jV version 3.3

User's Guide

30 October 2006

Contents

1. Introduction

- jV version 3 (jV3) is an application to display three-dimensional structure of protein and nucleic acid molecules. jV3 is developed in Java and uses JOGL API (https://jogl.dev.java.net/) which is a Java binding for OpenGL for 3D display processing. Therefore, jV3 runs on a platform on which the Java Runtime Environment (JRE) and JOGL are available (includes Windows, Mac OS X, and some typical Linux distributions). It is also possible to use jV3 on a web browser as a Java applet.
- The molecule file to be read is a Protein Data Bank (PDB) format file and a PDBML file which is a XML version of PDB. In addition to molecules, three-dimensional image of arbitrary polygons can be displayed together, where the polygon file to be read is a predefined XML file. Multiple molecular and polygon images are displayed at the same time, and can be operated individually. It is also possible to process an animation for molecules.
- The interactive mouse operations such as rotation or translation to molecular and polygon images are possible and display style of their image can be operated from the GUI menu bar. In addition, a command line interface is also available. GUI design and command syntax follows those in Rasmol (http://www.openrasmol.org/) with some extensions. The command line interface provides more detailed operations than GUI.

2. System requirements

jV3 runs on the Java Runtime Environment (JRE) with JOGL API. The version requirements are as follows.

- JRE (includes Java Plug-in) 1.4.2 or later (recommend 1.4.2_12)
  (http://java.sun.com/j2se/1.4.2/index.html)
- JOGL API 1.0, 1.1.1 or JSR-231 1.0.0 (recommend JSR-231)
  (https://jogl.dev.java.net/)

These softwares are available on typical operation systems such as Windows(2000/XP), Mac OS X 10.3 or later, and Red Hat Linux 7.3 or later.

3. Installing JOGL API

3.1. Automatic install

In this section, it is assumed that JRE 1.4.2 or later has been installed on your computer. Installing JOGL is just deploying several files to JRE. A tool that automatically performs the JOGL installation with a graphical user interface is available at http://ef-site.hgc.jp/eF-site/jV3.html. The recommendation version is JSR-231 1.0.0.

3.2. Manual install

It is also possible to install JOGL manually. According to the kind of your operation system, you should get the following files from https://jogl.dev.java.net/.

	Windows	Mac OS X	Linux
version 1.0	jogl.jar jogl.dll jogl_cg.dll	jogl.jar libjogl.jnilib	jogl.jar libjogl.so libjogl_cg.so
version 1.1.1	jogl.jar jogl.dll jogl_cg.dll	jogl.jar libjogl.jnilib libjogl_cg.jnilib	jogl.jar libjogl.so libjogl_cg.so
JSR-231 1.0.0	jogl.jar jogl.dll jogl_cg.dll jogl_awt.dll	jogl.jar libjogl.jnilib libjogl_cg.jnilib libjogl_awt.jnilib	jogl.jar libjogl.so libjogl_cg.so libjogl_awt.so libjogl_drihack.so

1) Windows

The folder that JRE has been installed is denoted by `<JRE>`. It depends on the environment, however it is typically as follows.

```
<JRE>=C:¥Program Files¥Java¥j2re1.4.2_12
```

The file 'jogl.jar' should be copied to `<JRE>¥lib¥ext`, and the other files should be copied to `<JRE>¥bin`.

2) Mac OS X

When operating from the command terminal, you should copy all files to `/Library/Java/Extensions`. When operating through the graphical interface, the above destination folder is shown as '`Macintosh HD/Library/Java/Extensions`'. In this operation, the password for root is required.

3) Linux

The directory that JRE has been installed is denoted by `<JRE>`. It depends on the environment, however it is typically as follows.

```
<JRE>=/usr/java/j2sdk1.4.2_12/jre
```

The file 'jogl.jar' should be copied to `<JRE>/lib/ext`, and the other files should be copied to `<JRE>/lib/i386`.

