Journal of Computer-Aided Molecular Design, **0:** 871–881, 2002.
KLUWER/ESCOM
© 2003 *Kluwer Academic Publishers. Printed in the Netherlands.*

Short Communication

AstexViewer^{TM†}: a visualisation aid for structure-based drug design

Michael J. Hartshorn*
Astex Technology Ltd., 436 Cambridge Science Park, Milton Road, Cambridge, CB5 9QA, UK

Received 13 September 2002; accepted in final form 26 November 2002

Key words: Structure-based drug design, web-based molecular graphics, molecular viewer, protein-ligand complexes

Summary

AstexViewer™ is a Java molecular graphics program that can be used for visualisation in many aspects of structure-based drug design. This paper describes its functionality, implementation and examples of its use. The program can run as an Applet in a web browser allowing structures to be displayed without installing additional software. Applications of its use are described for visualisation and as part of a structure based design platform. The software is being made freely available to the community and may be downloaded from http://www.astex-technology.com/AstexViewer.

Introduction

For many years molecular graphics has played a key role in our understanding of chemical structures. Initially, three-dimensional graphics were restricted to high performance graphics workstations and specialised software [1]. Over the last decade the Internet has become a widely used means for distributing information. Molecular graphics programs have been developed to display chemical structures downloaded over the Internet. These include RasMol [2], Chime [3] and Weblab Viewer [4]. These programs all need to be installed on the machine on which they are run. This presents many system-maintenance problems and requires software developers to ensure that their programs run on the variety of computers in use today.

The development of web based programming languages, such as Java [5], has allowed embedded programs to be written for molecular graphics. Until recently, these lacked the performance or functionality to be useful for structure based drug design. The most well known of these is WebMol [6], and it is principally used for visualisation of protein structures.

The key goal in the design of AstexViewer was to develop a system that allows a scientist to use a web browser to directly view molecular structures. Two main benefits of this approach are that almost everyone knows how to use a web browser and that the locations of structural information can be provided using links in HTML pages or e-mail messages. This viewer has to be fast enough to provide interactive manipulation of the 3D structure, and flexible enough to allow many structures to be loaded, and complex information to be shown.

Many graphics programs provide sophisticated graphical styles, such as shaded spheres and surfaces. We believe that coloured lines and text are sufficient for the majority of molecular graphics. For this reason AstexViewer provides only simple line drawing and text, but these can be used to construct wireframe molecular surfaces, electron density maps and other features.

[†]AstexViewer is a trademark of Astex Technology Ltd.
*Correspondence: Tel: +44 1223 226212; Fax: +44 1223 226201; E-mail: m.hartshorn@astex-technology.com

Key features

Protein/ligand complexes with electron density

Structure-based drug design is an accepted methodology for drug discovery. We have streamlined procedures for determining the structures of protein/ligand complexes by X-ray crystallography as we develop a new lead series. Once this process is in place the inspection of the structures and electron density maps becomes a key activity. Traditionally one would have to use software such as O [1] or Quanta [4] to examine these electron density maps. This software requires a Unix workstation and specialist training to be used effectively, making it difficult for medicinal chemists to examine the binding modes of lead compounds and their electron densities. This places a significant obstacle in an integrated structure-based design process; overcoming this obstacle was one of the main driving forces for the development of AstexViewer.

AstexViewer provides a simple authoring mechanism that allows protein/ligand structures and electron density to be displayed on any computer with a web browser. This gives all chemistry and modelling teams immediate access to key structural information relating to medicinal chemistry projects. Public domain structures and other in house structures can be overlaid to emphasize the important structural features of the complex. This is discussed in more detail in the Examining electron density section.

Multiple molecules

AstexViewer can load and display multiple molecules simultaneously. These can be turned on and off using the scripting language, which allows overlays of several protein structures and ligands to be prepared easily. AstexViewer can read PDB files, MDL mol files and basic coordinate information from Sybyl Mol2 files. An overlay of several carbonic anhydrase complexes is shown in Figure 1.

The display of each structure can be controlled from an HTML form button embedded in the web page. These have JavaScript actions associated with them that execute AstexViewer scripting language commands. The HTML commands shown below could be added to a web page to embed AstexViewer and control the display of a loaded structure. The example uses AstexViewer scripting language that is discussed in detail in the documentation section of the website http://www.astex-technology.com/AstexViewer/documentation/.

```
<applet width=400 height=300 archive=AstexViewer.jar
code=MoleculeViewerApplet name=av>
<param name=script value="molecule load mol1
    '1bnu.pdb'">
</applet>
...
<input type=checkbox checked name=mol1
onClick='document.av.execute("molecule display mol1
    toggle;")'>1BNU<br>
```

The applet definition creates an instance of AstexViewer with name `av`. This is referred to as `document.av` from JavaScript. The `execute()` method on the applet executes scripting commands. In this example the command toggles the display of the molecule named `mol1`.

