BUTTERWORTH HEINEMANN

XmMol: An X11 and motif program for macromolecular visualization and modeling

P. Tufféry

Unité de Recherches Biomathématiques et Biostatistiques, U 263 INSERM, Université Paris 7, Paris, France

XmMol is a desktop tool designed to provide both interactive molecular graphics on X11 displays and easy interface with external applications. A kernel provides an interactive wireframe display of macromolecules. It supports depth cueing, 3D clipping, and stereo. Various representations, coloring, and labeling modes are proposed. Docking and interactive backbone deformation tools are also supported. Communication protocols allow the user to develop new external features or to use XmMol as a visualization tool for external numerical programs.

INTRODUCTION

Despite the fact that molecular representations are increasingly varied and sophisticated,[1–5] most of the interactive molecular graphic programs developed so far appear to be of limited portability. This is mainly due to the high specificity of the graphic libraries used: specialized libraries, such as E&S GSR, GL, Phigs, or Dore, generally implemented on specific platforms, are most often used (see, e.g., Refs. 6–13, and many others). With the generalization of the X11 interface, it has become possible to design platform-independent molecular visualization tools.[14–16] Although the graphics capabilities of such tools are expected to be much less than those obtained on specialized workstations, most of the molecular structure analysis can be done on X stations, while reserving the use of specialized platforms for more sophisticated and elaborate representations. However, the use of widespread X platforms (including UNIX workstations and X terminals, but also PCs or Macs used as X servers) is making three-dimensional (3D) molecular visualization commonplace. Thus, the increasing usage of the X standard makes it more necessary that, faced with the various wishes of a large number of users, graphics programs must be easily interfaced with local applications.

XmMol was designed to satisfy two goals:

1. Interactive graphics under X11 displays: The program offers interactive wireframe molecular representations, coupled with interactive manual docking or modeling functionalities.
2. Easy customization, enhancements, and dialog with external programs: XmMol can communicate with external applications at different levels. In addition to built-in interfaces with other programs (MolScript[4] or Raster3D[6]), XmMol supports a dialog with external programs at two levels:
   a. A system level that allows XmMol to run external programs as shell commands. At this level, macros enable XmMol to communicate with external applications via coordinate files.
   b. A synchronous dialog to delegate tasks. At this level, the user can directly write new functions or simply interface preexisting external programs with XmMol. The dialog between external delegate tasks and XmMol is controlled by software interrupts. Finally, on machines supporting the System V Inter Process Communication facilities, sharing memory is possible for part of the data.

KERNEL DESCRIPTION

Programming considerations

The program is written in C, and its user interface is based on the X11 and Motif libraries. It automatically adapts itself to black-and-white, gray, or color displays. The graphics are entirely based on X11 drawing resources: No drawing algorithm is implemented with XmMol, unlike RasMol,[15] and the double buffering is simulated by using a pixmap. Owing to this choice, the graphic performance of XmMol is tightly dependent on the speed of the X server: the CPU supporting XmMol only computes the coordinate transformations and provides the X server with a series of graphic requests. Standard 3D viewing transformations[17,18] have been implemented.

To ensure a satisfactory interactivity, only wireframe representations are currently supported, unlike other X11 dedicated visualization programs such as PDBview,[14] RasMol,[15] or XMol.[16]

Color plates for this article are on p. 62.

Address reprint requests to Dr. Tufféry, Unité de Recherches Biomathématiques et Biostatistiques, U 263 INSERM, Université Paris 7, Tour 53, 1er Etage, 75251 Paris Cedex 05, France.

Received 7 June 1994; revised 15 August 1994; accepted 30 August 1994.

655 Avenue of the Americas, New York, NY 10010

0263-7855/95/$10.00
SSDI 0263-7855(94)00011-G

Instead, care was taken to optimize 3D rendering so that XmMol may support topics such as depth cueing, clipping, or stereo, while conserving a satisfactory interactivity.

The default coloring mode is, when possible, pseudocolor. As pseudocolor or gray-scale colormaps generally have a maximum of 256 entries, the depth cueing is generally restricted to 8 shades of a color, which can be adjusted, although it was found sufficient for the eye. No explicit Z buffer is used, but a Z sorting algorithm is used to assign shades of the colors.