4. Tutorials

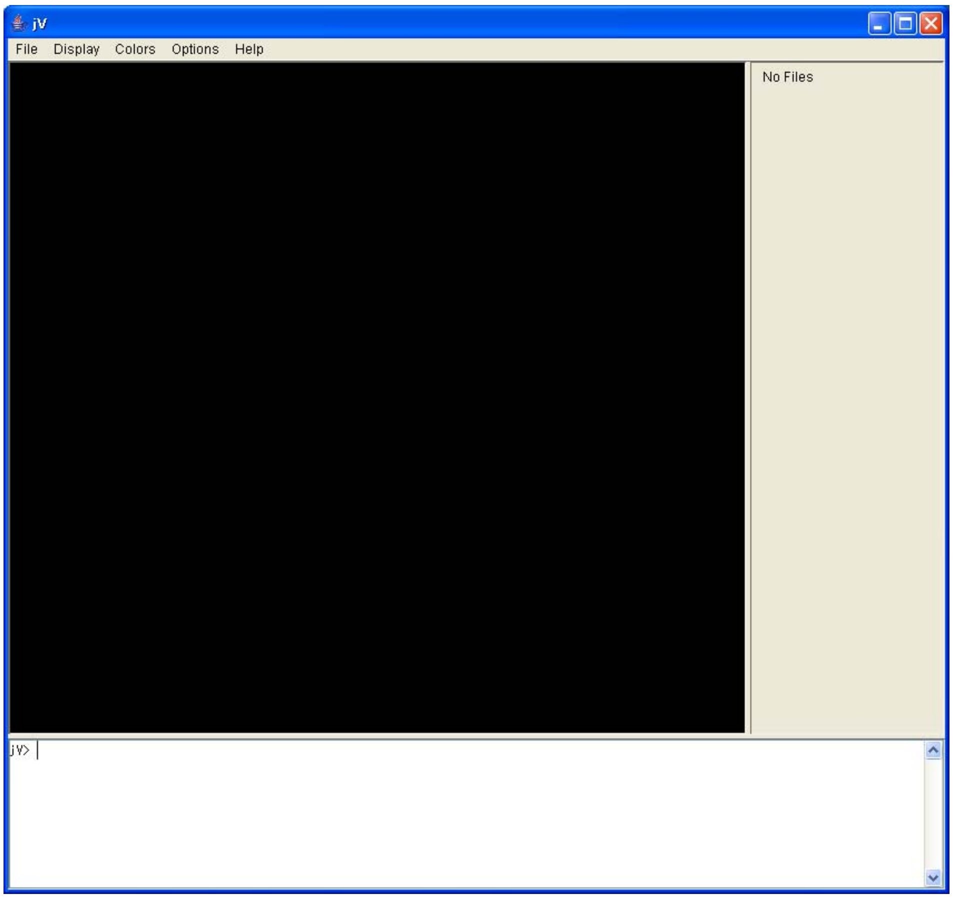
4.1. Starting up

It is assumed that Java Runtime Environment and JOGL have been appropriately installed. Once you extract the binary distribution file of jV3, the directory of the name, jV_(version), is generated (Ex. jV_3_3). The version is assumed to be 3_3_x hereafter. The application starts up by double-clicking the file 'jv_3_3_x.jar' in the jV_3_3_x directory or executing the following command from the command line interface.

```
    java -jar jv_3_3_x.jar
```

Figure 1 shows the start-up screen. The window has a 3-panel structure, and the upper-left panel with the black background is a 3D image display area. The status of files that have been loaded to the application is shown in the upper-right panel and the lower panel is a command-input and a message-output area.

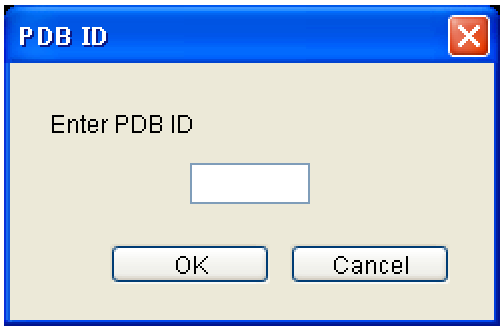
Figure 1. Start-up screen.