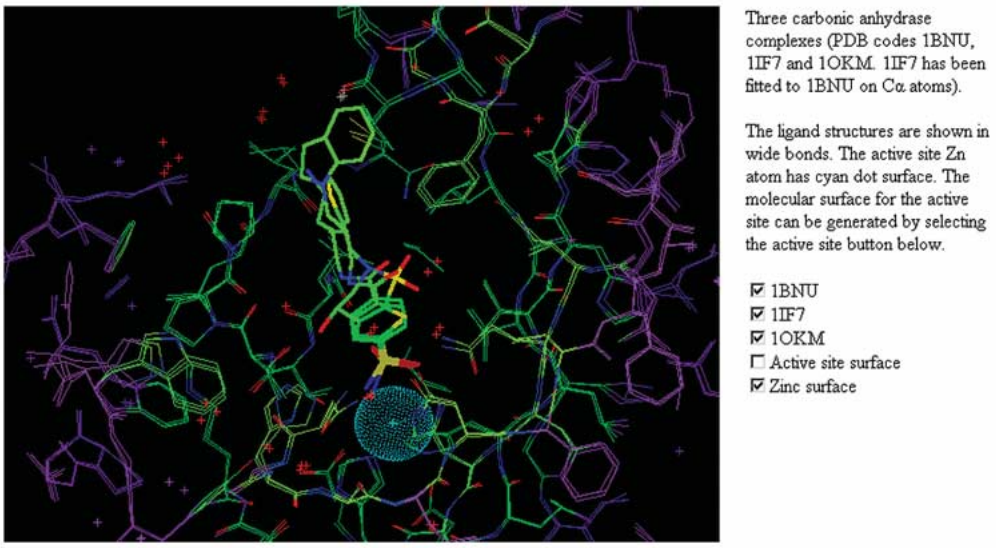
Symmetry generation

Crystal packing effects are important in understanding the binding of small molecules to many protein structures. These effects are not immediately obvious when one simply looks at the structure of a complex obtained from the Protein Data Bank [7]. For instance, consider the complex of Stromelysin-1 and an inhibitor (PDB code 1SLN) [8] shown in Figure 2A. The structure shows a number of hydrogen bonding interactions between the protein and ligand, but the overall structure of the complex looks unusual; for example, the phenyl group appears to be completely exposed to solvent.

The same structure is shown in Figure 2B but with the symmetry related atoms within 12 Å of the ligand also displayed. It can be seen that the exposed phenyl group now rests within a binding pocket formed by symmetry related molecules. For this reason, symmetry contacts are generated whenever one centres on a particular atom.

AstexViewer can automatically display the symmetry environment of a position in the molecule. The standard CCP4 [9] library of symmetry operators is used for generating the symmetry equivalents. If an electron density map is being displayed, symmetry information is taken from this; otherwise it is read from the CRYST1 record of the PDB file that is being displayed.

The symmetry generation process is shown in Figure 3. Determination of a bounding sphere that encloses the molecule increases the speed of symmetry generation. This bounding sphere is transformed by each of the symmetry operators, and the atom positions are only constructed if the bounding sphere


Three carbonic anhydrase complexes (PDB codes 1BNU, 1IF7 and 1OKM. 1IF7 has been fitted to 1BNU on Cα atoms).

The ligand structures are shown in wide bonds. The active site Zn atom has cyan dot surface. The molecular surface for the active site can be generated by selecting the active site button below.

☑ 1BNU
☑ 1IF7
☑ 1OKM
☐ Active site surface
☑ Zinc surface

Figure 1. Overlay of several carbonic anhydrase complexes (PDB codes 1BNU, 1IF7 and 1OKM). JavaScript buttons control the display of each structure and the generation of molecular surfaces.

would overlap the sphere in which we are generating symmetry contacts. This means that most symmetry operators can be discarded after one matrix transformation. Consequently, there is almost no overhead for generating the symmetry environment of a position in the molecule.

Scripting Language

AstexViewer supports a scripting language for controlling aspects of the display and loading molecules and electron density maps. Scripting language commands can be supplied at start-up time or sent to the AstexViewer applet from JavaScript page controls. This second feature allows the creation of web applications that manipulate molecular structures and other objects. A semicolon character (;) terminates all commands.

The scripting language is described in detail on the website. An example is given below to illustrate the range of functionality:

```
#
# Load pdb file 1sln.pdb and centre on the ligand
#
molecule load mol1 '1sln.pdb';
display wide name INH;
centre name INH;
```

Strings may be quoted using the single quote character (') and may contain wildcard characters: * for any characters, ? for any single character and [abc] or [a–z] for ranges of characters. The atom selection expressions are similar to those found in other molecular graphics or modelling programs such as CHARMM [10] and RasMol.

The principle feature of the scripting language is atom selection expressions. These allow sets of atoms to be identified by means of logical expressions in terms of their attributes, for example: id or name, residue name or number, chain name, molecule name or proximity to other groups of atoms. Individual selection statements can be grouped with parentheses and Boolean operators. Examples of atom selection expressions are shown below.