To optimize the performances for each X server, command line arguments control pixmap use, colormap allocation, and installation, and give an opportunity to force a black-and-white mode. In addition, clipping and depth cueing can be toggled on–off each time.

User interface

Owing to the extensive use of the Motif library, keyboard inputs are kept to a minimum. The different functions are available via mouse-driven selection and activated dialog boxes.

The basic interface exhibits three windows (see Figure 1).

1. The main window supports the graphics and a menu bar. Within the graphic window, the mouse buttons are used for picking (for atom labels, distances, etc.), for controlling the interactive 3D orientation of the displayed molecules (sensitive to the direction and the amplitude of the displacement of the mouse), or for controlling the modeling operators.
2. A 3D control panel offers a more exhaustive control of 3D transformation: interactive rotation, translation, scaling, zooming, as well as back and front clip planes or depth cueing controls can be activated via the mouse. The speed of the transformation is correlated with the position of the pointer within the control buttons.
3. A third window summarizes information about the files presently in-

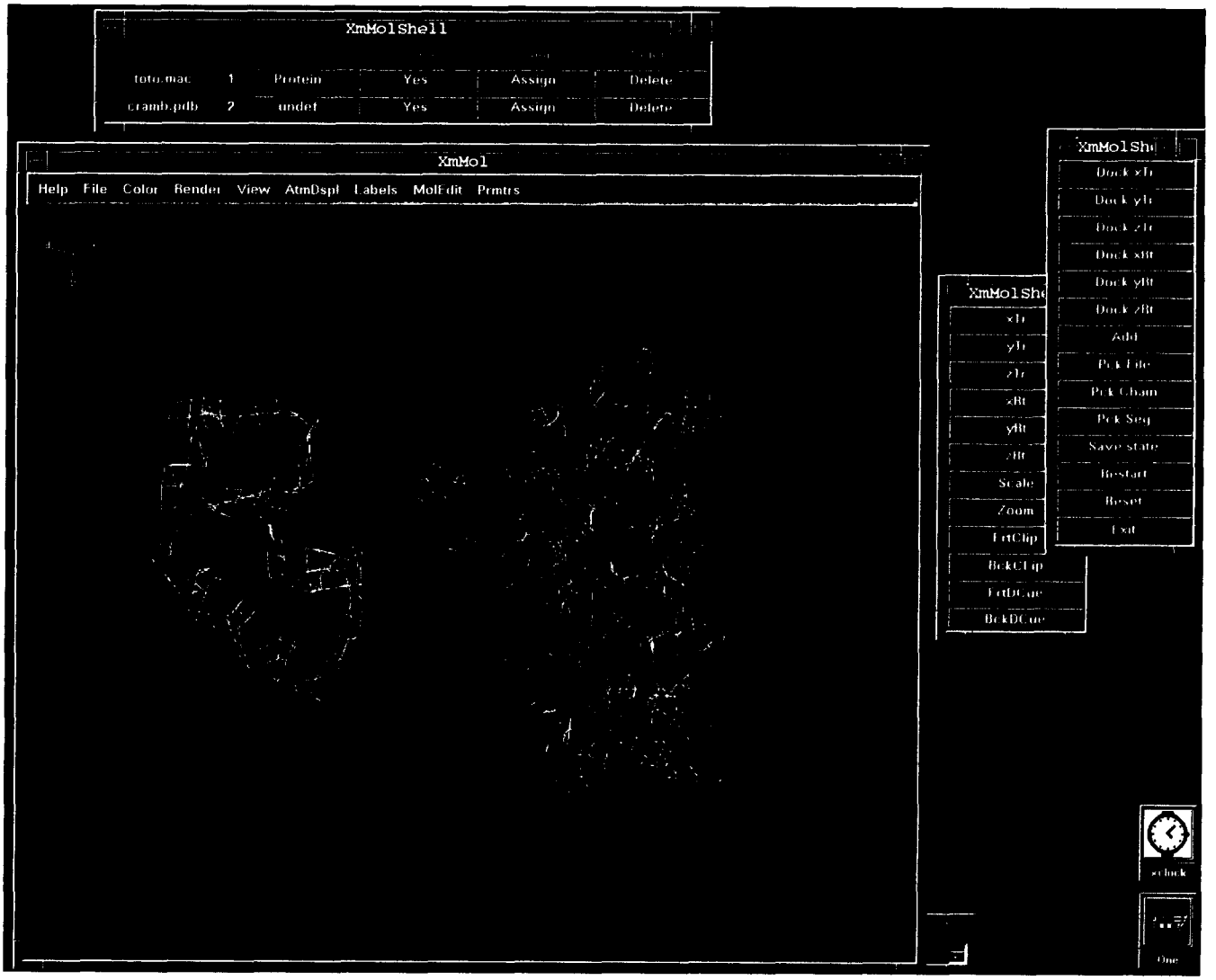

Figure 1. XmMol screen organization. The three main windows support (1) the menu bar and the graphic area; (2) a button panel for 3D transformations (x and y rotations can also be obtained through the mouse within the graphic area); and (3) a summary of the files input that gives access to some parameter control. Temporary additional windows may appear on request (such as the dock panel presented here). The cramb.pdb file was obtained from the PCurves program.[22]

put. For each file, a series of labels describes the main properties: file name, file internal identifier (Id; used for some external commands), and file type (protein, DNA, RNA, or other). Push buttons enable the on–off toggling of a file display, file type definition in case of mismatch, the reassignment of colors, and file closure.

Other specialized windows are managed for atom display selections, color assignments, docking, user dialogs, or help.

XmMol X resources (window sizes, colors, fonts, and locations) can be accessed within the X11 resource file (.Xdefaults or .Xresources) file. Configuration files allow one to redefine most of the static parameters, such as the X11 colors used by default, or the interatomic distances used for bond detection.

Coordinate input/output

For coordinates, XmMol uses an internal format that corresponds to that of the Flex force field.[19] Input can be performed either in PDB or XmMol formats. Output can be made in PDB, XmMol, or simple XYZ formats. It accepts chains, fragments, or displayed atoms masks. The number of files (or atoms) that can be input at the same time is limited only by the hardware, all memory allocation being done dynamically.