4.2. Displaying molecules

If you are working in the environment that is getting on the Internet and know the PDB ID of the molecule that you want to display, the simplest way is as follows. A dialog shown in Figure 2 is opened by selecting the [File] – [Open – Remote] – [PDB ID] menu item. Enter a PDB ID '12as', for example, to the text field and click the 'OK' button, then the PDBML file is read through the Internet and the molecule is displayed. The URL where the PDBML file is retrieved from is specified in the 'properties.txt' file in the jV_3_3_x directory. A molecule is displayed in a wire frame model with colored by atom type in the initial state.

Figure 2. Dialog to open the molecule specified by PDB ID.



Let us next attempt to load an arbitrary file on the Internet. Here we use 'http://ef-site.hgc.jp/eF-site/jV/1yec.pdb' as an example. The file is a conventional PDB file whose PDB ID is 1yec. In this case we choose the [File] – [Open – Remote] – [PDB] menu item. Then the open-remote dialog is appeared. Enter the above URL to the text field at the top of the dialog and click the 'OK' button and the molecule is read in and displayed. In jV3, each molecule is displayed according to the space coordination described in its file. Therefore, when loading and displaying more than one molecule together, they can be viewed with an appropriate separation in some cases; however, they may be displayed far separately or may be overlapped depending on the combination. In any case, when adding a new file, the position of the viewpoint is automatically controlled for the whole to be displayed.

The coordinate system is the right hand system with the x-y plane being on the screen (the x-axis is a horizontal line with positive values to the right and the y-axis is a vertical line with positive values up). Although, the above examples are about operations for remote files, a PDB file on your local hard disk can also be opened from the [File] – [Open – Local] – [PDB] menu item.

4.3. Operating molecules

Interactive mouse operations are possible for 3D images. By left-dragging the mouse on the 3D-display panel, molecules are rotated around the x-axis or y-axis. By left-dragging the mouse while holding the Shift key or Alt key on the keyboard, molecules are translated along the z-axis. All mouse operations are as follows (as for the Macintosh environment, the Command key + left-drag substitutes for the right-drag).

Mouse operations	Actions
Left-drag	The image is rotated around the x (for a vertical motion) and y (for a horizontal motion) axes.
Right-drag	The image is translated on the x-y plane.
[Alt or Shift] + left-drag	The image is translated along the z-axis.
[Alt or Shift] + right-drag	The image is rotated around the z-axis.
[Ctrl] + left-drag	The z-clipping plane is shifted when [Option]-[Slab] is checked.
Left-click	An atom of the molecule displayed on the screen is picked.