`atom CA`	select all Cα atoms (and Calciums)
`atom CA and aminoacid`	select all Cα atoms
`(atom C or atom CA or atom N) and aminoacid`	select backbone atoms
`name GLY`	select atoms in all Glycine residues
`molecule mol1`	select atoms in mol1
`molecule m*`	select atoms in all molecules whose names begin with m
`chain A`	select atoms in chain A
`sphere 5 around residue 100`	5 Å sphere of atoms around (and including) residue 100

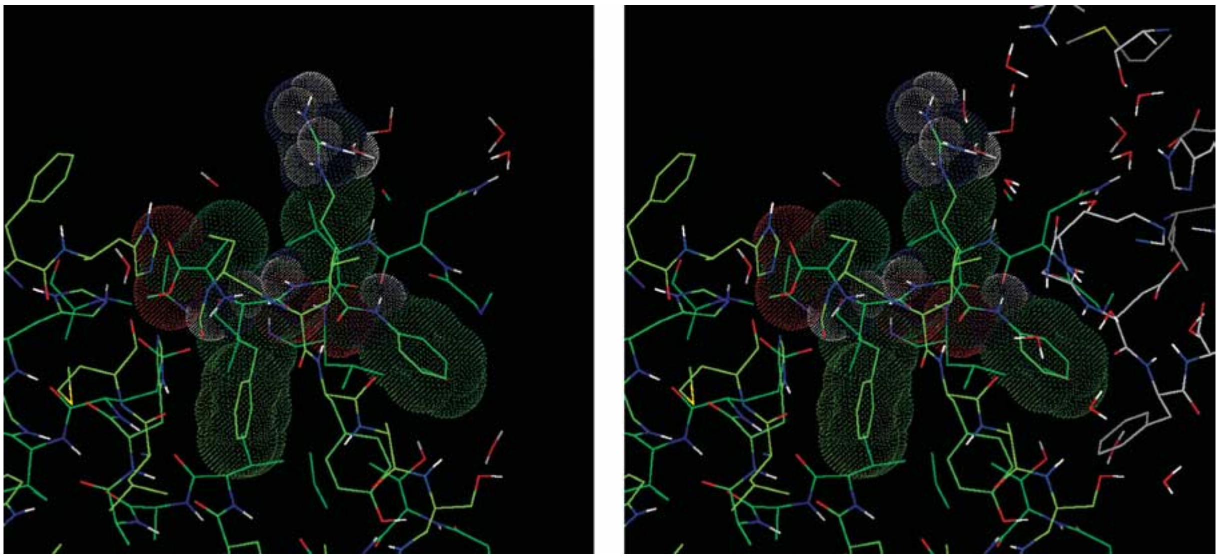


Figure 2. A. Complex of Stromelysin-1 and an inhibitor (PDB code 1SLN). The ligand is shown with a dot surface. B. The same complex as Figure 2A, but with symmetry-related atoms displayed. The binding interactions of the phenyl group are much clearer.

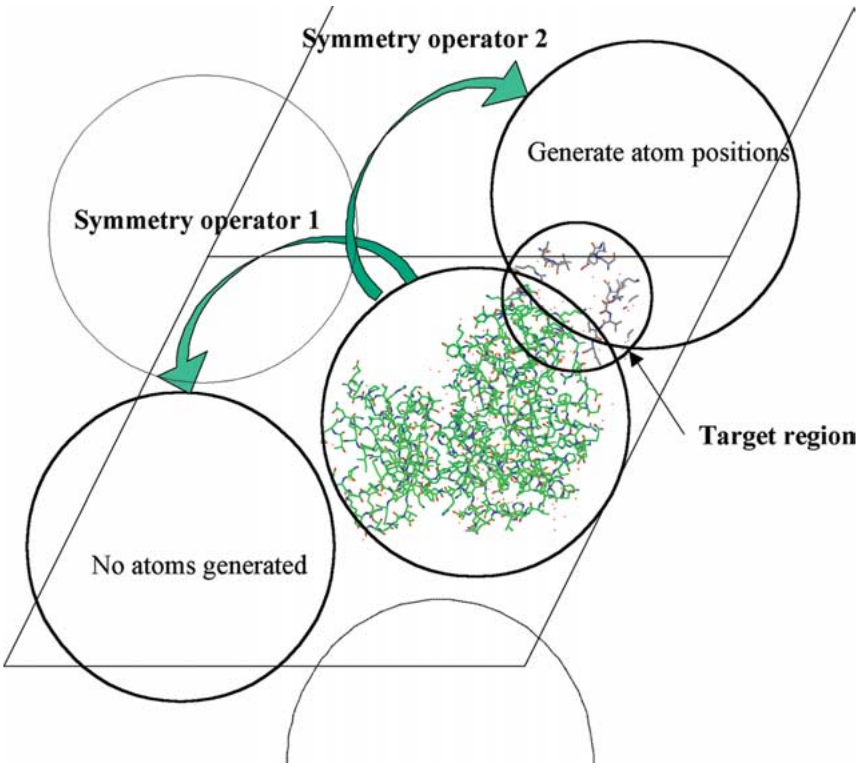


Figure 3. Two-dimensional outline of the symmetry generation process. The bounding sphere for the molecule is shown. When transformed by symmetry operator 1, the bounding sphere does not overlap the target region (i.e. the viewed part of the structure) and so no transformed atoms coordinates are generated. When transformed by symmetry operator 2, the bounding sphere does overlap the target region and so all atom coordinates are transformed. Those within the target region are retained and displayed in grey. Dotted lines indicate the position of the bounding volume when transformed by other symmetry operators.