For each file loaded, a series of analyses if performed:

A file type (Protein, RNA, DNA, or other), based on the residue and atom names, is assigned.

A bond analysis is performed, using interatomic distances and atomic types, for files that have no bonding information. However, the CONECT fields of PDB files are always preserved—this implies that any nonmolecular representation that respects the PDB format can be read (see Figure 1). Forgotten or mismatched bonds can be corrected graphically.

Chain information is stored, and non explicit *fragments* (consecutive residues sets that are not linked but are not labelled as different chains) are detected.

Protein secondary structure is either read from PDB file HELIX, SHEET, or TURN fields or assigned by considering a hydrogen bonding pattern supplemented by phi and psi analysis for helices and sheets.

A more precise hydrogen bond detection is made for files that have explicit hydrogen coordinates, based on distance and angle criteria.

Ribbons of the structure are computed (according to Carson's algorithm[20]).

Molecular representations

The drawing modes currently supported are a default *all bonds* mode, a *trace* mode, or a *ribbon* mode (Color Plate 1). For the default mode, much attention was given to the possibilities of hiding atoms or not, because the use of X11 line drawing functions makes the number of lines crucial for performance. Apart from the possibility of toggling on–off the display of a whole file, a quick menu allows one to toggle the display of picked files, chains, segments, or to hide side chain hydrogens or heteroatoms. A more sophisticated selection process allows the selection of any combination of atoms to be displayed. Two colors are used to differentiate atoms that will be visible or not at the exit of the selection process. Different levels of display masks can be combined. In the end, the precision of this display selection mechanism is one atom.

Various coloring modes are proposed: half bond, coloring by files, by chains, by fragments, by protein secondary structure type, by backbone versus side chains, by residues, and by residue types or properties. For black and white displays, the coloring is superseded by the use of dashed lines. Because the colors vary depending on the screens used, the RGB values can be interactively edited by using sliders. Subsequently, the pixel identifiers assigned for each chain, segment, and residue type can be changed. Because it is difficult to adjust the gray levels for some gray-scale displays, an opportunity is left to toggle color or black-and-white modes.