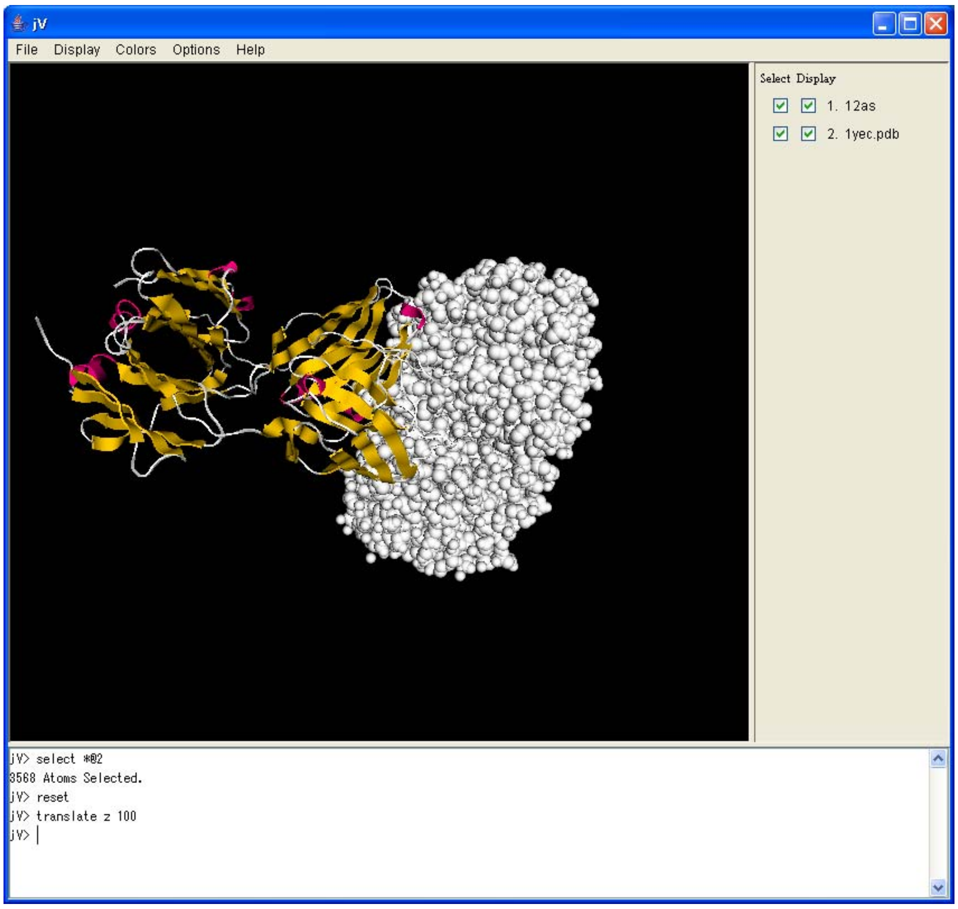
In the upper-right panel in the application window (see Figure 3), the file names that have been loaded to the application are listed, where sequential integers are automatically assigned to the individual file. Each integer is a file ID that can be used to specify the file in the command line interface. In addition, two checkboxes are attached to each file. If you change the status of a 'display' checkbox, the corresponding molecule is switched to be displayed or not to be displayed. If you change the status of a 'select' checkbox, the corresponding molecule is switched to be a target of the mouse operations or not. For example, set the both 'display' checkboxes on, and the both 'select' checkboxes off. In this case, either molecule does not move against mouse operations.

Subsequently set the 'select' checkbox of file 1 on. In this case, mouse operations work for the molecule 1. Thus, each file is assigned either selected or not selected, and transform operations, such as rotation or translation, act only on selected files. On the other hand, every atom in a molecule is also assigned either selected or not selected, independently of whether the molecule itself is selected or not. Operations that modify display models or colors work for selected atoms. To see this, choose the [Display] – [Spacefill] menu item. Then the both molecules are displayed with the space fill model because all atoms are initially selected. In the same way, if you choose the [Colors] – [Monochrome] menu item, the both molecules are colored white. These operations are independent of the status of 'select' checkboxes mentioned above.

Let us next try a more detailed operation. Type 'select *@2' in the command line area and press Enter key. This command means to select all atoms contained in the file 2 (the usage of the select command is summarized in the reference manual included in the binary distribution of jV3). Subsequently, choose the [Display] – [Cartoon] menu item. As a result, only the molecule 2 is displayed in the cartoon model. Furthermore select the [Colors] – [Structure] menu item, and each residue of the molecule 2 is colored according to its secondary structure type (see Figure 3).

In order to reproduce the same view as that in Figure 3, first execute 'reset' command. It restores the position of the viewpoint and the transform of every file to the initial state. Note that the command line interface tries to complement a command word when the Tab key is pressed. If you enter 'res' to the command line and subsequently press the Tab key, the command 'reset' is automatically entered. Next set the both 'select' checkboxes on, and execute 'translate z 100' command. This command means to translate the molecules along the z-axis by 100 angstroms.

Figure 3. An example of molecular images.