Once groups of atoms have been selected, they can be modified in various ways: to change their colour or their display style, for example. The atom selection expressions also form the basis of the surface generation commands described in the next section.

Surfaces

AstexViewer can generate Connolly molecular surfaces [11] for whole or partial structures. The surface is generated by measuring the distance to the solvent-accessible surface for a set of points defined on a regular grid. The grid is then contoured at the desired probe radius (usually 1.4 Å) to yield the molecular surface. This process is outlined in Figure 4.

The resulting molecular surface is displayed as a wireframe contour map that allows the underlying molecular structure to be seen. Example molecular surfaces are shown in Figures 5A and 5B for an H-ras p21 mutant with a GTP analogue bound [12]. Figure 5A shows the complete molecular surface for the ligand bound in the active site. Figure 5B shows the molecular surface for those protein atoms that are within 6 Å of the ligand.

Web proxy

Conventional web applications provide forms for specifying the values and operations that one wishes to carry out. When the choices have been made a 'submit' button is pressed and the data is returned to the server program. This program then outputs another page with the results of the process. This model is not appropriate for an interactive application, as the form submission and rendering of the output is very distracting.

To address this problem, AstexViewer provides a web proxy mechanism that enables it to submit requests back to the server from which it was downloaded (this restriction is enforced by the Java Applet security model [5]). This allows interactive operations to be carried out without refreshing the page and makes web pages that include AstexViewer behave much more like a conventional molecular modelling application. The following JavaScript/HTML example shows a function that could be used to provide the next molecular structure in a sequence. The function retrieves the URL of the next structure and then instructs AstexViewer to load it.

```
<script language=javascript>
function get_next_mol(){
  var next_mol =
        document.form.av.fetch('next_mol.cgi');
  document.av.execute('molecule load mol1 ' +
next_mol + ';');
}
</script>
...
<html>
<input type=button onclick='get_next_mol();'>
...
</html>
```

The `fetch()` method allows communication with programs on a remote web server. This remote program might be a database that provides information about a particular in house electron density complex. In this fashion information can be retrieved without relying on database connectivity within the web browser.

Implementation

AstexViewer is written using the Java Programming Language (version 1.1) [5]. Java is a portable bytecode language that can run as an application under practically all operating systems, and as Applets within web browsers.

For a long time the language suffered from poor execution speeds as the bytecode is run by an interpreter. The advent of Just-In-Time compilers, which turn bytecode into native executable code, has transformed the execution speed of Java programs on a number of platforms. Particularly good implementations of Java include those available for Microsoft Windows, within Microsoft Internet Explorer, and on Solaris. Netscape does not offer as good performance as Internet Explorer.

Of particular importance are the interpreters for Microsoft systems as these are prevalent within drug discovery organisations. AstexViewer lets meaningful structure-based content be displayed on any PC within an organisation.

The performance of AstexViewer is best measured in terms of how quickly it can load molecular structures and electron density maps and how quickly it can draw these structures once loaded. To get reasonable performance in both of these areas custom code had to be developed.

File reading performance

Java provides a number of classes for reading file input. However, the performance of these is poor com-

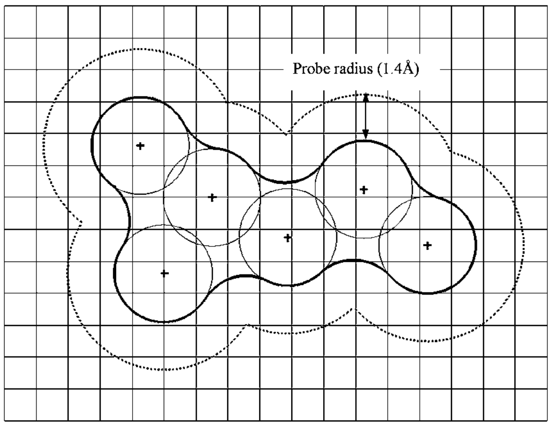


Figure 4. Two-dimensional illustration of the surface generation process. The light circles indicate the VDW radii of the atoms. The dotted line indicates the solvent accessible surface of the example molecule. This is the union of the solvent extend spheres (VDW radius + 1.4 Å). The molecular surface is the volume of space into which no part of a probe solvent sphere can penetrate. This is approximated here by measuring the distance to the nearest point on the solvent accessible surface for each grid point. The resulting grid is contoured at a value equal to the probe radius.