Atom labels can be set at any time by picking the displayed structure. Two formats are supported: a long one includes chain identifier (if present), residue name, number, and atom name, and the short one is restrained to atom name. Residue labels can also be set (for all of the residues together). Interatomic distances, valences, and dihedral angles can be displayed, as well as hydrogen bonds for files that contain hydrogens.

Finally, stereo is supported (see Figure 2), and its parameters (the margin between the two regions and the rotation between the two views) are adjustable interactively. All the functionalities of XmMol (picking, docking, etc.) are preserved when stereo is active.

Modeling tools

Docking is supported. The moving atoms set can correspond to whole files, but also to chains or segments of a file. The docking coordinate transformation is made by concatenating a matrix to the global coordinate transformation matrix. This gives the opportunity, in the end, of choosing to keep the initial coordinates or to transform them permanently.

Some operators (stretching and invariant stretching) of the Forme package[21] have also been implemented. These operators allow the modification of the backbone coordinates by selecting a "deformation region" and applying a "force" to an anchor point. The structure of the backbone is then interactively modified so that the anchor point moves along the force direction. The mouse controls the speed and direction of the deformation, that is, in this implementation, within the plane containing the anchor point that is parallel to the operator's eye. Such operators currently work for proteins as well as for nucleic acids (at a single-strand level).

Hardcopy plots

Output of the views can be obtained in three formats:

1. PostScript files: The files include all the information concerning the display of the atoms (including stereo), the display of labels, interatomic distances, valence or dihedral angles (see Figure 2). Black-and-white and color PostScript files can be generated.
2. MolScript[4] input files: XmMol can generate MolScript input files for three types of representation: wire-

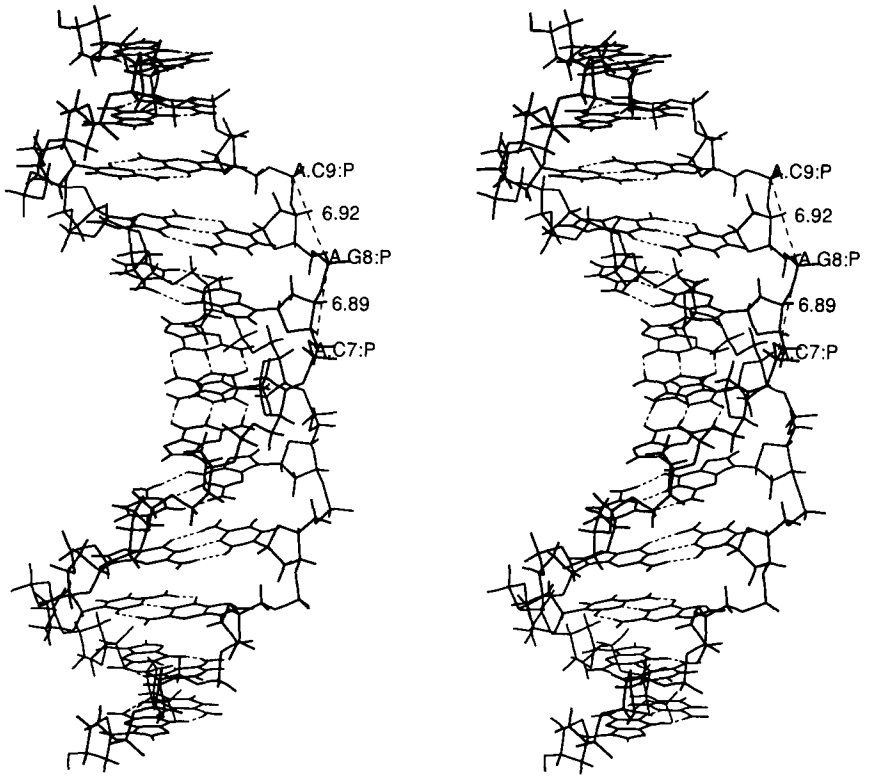

Figure 2. Example of a stereo plot of a DNA structure obtained through the PostScript driver of XmMol, including some hydrogen bond, atom label, and interatom distance information.

frame, ball and stick, or ribbon (for proteins). XmMol viewing transformation is preserved. Labels are included and stereo plots can be generated.