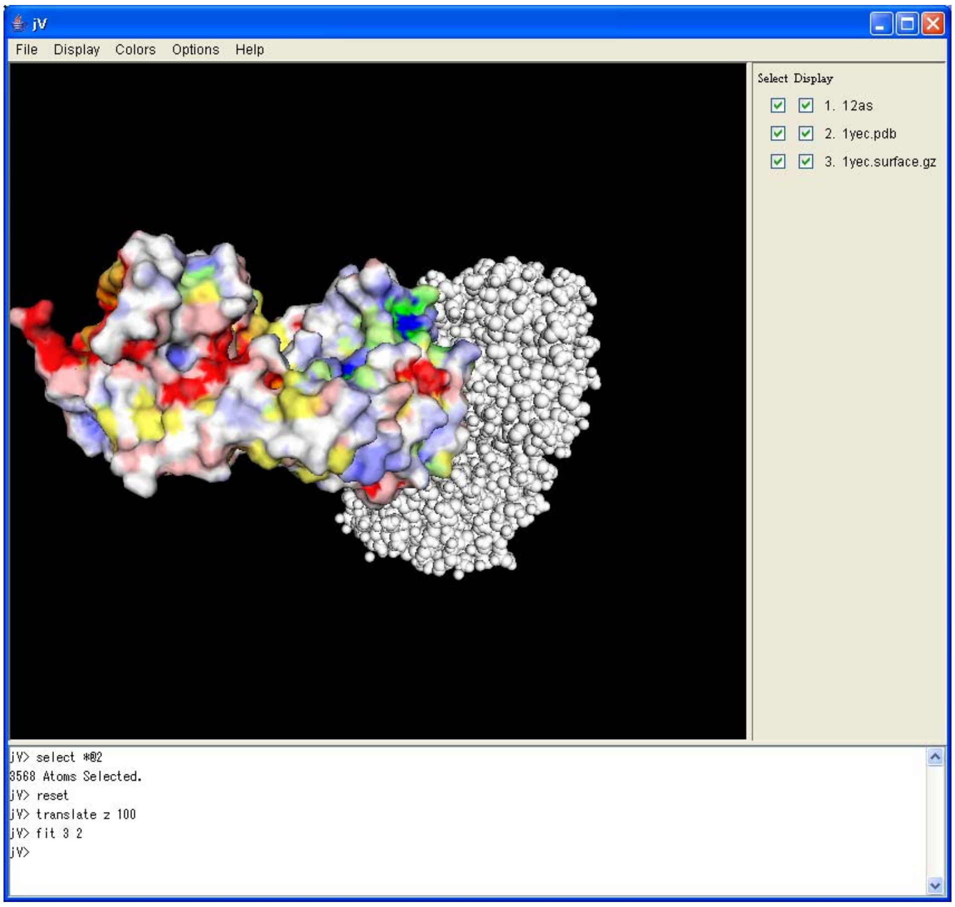
```
jV> select *@2
3568 Atoms Selected.
jV> reset
jV> translate z 100
jV> |
```

4.4. Displaying polygons

In addition to molecules, arbitrary shapes described by polygons can be read in and displayed together, where the polygon file is an XML file, whose document type is described in the section 6. Here, we use 'http://ef-site.hgc.jp/eF-site/jV/1yec.surface.gz' as an example. This polygon file represents a molecular surface of the molecule 2 (1yec) and the surface is colored according to the electrostatic potential and the hydrophobicity. It is assumed that the current image is the same as that in Figure 3. Then, open the open-remote dialog by choosing the [File] – [Open – Remote] – [Polygon] menu item, enter the above URL to the text field at the top of the dialog, and click the 'OK' button. As a result, the polygon is loaded, displayed and assigned file ID 3.

Note that the polygon is displayed at the different position from the molecule 2 since the molecule 2 is translated. However, because the polygon and the molecule 2 are described by the same coordinates in their files, their images can be exactly superposed by executing 'fit 3 2' from the command line, where the 'fit' command performs the copy of the transform matrix. Finally, the image in Figure 4 appears.