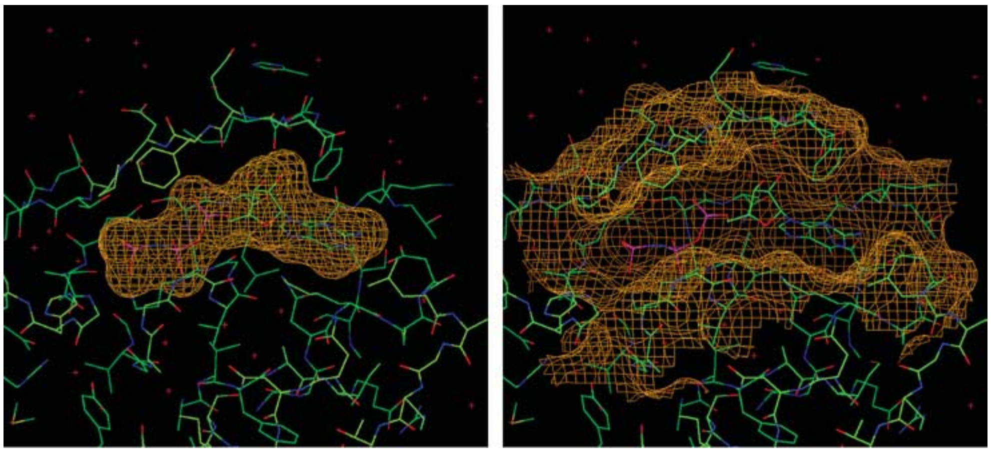


Figure 5. A. Grid based molecular surface for the GTP analogue bound to an H-ras p21 mutant protein structure (PDB code 621P). B. Grid based molecular surface for the binding site of the protein structure in Figure 5A.

pared to the standard input library of the C programming language. The Java input classes are typically geared to reading Unicode (a multibyte character representation), but most structure file formats are ASCII (e.g. PDB, Mol file). This conversion from ASCII characters to Java Unicode representation significantly degrades the performance of reading files.

To address this problem, a very low level file input mechanism was developed. This uses the Java InputStream classes to provide a byte level input mechanism. The file input methods provide their own buffering and their own numeric conversions. Also, the input records are never converted to Unicode strings saving enormous amounts in Java object allocation overhead. In our experience, this low level input class provides file reading performance between 5 and 10 times that of the corresponding methods in the core Java libraries.

Graphics library

The standard Java 1.1 implementation does not provide libraries for 3D graphics. The Java 3D standard does provide access to the OpenGL [13] drawing library, but this is not installed in most web browsers and so could not be used for the graphics rendering within AstexViewer. All graphics operations are performed in an internal framebuffer for maintaining the colours of pixels, and a Z-buffer for performing hidden line removal.

Graphics primitives
As there are no standard 3D libraries available for Java, a graphics library providing simple graphics methods had to be developed. The basic drawing style is straight lines, implemented using the standard Bresenham line drawing algorithm [14]. In addition to straight lines, the images need to be annotated with text for labelling atoms and showing distance and angle monitors. Text rendering uses an internal font so that hidden line removal and depth cueing can be performed on text objects in a similar fashion as for line drawing.

Hidden line removal
It is well known that molecular graphics benefits from a clear indication of which lines are in front of other lines. If this is not done, visually confusing effects arise as lines that are at the 'back' of the 3D image obscure those that are at the 'front'. Hidden line removal is based around a standard Z-buffer algorithm [15] which records the current distance of each pixel from the eye of the observer. Before a pixel colour is updated, the Z-value of the new pixel is compared to that already stored in the Z-buffer. The pixel colour is only updated if the new pixel is closer to the observer than the old pixel.

Colour model
Images are generated by drawing lines and text items into an internal graphics buffer. Copying this buffer onto the screen is another slow step in Java applications. As a consequence it was decided that only 8 bit graphics would be used, rather than 32 bit, as only one quarter as much data is copied to screen. However, the use of 8 bit graphics limits the number of possible display colours to only 256. It has been demonstrated that depth cueing (a process that makes distant lines darker) is important for indicating which parts of a three-dimensional scene are in front of others; depth cueing requires multiple shades of each colour to achieve this. A reasonable compromise was found by using 8 shades of each of 32 colours. The colour ramp starts with the fully saturated colour and progressively dims the colour towards black. The bright shade entries are used for pixels that are nearer the observer and the darker shades for those that are furthest away.

Buffer clearing
During the development of AstexViewer it was observed that clearing the colour and Z-buffers was taking a significant fraction of the time associated with rendering one image. This is partly because Java provides no simple method for simply clearing an array to a particular value; one has to manually clear arrays with a loop over all array entries or use a convoluted approach involving the `System.arraycopy()method`. This is a shame as many computer architectures have highly optimised procedures for clearing blocks of memory.

To work around this, a novel Z-buffer method was devised that does not require clearing every frame. This approach uses a 32-bit integer value for maintaining the Z-buffer values. The simple line graphics used in AstexViewer mean that an 8-bit value (256 depth levels) is sufficient for the depth buffer. Each frame adds 256 to an offset that is used in all Z-buffer comparisons. This has the effect that the Z-buffer can be used without clearing its contents from frame to frame. There are some small overheads in using the offsets but they are far outweighed by the savings in buffer clears (which were amounting to 25% of runtime).

Performance

The result of these new approaches is the Java visualisation system AstexViewer. The class files for the program total 107 kilobytes when compressed into a Java archive (jar file). This means that about 14 copies of the program would fit on one floppy disk. Compare this to the size of most modern molecular graphics programs, which typically occupy 10's of Mb of disk space.

The jar file storage system means that the whole application can download in a few seconds on almost any network connection. The application is smaller than most protein structure files, hence most of the delay in viewing web pages is due to structures downloading.