3. Raster3D[6] input files: The outputs of XmMol correspond to that of the programs *setup* or *rods*, and are direct inputs to the program *render*. Translation, rotation, scaling, and atom display masks are taken into account, as well as the colors used within XmMol. All the *render* file header parameters can be accessed by the user.

Hardware requirements

XmMol has currently been tested on various machines (HP 700 series, SGI Crimson, IBM RS 6000 series, and DEC alpha) coupled with various X servers (HP 700 RX, IBM 135 X station, Tektronix, Apple Macintosh, or PCs). Table 1 summarizes some indicative performances. The X stations had only 2 MB of video memory.

DIALOG WITH EXTERNAL PROGRAMS

To allow XmMol to be used in conjunction with as many external programs as possible, two levels of interface have been considered.

Dialog at the system level

The first level applies to the dialog with external applications to be used without any modification. This is simply solved by using the C system facility, which allows a program to run external shell commands. The commands are filtered by the XmMol command parser. It supports macros that enable one to describe what will be output from XmMol as parameters of the external commands, and what will be input. The exchange of information with external programs is currently based on input and output of coordinate files. The formats supported are PDB, XmMol, and XYZ.

Dialog based on a synchronous communication protocol

The second level applies to the dialog with external programs that have been specifically adapted to be called from within XmMol. This level takes advantage of the XmMol kernel but does not require any modification of it. The external "Delegate" processes are forked as children of XmMol. They can thus exchange information with it by using a communication protocol based on software interrupts, and named *pipes* (this mechanism is apparented to that proposed within MIDAS[8]). To achieve this, the external applications must link a dialog library that allows to establish the communication. However, the programs remain totally independent of XmMol, because all the calls to the communication library will remain silent if the program is not called from XmMol. As well, the protocol was designed so that XmMol functionalities are not affected during the connection: For example, interactive manipulation of the coordinates or the modification of the colors are still possible while running a delegate task.

A delegate program must include at least two lines:

```
XmMolConnect();
```

and

```
XmMolDisconnect();
```

They will establish and close the dialog. Having established a connection, the two processes can then exchange requests. All requests are made via standard dialog calls to the dialog library. For example:

```
XmMolSendMessage("XmXmPipe-
  CrdIn toto.pdb");
datalength = 3*sizeof(double);
for(i=0;i<nAtoms ; ++i) {
        XmMolSendData(&coords[i],
          &datalength);
}
```

are the only extra lines to plug into a minimizer code so that XmMol displays the evolution of the coordinates of the file toto.pdb during a minimization.

The perspectives offered by this type of interface have been kept as large as possible by leaving a large number of data and commands accessible to external tasks.

1. Any field of any file input within XmMol can be accessed or allocated (atom coordinates, names, labels, connectivity matrices, etc.). This is designed to give the users an opportunity to write programs that allow XmMol to input atomic coordinate files of any format. In addition, as mentioned above, this provides an easy way of visualizing

Table 1. Indicative performances for different sample files

File[a]	Mode[b]	CPU[c]	Xserver[d]	Dx s^{-1} [e]
I	HB	HP	Console	5.9
		HP	XHP	3.8
		HP	Mac	1.9
		HP2	PC	3.4
		IBM	XIBM	1.5
		SGI	Console	3.2
	SC	HP	Console	10
		HP	XHP	6.3
		HP	Mac	2.9
		HP2	PC	6.2
		IBM	XIBM	2.3
		SGI	Console	5.5
	NDC	HP	Console	16.6
		HP	XHP	8.3
		HP	Mac	3.2
		HP2	PC	7.7
		IBM	XIBM	2.9
	Trace	HP	XHP	50
		HP	Mac	11.1
		HP	PC	33.3
	Ribbon	HP	XHP	9
		HP	Mac	4.2
		HP	PC	12.5
II	HB	HP	Console	11.1
		HP	HP	11.1
		IBM	XIBM	4
		HP	Mac	4.5
		HP	PC	12.5
		SGI	XIBM	5
III	HB	HP	XHP	1.3
		HP	Mac	0.9
		HP	PC	2.0
	SC	HP	XHP	2.1
		HP	Mac	1.6
		HP	PC	3.4
	Trace	HP	XHP	14.2
		HP	Mac	6.3
		HP	PC	25
	Ribbon	HP	XHP	1.9
		HP	Mac	1.2
		HP	PC	2.8