Figure 4. An example of polygon images.



```
jV> select *@2
3568 Atoms Selected.
jV> reset
jV> translate z 100
jV> fit 3 2
jV>
```

You can control the transparency rate of polygons. Try 'set transparency 0.5' from the command line, where 0.5 stands for 50% transparency. To restore the image, execute 'reset transparency' or 'set transparency 0'. In the same sense as each atom of a molecule, each vertex of a polygon is assigned either selected or not selected. All vertices are selected in the initial state. Execute a command 'displayvertex off' and all vertices become invisible. Next execute 'selectvertex within(10.0, 1&*H@2)', where this means to select vertices that exist within 10 angstroms from each atom that belongs to the first residue of the 'H' chain of the molecule 2. Subsequently execute 'displayvertex on'. As a result, only the selected vertices are appeared. The usage of the selectvertex command is summarized in the reference manual.

4.5. Working with xPSSS

Functional site information on a molecule stored at xPSSS (xml-based Protein Structure Search Service) is available in jV3. Before trying this, let us close the molecule 2 and the polygon 3 to make the situation simple. Select the [File] – [Close] – [2. 1yec.pdb] and [File] – [Close] – [3. 1yec.surface.gz] menu items sequentially. Then execute a 'reset' command and the transform of the molecule 1 and the position of the viewpoint are initialized.

Next, confirm that the 'select' checkbox of the file 1 is checked, and try a 'show xps3' command. The functional site information of the molecule '12as' is read from the xPSSS and 'type' and 'subtype' keywords for each functional site are listed. These keywords can be used to select a set of atoms. For example, if a keyword 'CATRES' exists (probably it does), the command 'select xps3:CATRES' selects the corresponding atoms.

In addition, gene ontology information can be obtained from the xPSSS. The gene ontology information for the molecule 12as is displayed by the command 'show godata'. Similar operations can be applied to other external databases on the Internet provided that the external database supports the interface that jV3 requires. See the section 7 for details.

4.6. Animation

The animation of molecules can be performed in jV3. The animation file takes the PDB format, where the MODEL line divides each frame. Let us open an animation file 'http://ef-site.hgc.jp/eF-site/jV/anim.pdb.gz' as an example from the open-remote dialog shown by selecting the [File] – [Open – Remote] – [Animation] menu item. When an animation file is loaded to the application, the first frame is automatically displayed in the wire frame model. You can start the animation by using the animation-control dialog that is shown by choosing the [Options] – [Animation] menu item. The dialog provides operations such as start and stop of the animation, selection of the animation mode, and adjustment of the speed. Some sample figures of the dialog are shown in the section 5.3.

4.7. Saving your work

While in your work, you can create a script file that contains a set of commands that reproduces the current work by using the [File] – [Save] – [Script] menu item. The script file can be loaded by the [File] – [Open – Local] – [Script] menu item. On the other hand, the [File] – [Save] – [PDB] menu item enables to save a molecule to a PDB format file with the coordinates of the atoms being transformed according to the current image. Here, only one molecule must be selected and only selected atoms in the molecule are saved. The other two menu items in the [File] – [Save] menu, [PNG] and [JPEG], save the current screen image to an image file in the PNG and JPEG format, respectively. The created image file has the same size as that of the 3D-display panel in the application.

4.8. Terminating the application

The application is terminated when the [File] – [Exit] menu item is chosen or the application window is closed by the way your operation system provides. You can also quit the application by executing 'exit' or 'quit' command.

5. GUI

5.1. Menu bar

The organization of the menu bar and the function of its components are shown below.

1) File menu

Component		Function
Open - Local	PDBML	A local PDBML file is opened.
	PDB	A local PDB file is opened.
	Polygon	A local polygon file is opened.
	Script	A local script file is opened.
	Animation	A local animation file is opened.
Open - Remote	PDBML	A remote PDBML file is opened.
	PDB	A remote PDB file is opened.
	Polygon	A remote polygon file is opened.
	Script	A remote script file is opened.
	Animation	A remote animation file is opened.
	PDB ID	A PDBML file at the PDBML FTP site is opened.
	eF-site ID	A set of molecular and polygon files stored in eF-site database are opened.
Information		Information about the selected files are shown.
Close		The specified file is closed.
Save	PDB	A PDB format file that contains the current atom coordinates is created.
	Script	A script by which the present condition is reproduced is created.
	PNG	The current image is saved as PNG.
	JPEG	The current image is saved as JPEG.
Exit		The application is terminated.