The software will run on essentially all platforms with a Java interpreter, including Linux, Mac/OS and Windows. However, AstexViewer has been used most extensively as an Applet in Internet Explorer 5.0 on Windows 2000 and performs best on this combination. Running on a 1GHz Pentium III, a 1000 atom protein structure with an electron density map around the bound ligand can be rotated at around 45 frames per second. This performance means that overlays of many structures can be viewed within a web browser and manipulated interactively.

Application: protein/ligand complexes

Examining electron density

The main focus at Astex is high throughput crystallography for solving protein/ligand complexes. We have an automated procedure, AutoSolve® [16], for interpreting electron density maps. This means that it has to be as simple as possible to examine the results generated by protein/ligand studies. Initial versions of AutoSolve® used a standalone program to generate an HTML file that displays the contents of a directory. Later versions generate the pages dynamically from information stored in our corporate database.

The web page that is generated by this program is shown in Figure 6. The controls at the top left of the page allow the protein structure and a variety of ligand solutions to be turned on and off. The controls below this are used to change the display of electron density maps. Multiple electron density maps can be displayed with 3 contour levels per map. As there is no slider control in HMTL forms, the contour levels are set with a menu providing values in the range -5 to $+5$, in 0.1 increments.

The other controls within AstexViewer can be used to measure distances, and assess the quality of the ligand complex, which has proved very useful for checking our in house protein/ligand complexes. The structures can be viewed from a set of links contained in a table, and the whole process carried out very rapidly. AutoSolve® software records the chosen solution in our database, so that the correct structures can be presented to medicinal chemists and molecular modellers.

Several examples of electron density maps can be found on the AstexViewer website (http://www.astex-technology.com/AstexViewer/electron_density/). This page contains F_o-F_c OMIT maps that were calculated for 70 complexes from a previous docking validation study [17]. The electron density maps were generated for all complexes within the validation set whose structure factors were deposited at the Protein Databank.

Comparison of protein/ligand complexes

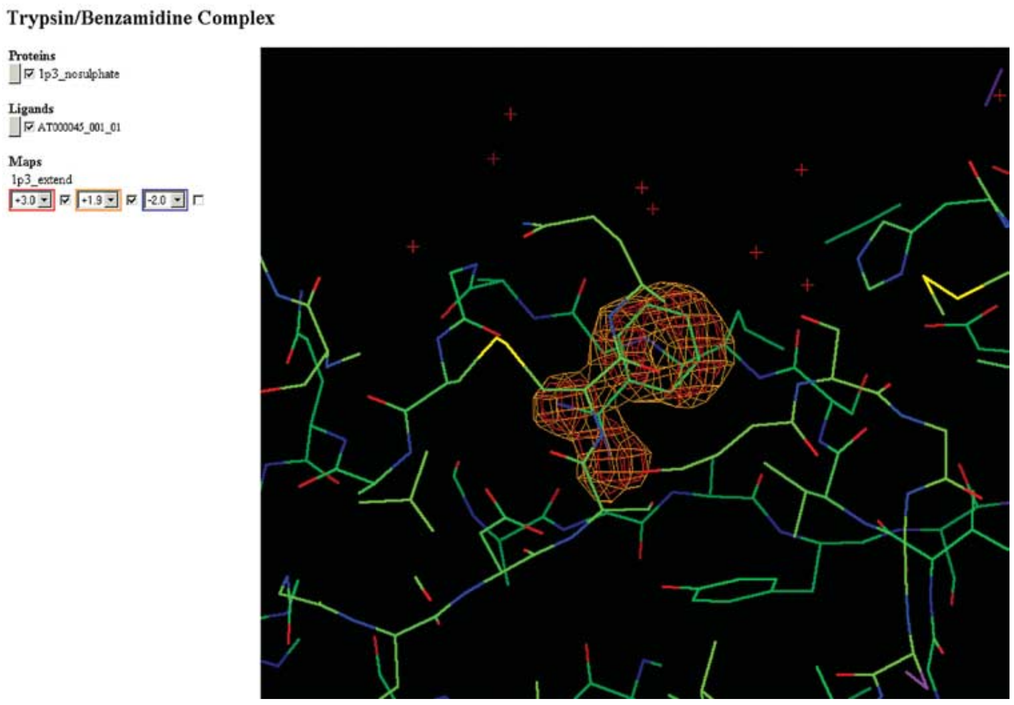
AstexViewer can load many protein and ligand structures simultaneously. As mentioned previously, the display of these structures can be manipulated using web page controls. This makes it straightforward to develop web-based applications for displaying and examining the binding of many ligands to the same protein structure, a key activity for structure based drug design.

In a previous study we used AstexViewer for comparing the solutions of docking experiments against multiple conformations of the same protein structure [18]. These experiments were performed to understand what factors affect the choice of protein structures for virtual screening. A set of neuraminidase complexes was chosen from the PDB (PDB codes 1MWE, 1INV, 1INW, 1F8B, 1F8C, 1F8D, 1F8E, 1NNC, 1INF, 1B9T, 1B9V, 1A4Q, 2QWI, 2QWJ, 2QWK). The ligands of each complex were docked against the protein structures of each complex and the quality of the solutions was assessed. This cross docking experiment showed that some of the neuraminidase protein conformations could accommodate all ligand structures while others could only recreate the binding mode of their original ligand. The overlaid complexes were examined with AstexViewer to identify the reasons for this. The overlay is available from the AstexViewer website (http://www.astex-technology.com/AstexViewer/neuraminidase).