[a]*File* corresponds to the following: I, The 7 helices of the bacteriorhodopsin (1BRD), hydrogens added, 171 residues, 2 805 atoms, corresponding to 6 656 lines for half-bond color; II, 2 helices (A and G + retinal) of the bateriorhodopsin, hydrogens added, 50 residues, 841 atoms, 1 980 lines for half-bond color; III, cardiopicornavirus coat protein (2MEV), 1 040 residues, 6 507 atoms, 24 883 lines for half-bond color.

[b]*Mode* corresponds to the drawing modes: half-bond (HB), single color (SC) (one color for the whole file, approximately half the number of lines than for half-bond). *Trace* corresponds to the C_α trace of these files, and *Ribbon* is for splines of the structures, single-file colored. All the drawings included 3D clipping and depth cueing (eight shades) for each color, except for the no depth cueing (NDC) test.

[c]*CPU* corresponds to the different machines running XmMol: HP 735 at 66 MHz (HP), HP 735 at 99 MHz (HP2), IBM 530 (IBM), SGI Crimson (SGI).

[d]*Xserver* corresponds to the different displays: consoles of these machines, HP X terminal 700 RX (HP), IBM X station 130 (IBM), Macintosh IIfx (Mac), PC 486 DX2 66 using a video card Stealth Pro, and running LINUX (PC). For the Mac server, the graphic window size was 500 × 400 pixels. For the PC server, the graphic window size was 600 × 600 pixels. For other servers, the window size was 700 × 700 pixels.

[e]*Dx* s^{-1} corresponds to an estimated number of structure drawings per second. These were established from the real time elapsed for a series of 100 screen refreshments in each case. The test was performed under working conditions (possibly many sessions on the machines) and the results are thus only indicative.

externally run minimizations and molecular dynamics, or to implement fitting routines, for example.

2. Any data relative to the 3D rendering can be accessed, as well as any display mask. This gives the opportunity of simply piloting the orientation of the files displayed.
3. Some features of the XmMol X11 interface can be called.
   a. Some specialized X11 widgets can be managed from delegate

tasks, and allowed to perform information or queries: file names, atom ranges, and so on.

b. It is also possible that a delegate task may implement new items into the menu bar of XmMol. This gives a possibility to have menu items that call any external UNIX command or any delegate task as if it is part of XmMol.

4. Finally, the use of shared memory is enabled for machines supporting the System V Inter Process Communication facilities. The current version of XmMol will give external programs a restrained access to the top-level XmMol parameters (i.e., not the coordinate files information), as well as an access to the data field of an Ximage. This gives the user an opportunity to interface its own rendering modes and thus to give XmMol access to representations that are not wireframes. External rendering routines can be used to produce static images, taking advantage of 3D transformations implemented within XmMol. But they can also be called automatically by XmMol in replacement of XmMol's own renderer, at any interactive modification of the view, or when a pick is registered.

Such communication was tested for C and Fortran applications. Files of commands can also be parsed directly by XmMol.

CONCLUSIONS AND PERSPECTIVES

The aim of XmMol is to provide an easy starting point for further investigation of X11-based visualization programs. The program "as is" can be used on a wide range of machines as a simple molecular structure survey tool that exhibits some modeling possibilities. Furthermore, various mechanisms allow the user to interface XmMol easily with external applications. External programs can simply use the XmMol visualization facility or can be integrated as new functionalities. Such enhancements could be input/output procedures as well as the implementation of new graphic modes, for example. In this regard, it is possible to build a library of molecular graphic utilities interfaced with XmMol. Future evolution of the program itself entails making a more intensive use of shared memory, because it will improve the communication with external programs.

