ELSEVIER

Journal of Molecular Graphics and Modelling 20 (2002) 411–413

Journal of Molecular Graphics and Modelling

www.elsevier.com/locate/JMGM

MOSBY: a molecular structure viewer program with portability and extensibility

Yutaka Ueno*, Kiyoshi Asai

Computational Biology Research Center, National Institute of Advanced Industrial Science and Technology, AIST Tsukuba Central 2, Tsukuba 305-8568, Japan

Accepted 26 November 2001

Abstract

A molecular structure viewer program, MOSBY has been developed for studies that use atomic coordinates to understand the structures of protein molecules. The program is designed to be portable with a comprehensive user interface by our high-throughput graphics library. In addition, it cooperates with extension modules customized for individual research topics and analysis. For example, an electron density module loads and displays electron density maps derived in X-ray crystallographic analysis superimposed to an atomic model. A molecular dynamics module reads a trajectory file of the results of molecular dynamics calculations and animates the structure. These plug-in modules are devised to function without modification to the MOSBY program. For variations of analysis and calculations with atomic coordinates, the portability and extensibility illustrated by MOSBY play an important rule in scientific computational tools with active software development.

Keywords: Molecular graphics; Software component; Graphics library; Electron density; Dynamics; Animation

1. Summary

With an increasing repository of atomic coordinate data in the Protein Data Bank (PDB), molecular graphics have become a fundamental tool in the wider area of molecular biology. While graphics acceleration hardware have been expected to facilitate molecular modeling tasks, RasMol [1] achieved acceptably fast rendering for several thousand atoms only with software code. However, it is incapable for extensions for practical computational tasks: viewing electron density maps derived from a X-ray crystallographic analysis; animation of the movement obtained in molecular dynamics simulations.

In order to support variations of computational tasks with atomic structures of molecules, we have started to develop a molecular structure viewer program. Our goal is to provide a software platform that runs on common hardware and allows users to add new functions with only an average knowledge of programming. In order to achieve desired portability and extensibility, the software design itself was reconsidered. Our software code was developed with following characteristics:

High-throughput graphics: Our three-dimensional (3D) graphics library [2] was designed for a molecular structure viewer program to provide both portability and high-throughput rendering without hardware acceleration. It was designed for a fast sphere rendering by a direct scan-conversion algorithm. The library renders primitives of lines, boxes, character text, circles, polygons, spheres and cylinders into an off-screen image buffer which is then transferred to the raster graphics hardware to display on the monitor. This method is also suitable for current computer hardware.

Edge lines in a molecular picture: Since protein molecules make a complex with other proteins at work, or a protein itself has characterized segments of domains and a region of interest, drawing thick edge lines on boundaries between objects provides comprehensive renderings [3]. Our graphics library supports an overlay plane of the picture image buffer, which facilitates the drawing of edge lines. When primitives are rendered to the image buffer, the specified attribute value is also stored to the area of the object in the overlay plane. The boundary of the differently attributed object can be detected simply by comparing adjacent pixel values of the overlay plane.

A plug-in software architecture: In order to add new functions to an application program, the most desired method is to use dynamic linking of plug-in style extension modules. Users can choose only specified modules loaded at

* Corresponding author. Tel.: +81-298-61-5965; fax: +81-298-61-5722.
E-mail address: uenoyt@ni.aist.go.jp (Y. Ueno).

1093-3263/02/$ – see front matter
PII: S1093-3263(01)00140-1

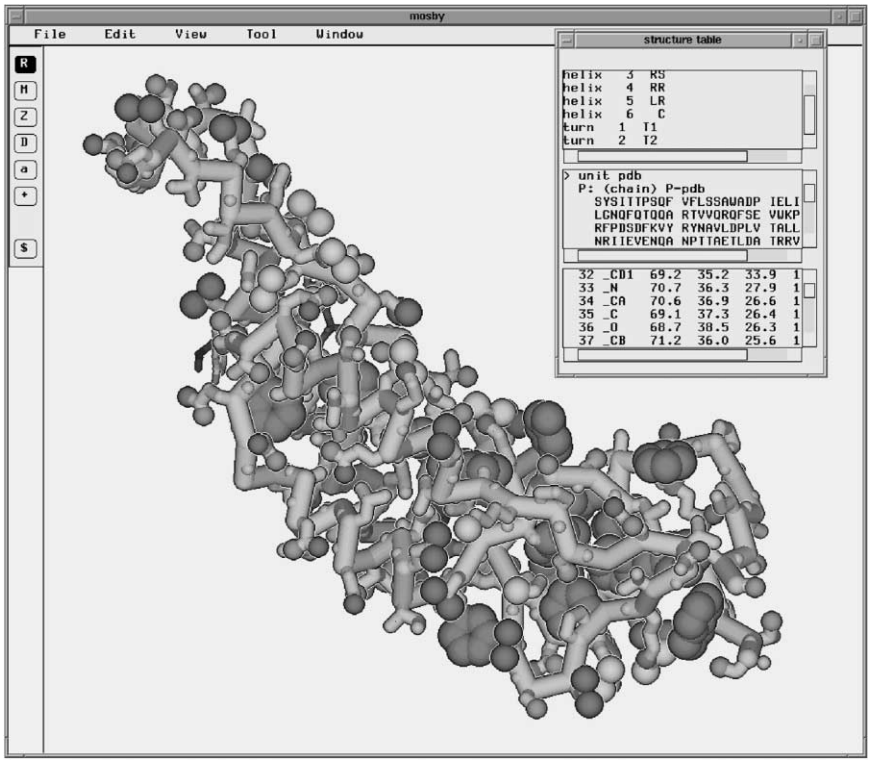

Fig. 1. A snapshot of MOSBY program. The peptide backbone of TMV protein is accented with ionic oxygen and nitrogen atoms, and aromatic residues [3]. Molecular tool buttons on the left side of the window select the mode of mouse operation. A structure table window shows the amino acid sequence and secondary structure groups of the protein. The selection of atoms or residues is also reflected in the molecular model window, and vice versa.

the startup of the program. We have introduced a new plug-in software component architecture [4], so that the main program can equip a new function and new command without prior knowledge of the upcoming extension modules.