Overlaid structures provide a very clear picture of the key binding interactions. This insight can be used for developing pharmacophores or directly guiding the design process for novel inhibitors. Figure 7 shows a set of neuraminidase complexes overlaid on the active site. The key features of the active site are highly conserved and the main interactions of the ligands are clearly visible. Most of the ligands contain a carboxylic acid, which forms a strong interaction with the three ARG residues of the active site. In the previous study we showed that there was some ‘hypersensitivity’ of the docking results to the PDB structure that was used to dock against. Examination of the binding modes show that interactions with the sidechains of residues ASP 151 and ARG 152 contribute to the ‘hypersensitivity’ of the docking results against structures 1MWE and 1INF (or 2QWJ).

Conclusions

AstexViewer has proved enormously useful for conveying structural information to scientists within our organisation. Electron density maps can be examined and multiple structures displayed on any computer attached to the company network. Previously, these tasks would have required a Unix workstation and a person trained in using crystallography software. Maintenance of the system is straightforward



Trypsin/Benzamidine Complex

Proteins

☑ 1p3_nosulphate

Ligands

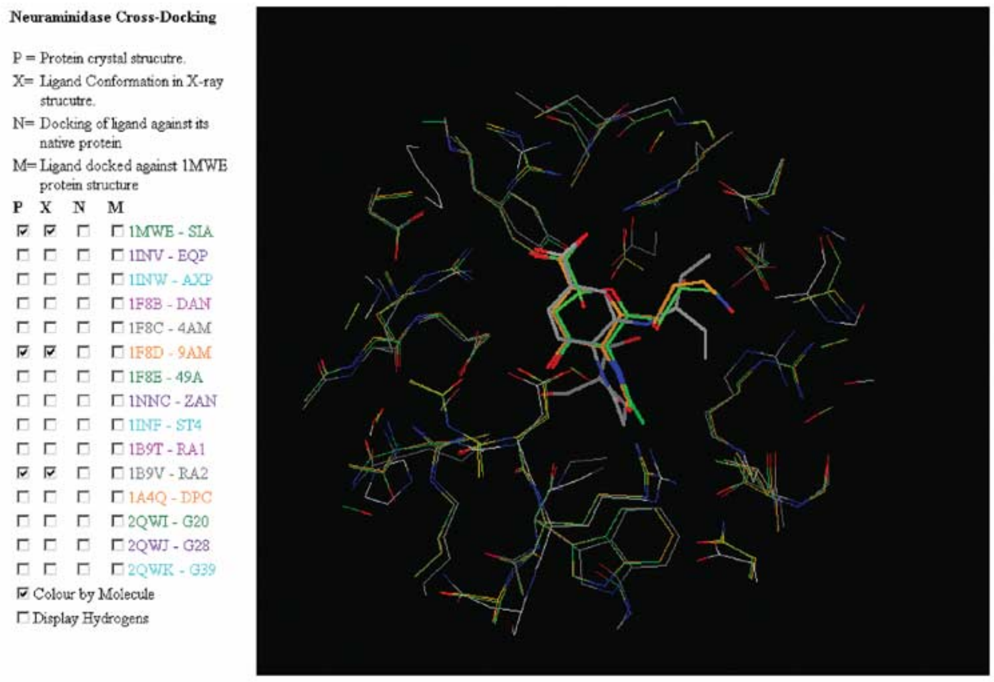
☑ AT000045_001_01

Maps

1p3_extend

+3.0 ☑ +1.9 ☑ -2.0 ☐

Figure 6. Web page for displaying protein/ligand complex with electron density maps. The web buttons at the top left turn the protein and ligand structures on and off. The coloured buttons underneath control the display of three different contour levels. The electron density has been clipped to the ligand for clarity, but much larger maps can be displayed.



Neuraminidase Cross-Docking

P = Protein crystal strucutre.

X= Ligand Conformation in X-ray strucutre.

N= Docking of ligand against its native protein

M= Ligand docked against 1MWE protein structure

P	X	N	M	
☑	☑	☐	☐	1MWE - SIA
☐	☐	☐	☐	1INV - EQP
☐	☐	☐	☐	1INW - AXP
☐	☐	☐	☐	1F8B - DAN
☐	☐	☐	☐	1F8C - 4AM
☑	☑	☐	☐	1F8D - 9AM
☐	☐	☐	☐	1F8E - 49A
☐	☐	☐	☐	1NNC - ZAN
☐	☐	☐	☐	1INF - ST4
☐	☐	☐	☐	1B9T - RA1
☑	☑	☐	☐	1B9V - RA2
☐	☐	☐	☐	1A4Q - DPC
☐	☐	☐	☐	2QWI - G20
☐	☐	☐	☐	2QWJ - G28
☐	☐	☐	☐	2QWK - G39

☑ Colour by Molecule

☐ Display Hydrogens

Figure 7. Overlay of a selection of protein and ligand structures from a cross-docking experiment. The protein and ligand structures can be displayed independently from the web page controls. Other buttons control whether hydrogens are displayed and how the molecules are coloured.

as there are no client side tools to install. The Applet, structures and electron density maps download in seconds.