Run files of the program and the communication library are available on a nonprofit basis to any academic institution. Developers interested in interfacing external applications can also contact the author.

REFERENCES

1 Silla, E., Vilar, F., Nilsson, O., Pascal-Ahuir, J.L., and Tapia, O. Molecular volumes and surfaces of biomolecules via GEPOL: A fast and efficient algorithm. *J. Mol. Graph.* 1990, **8,** 168–172

2 Flower, D.R. Improved ribbon drawing programs. *J. Mol. Graph.* 1991, **9,** 257–258

3 Gertner, B.J., Whitnell, R.M., and Wilson, K.R. Fast space-filling molecular graphics using dynamic partitioning amongst parallel processors. *J. Mol. Graph.* 1991, **9,** 139–147

4 Kraulis, J. MOLSCRIPT: A program to produce both detailed and schematic plots of protein structures. *J. Appl. Cryst.* 1991, **24,** 946–950

5 Evans, S.V. SETOR: Hardware lighted three-dimensional solid model representations of macromolecules. *J. Mol. Graph.* 1993, **11,** 134–138

6 Bacon, D.J. and Anderson, W.F. A fast algorithm for rendering space-filling molecule pictures. *J. Mol. Graph.* 1988, **6,** 219–220

7 Cherfils, J., Vaney, M.C., Morize, I., Surcouf, E., Colloc'h, N., and Mornon, J.P. MANOSK: A graphics program for analyzing and modelling molecular structure and functions. *J. Mol. Graph.* 1988, **6,** 155–160

8 Ferrin, T.E., Huang, C.C., Jarvis, L.E., and Langridge, R. The MIDAS display system. *J. Mol. Graph.* 1988, **6,** 13–27

9 Schomburg, D. and Reichelt, J. BRAGI: A comprehensive protein modeling program system. *J. Mol. Graph.* 1988, **6,** 161–165

10 Vriend, G. WHAT IF: A molecular modeling and drug design program. *J. Mol. Graph.* 1990, **8,** 52–56

11 Fujii, I., Morimoto, Y., Higuchi, Y., and Yasuoka, N. XELE—a polypeptide model-building program for a graphics workstation. *J. Mol. Graph.* 1992, **10,** 185–189

12 Pattou, D., Bach, L.H., Kocher, J.P., and Maigret, B. MOL3D—a molecular and interactive program for molecular modeling and conformational analysis. II. Extended modules. *J. Mol. Graph.* 1992, **10,** 241–246

13 Vassylyev, D.G. and Adzhubei, A.A. Tek FRODO: A new version of FRODO for tektronix graphics stations. *J. Mol. Graph.* 1992, **10,** 101–104

14 Shindyalov, I., Chang, W., and Bourne, P. *PDBview,* version 1.1. Columbia University, New York, 1993

15 Sayle, R., and Bissel, A. *Rasmol.* Biocomputing Research Unit, University of Edinburgh, Edinburgh, 1993

16 *XMol,* version 1.3.1. Minnesota Supercomputing Center, Minneapolis, MN, 1993

17 Foley, J.D. and Van Dam, A. *Fundamental of Interactive Computer Graphics.* Addison-Wesley, Reading, MA, 1982

18 Newman, W.M. and Sproul, R.F. *Principles of Interactive Computer Graphics,* McGraw Hill, Tokyo, 1981

19 Lavery, R., Sklenar, H., Zakrzewska, K., and Pullman, B. The flexibility of the nucleic acids. II. The calculation of internal energy and applications to mononucleotide repeat DNA. *J. Biomol. Struct. Dyn.* 1986, **3,** 989–1014

20 Carson, M. Ribbon models of macromolecules. *J. Mol. Graph.* 1987, **5,** 103–106

21 Tuffery, P., Vaney, M.C., Mornon, J.P., and Hazout, S. Forme: An interactive package for protein backbone deformation. *J. Mol. Graph.* 1991, **9,** 175–181

22 Sklenar, H., Etchebest, C., and Lavery, R. Describing protein structure: A general algorithm yielding complete helicoidal parameters and a unique overall axis. *Proteins* 1989, **6,** 46–60