A Scripting framework: Scripting is the process of automating application program tasks by using commands to the application, that can be prepared in a script file. We developed a simple scripting framework [5] so that different application programs can employ the same scripting feature. The inter-process communication function is used for the message passing. The message can be used as an alternative to the keyboard input, that is, as a command to the application program.

Using those software method, our MOSBY program, which stands for MOlecular Structure Browser with anal-Ysis, was implemented. Since required software code are general topics in writing application programs, we packaged these features as our programming library called ASHLEY (Application Support Hybrid Library for easY programming). The library also maintains portability to different operating systems. Our first development employed a UNIX and X-Window system (SGI, DEC, HP, Sun and Linux). Then, the code was ported to MacOS and Windows operating system minimizing machine-dependent code in ASHLEY library. The source code of the program and extension modules are accessible from our web site (http://staff.aist.go.jp/yutaka.ueno/mosby/).

Fig. 1 is a sample snapshot of MOSBY with the tobacco mosaic virus (TMV) coat protein [3] (PDB file 2TMV). Atomic coordinates are stored in our molecular data list which maintains multiple molecules loaded from PDB files. Symmetric units are also supported (Fig. 2) by matrix operations. Currently, extension modules are developed for:

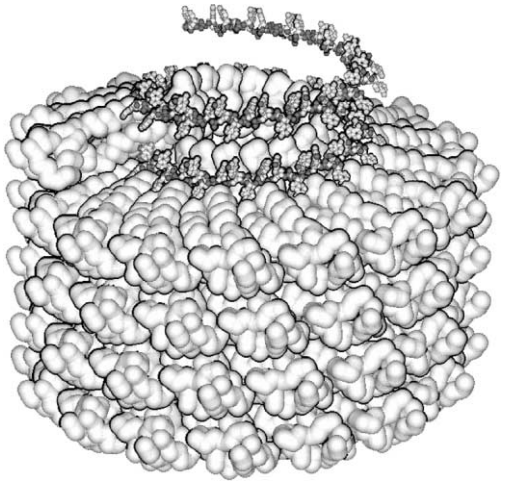

Fig. 2. The filament structure of TMV. The rod shape of the virus is made of this helical arrangement of the coat protein, in the genetic helix of three turns with 49 units. Peptide backbones are in 4 Å radius cylinders, and the RNA chain is depicted in VDW spheres [3].

electron density map display for the X-ray crystallography results; trajectory animations for the molecular dynamics simulation [7]; an import filter for MDL file format; stereo display enabler (Silicon Graphics Inc.); ribbon model display extensions, and so on.

Considering the portability, OpenGL [6] graphics library is advocated as a standard. However, we have chosen more portable and high-throughput solution only with software code. We found that texture mapping polygons are greatly supported by the OpenGL based hardware, which was not useful in standard MOSBY functions. In terms of the extensibility, a general visualization system such as AVS have also demonstrated various computational modules with molecular graphics. Although, MOSBY is a much simple and light weight application program dedicated to molecular graphics tasks.

While classical molecular graphics programs have been either high-end systems on workstations for specialists or low-end systems on personal computers for general biologists, our program is designed to be used by both types of users. Its portability supports wider use in different hardware giving rise to more scientific discussions. Then, different computational tasks are supported by means of custom made extension modules. The extensibility allows flexible software configurations and developments for different users' requests without making a huge software system by numerous functions. Our MOSBY illustrated that the portability and extensibility are prerequisites for a software platform in the scientific computing.

References

[1] R.A. Sayle, E.J. Milner-White, Trends Biochem. Sci. 20 (1995) 374–376.
[2] Y. Ueno, K. Asai, in: Altman, et al. (Eds.), Proceedings of the Pacific Symposium on Biocomputing 98, World Scientific Press, Singapore, 1998, 201–212.
[3] K. Namba, D.L.D. Caspar, G. Stubbs, Biophys. J. 53 (1988) 469–475.
[4] Y. Ueno, K. Asai, in: Gaasterland, et al. (Eds.), Proceedings of the Intelligent Systems for Molecular Biology, AAAI Press, Menlo Park, USA, 1997, pp. 329–332.
[5] Y. Ueno, K. Asai, M. Arita, in: K. Asai, et al. (Eds.), Genome Informatics 1999, Universal Academy Press, Tokyo, 1999, pp. 166–175.
[6] Silicon Graphics Inc., The OpenGL Programming Guide, Addison-Wesley, Reading, USA.
[7] Y. Komeiji, M. Uebayasi, R. Takata, A. Shimizu, K. Itsukashi, M. Taiji, J. Comp. Chem. 18 (1997) 1546–1563.