development of computer hardware over the last years, today's entry–level computer have enough computing and graphic power for sophisticated three–dimensional displaying. Nevertheless, corresponding routines for 3D–visualization need to be programmed in an efficient manner in order to avoid bottlenecks. Conceptual issues include navigation in three–dimensional space, for data–exploration as well as the architecture of software.

We present a library for the R software (Ihaka and Gentleman 1996) which provides 3D–visualization capabilities. The library is written in C++ using OpenGL and has been ported to Win32 and X11 platforms.

Software Architecture

The implementation described here was driven by an object-oriented software design. Important interfaces required for implementing a 3D plug-in are presently not covered by the R software. This led to the development of a foundation layer which abstracts core system facilities, such as graphical user-interfaces and OpenGL contexts. While dependencies to other libraries are reduced to a minimum, a lot of flexibility for designing the interactive navigation user-interface is retained. The application of Design Patterns (Gamma, Helm, Johnson and Vlissides, 1994) proved helpful in this context. The general architecture was of crucial importance for implementing an R plug-in which is portable across platforms while being an integral part of the R programming interface.

The *visualization model* comprises five basic object types, namely *shapes*, *viewpoint*, *lights*, *bounding box* and *background*. The *scene desciption* is stored in a composite object model which is evaluated by the system at a highly frequent rate. A stack semantic for multiple objects (*shapes* and *lights*) allows for adding, replacing or removing particular or all objects from the scenery.

The *navigation concept* provides an environment for interactive exploration of the three–dimensional scenery. The viewpoint is specified in polar coordinates on a sphere which surrounds the scenery, while the viewing direction is automatically directed towards the center of the scenery. A pointing device (commonly a mouse), which allows for movement in two directions, is used for navigation on the sphere. Using the left button and moving the mouse results in movement along the sphere, while the right button controls the zooming level. Additionally, the perspective distortion (field of view) can be set using both mouse buttons. The navigational status can also be set within the console using the corresponding viewpoint–function. Currently, interactivity is restricted to navigation, interaction with data is currently not implemented.

The functionset

The API currently provides 21 functions, which can be divided into six categories. *Device management functions* control the window device – they open and close devices, set the active device or shut down the device. Objects can be removed from the scenery using the *scene management functions*. Snapshots from a particular scene can be taken with the *export funtion*. Currently, the export function only supports PNG–bitmaps. The core of the library comprises *shape functions* that are used for creating "primitives", such as points, lines, triangles etc. in three–dimensional space. The environment, including the viewpoint, background and the bounding box, can be modyfied with the *environment funtions*. The functions are structured in a similar way to the corresponding R–functions, so that users familiar with R can quickly learn their usage. A complete list of the currently implemented functions is given in Adler and Nenadić (2003).

Shape functions provide the "low–level" plotting tools. The following "primitives" are currently supported: *points*, *lines*, *triangles*, *quads* (planes), *spheres*, *surfaces* and *sprites* (particles). These functions allow the user to creating basic 3D objects and to add these to the current scenery. More complex objects can be easily constructed using these primitives – e.g. a three-dimensional histogram can be constructed from a set of quads (planes). Additional attributes, such as color and transparency, are controlled with the *appearance function*. The appearance features, such as lighting, alpha blending (transparency), texture mapping, fog effects, internal smoothing and side–dependant rendering are not essential for displaying three–dimensional data, but nevertheless they can be useful for exploratory data analysis. For example, visualizing and comparing bivariate densities can be performed using transparency effects and side–dependant rendering.

Selected examples of applications in applied statistics are given in Nenadić, Adler, and Zucchini (2003). Binaries and sources from the software, including access to the cvs developer branch, are accessible at http://134.76.173.220/ dadler/rgl/index.html.

REFERENCES

Adler, D. and Nenadić, O. (2003), A Framework for an R to OpenGL Interface for Interactive 3D graphics, *Proceedings of the Distributed Statistical Computing DSC 2003* (to appear).

Gamme, E. and Helm, R. and Johnson, R. and Vlissides J. (1994), *Design Patterns, Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Ihaka, R. and Gentleman, R. (1996), R: A Language for Data Analysis and Graphics, *Journal of Computational and Graphical Statistics*, **5(3)**, 299–314.

Nenadić, O., Adler, D., Zucchini, W. (2003), RGL: A R–library for 3D visualization with OpenGL, *Proceedings of the Interface 2003* (to appear).

RÉSUMÉ

Nous presentons la bibliothèque RGL pour le logiciel de statistique R qui fournit le visualization graphique en trois dimensions. Dans le total, RGL contenit 21 fonctions pour R qui sont similair à le structure des fonctions sur R. RGL est creé sur C++ et est porteé aux systemes Win32 et X11.