More recently we have also used AstexViewer to construct an entirely web-based visualisation system for our virtual screening platform. Using the system, sets of compounds can be selected from a large in house database, and docked against a target protein using GOLD [19] running on our in house Linux cluster. The interface allows selection and sorting of docking results and these can be overlaid on the protein structure using AstexViewer. This has proved enormously useful, letting us bring the results of this powerful modelling technique to the desktop of all interested users.

Other programs such as RasMol and its derivative, Chime, provide some capabilities in this area, but they do not support some of the key features required. In particular, they have no direct methods for displaying contoured electron density maps, although this can be achieved by loading the contours as atoms and bonds from a structure file. The main features that make AstexViewer useful for structural studies is the speed with which it can load molecules, and the fact that it can support the display of many structures from separate files and locations simultaneously. The powerful atom selection expressions enable the key features of a molecular structure to be identified with ease.

WebLab viewer provides a Microsoft Windows compatible ActiveX control for embedding structure visualisation in web pages. This allows the creation of sophisticated graphical representations and has the ability to take advantage of compiled programming languages and hardware graphics acceleration. However, the ActiveX component model is extremely closely tied to Microsoft operating systems, and this places an unnecessary restriction on the kinds of computers that can use the software. This application also requires the download and installation of several megabytes of program information before structures can be viewed. This should be compared with the small size of AstexViewer and the automated installation procedure.

It is hoped to extend the functionality of AstexViewer in future versions. Enhancements may include the ability to read additional file formats, such as XML encodings of molecular structures. The high frame rates that are obtained for line drawings suggest that more sophisticated rendering styles may be possible within AstexViewer. These could include shaded triangles that could be used for displaying protein schematics or solid molecular surfaces.

Software availability

We have made the Java byte code for AstexViewer freely available to academic and industrial users. The software may be used for internal research purposes and to display structures on external websites, provided that Astex Technology is acknowledged. The details of the license agreement and the software may be obtained from http://www.astex-technology.com/AstexViewer. The software is also available in recent versions of the CCP4 crystallography package. Live versions of the figures in this paper along with other examples can also be found on the website. Detailed description of the scripting language and the JavaScript interface can also be found on the website.

Acknowledgements

The author would like to thank Ian Tickle, Andrew Sharff, Marcel Verdonk, Chris Murray and Paul Watson for providing suggestions for useful functionality.

References

1. Jones, T.A., Zou, J.Y., Cowan, S.W. and Kjeldgaard, M., Acta Cryst A47 (1991) 110–119.
2. Sayle R. A. and Milner-White E. J., Trends in Biochemical Science (TIBS), Vol. 20, No. 9 (1995) 374.
3. MDL Information Systems, Inc., 14600 Catalina Street, San Leandro, CA 94577, USA.
4. Accelrys Inc., 9685 Scranton Road, San Diego, CA 92121-3752, USA.
5. Sun Microsystems, Palo Alto, CA (http://www.javasoft.com/).
6. Walther D. Trends Biochem Sci. 22 (1997) 274–275.
7. Berman H. M., Westbrook J., Feng Z., Gilliland G., Bhat T. N., Weissig H., Shindyalov I. N., Bourne P. E., Nucleic Acids Res 28 (2000) 235–242.
8. Becker J. W., Marcy A. I., Rokosz L. L., Axel M. G., Burbaum J. J., Fitzgerald P. M. D., Cameron P. M., Esser C. K., Hagmann W. K., Hermes J. D., Springer J. D., Protein Sci. 4 (1995) 1966.
9. Collaborative Computational Project, Number 4. Acta Cryst. D50 (1994) 760–763.
10. Brooks B. R., Bruccoleri R. E., Olafson B. D., States D. J., Swaminathan S. and Karplus M. J., Comp. Chem. 4 (1983) 187–217.
11. Connolly M. L., Science 221 (1983) 709–713.
12. Pai E. F., Krengel U., Petsko G. A., Goody R. S., Kabsch W., Wittinghofer A., EMBO J. 9 (1990) 2351.

13. The OpenGL Programming Guide, 2nd ed., SGI inc., Addison-Wesley, Reading, MA, 1997.
14. J. E. Bresenham, IBM Systems Journal 4 (1965) 25–30.
15. Catmull E. E., Ph.D. Thesis, A Subdivision Algorithm for Computer Display of Curved Surfaces, Dept. of CS, U. of Utah (1974).
16. Tickle I. J., Hartshorn M. J., Jhoti H., unpublished results.
17. Nissink J. W. M, Murray C. W., Hartshorn M. J., Verdonk M. L., Cole J. C., and Taylor R., Proteins, accepted for publication.
18. Birch L., Murray C. W., Hartshorn M. J., Tickle I. J. and Verdonk M. L., J. Comput.-Aid. Mol. Des. 16 (2002) 855–869.
19. Jones G., Willett P., Glen R.C., Leach A. R., and Taylor R., J. Mol. Biol. 263 (1997) 727–748.