2) Display menu

Component	Function
Wireframe	The selected atoms are displayed in a wire frame model.
Backbone	The selected atoms are displayed in a backbone model.
Sticks	The selected atoms are displayed in a stick model.
Spacefill	The selected atoms are displayed in a space-fill model.
Ball&Stick	The selected atoms are displayed in a ball & stick model.
Ribbons	The selected atoms are displayed in a ribbon model.
Cartoon	The selected atoms are displayed in a cartoon model.

3) Colors menu

Component	Function
Monochrome	The selected atoms are colored in white.
CPK	The selected atoms are colored by CPK color scheme.
Shapely	The selected atoms are colored by the color scheme in which each amino acid and nucleic acid is assigned a unique color according to the amino acid and nucleic acid properties.
Group	The atoms of every chain are drawn as a smooth spectrum from red (N-terminal of the molecule) to blue (C-terminal).
Chain	The selected atoms are colored by the color scheme in which each chain is assigned a unique color.
Temperature	The selected atoms are drawn as a smooth spectrum from red (high value) to blue (low value) according to the value of the temperature factor.
Structure	The secondary structures are colored by the color scheme in which each secondary structure is assigned a unique color.
Charge	The selected atoms are drawn as a smooth spectrum from blue (positive) to red (negative) according to the charge.
Amino	The selected atoms are colored by the color scheme in which each amino acid is assigned a unique color.

4) Options menu

Component	Function
Hetero Atoms	Users can choose whether the heterogeneous atoms described by 'HETATM' in PDB files are selected or not.
Hydrogens	Users can choose whether the hydrogen atoms are selected or not.
Slab	Users can choose whether the z-clipping plane is located on the default position or is shifted.
Stereo	Users can choose whether the stereo display is enabled or not.
Load To Center	Users can choose whether a new image is added to the center of screen or not when a new file is opened.
Pick Off	The mouse-pick is disabled.
Pick Ident	The atom and residue names, their serial numbers, chain identifier and file ID of the object picked by mouse-click are represented.
Pick Coordinates	The atom and residue names, their serial numbers, chain identifier, file ID and coordinate of the object picked by mouse-click are represented.
Pick Distance	The distance between the first and second clicked atoms is calculated and shown.
Pick Center	The center of rotation and window are transferred to the clicked position.
Pick Select	Only clicked files are operated.
Animation	The animation control dialog is opened.

5) Help menu

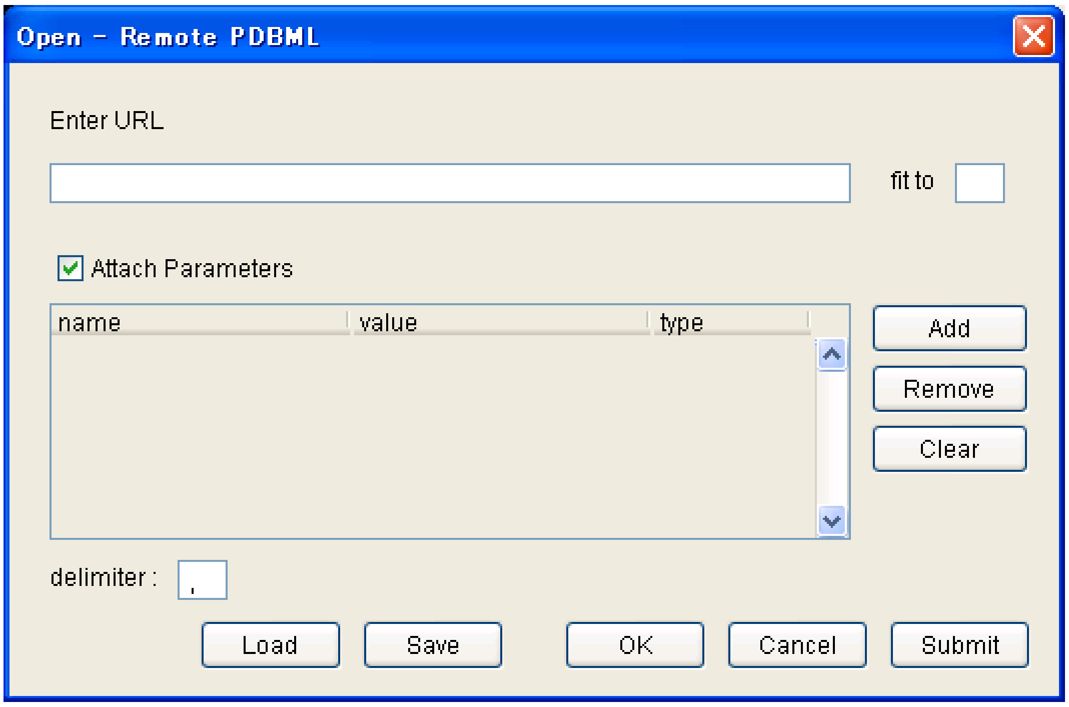
Component	Function
About jV	Information about this application is shown.

5.2. Open-Remote dialog

(1) URL with parameters

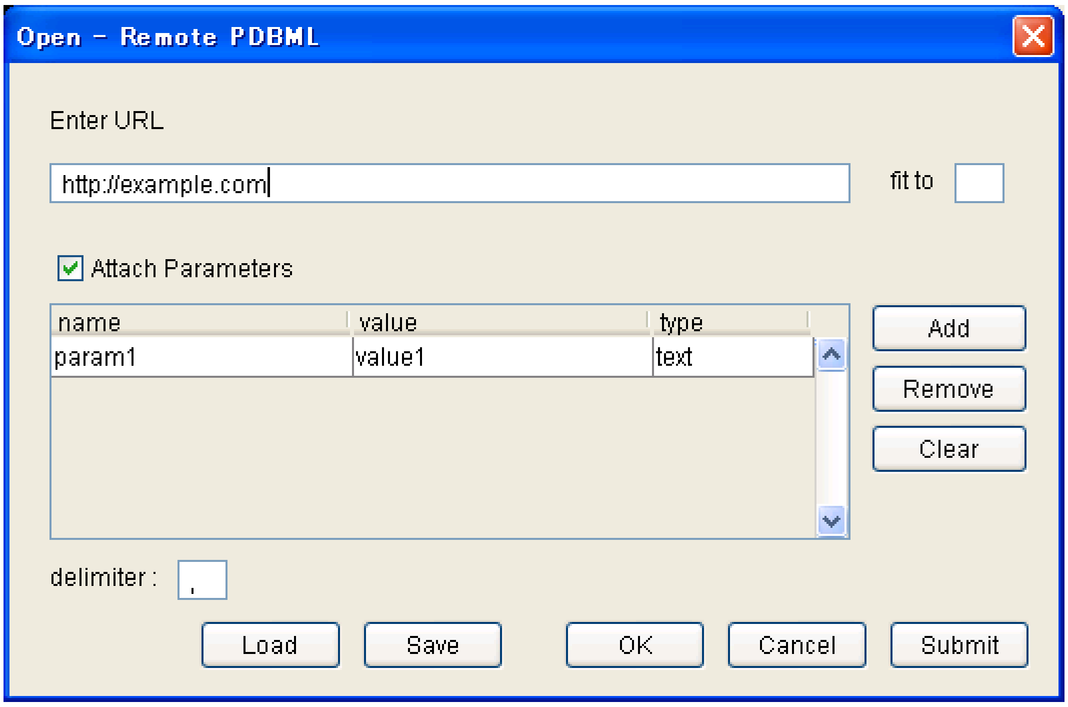
Selecting [PDBML], [PDB], [Polygon], [Script] or [Animation] menu item in the [File] – [Open – Remote] menu opens the open-remote dialog that can handle a URL with parameters. For example, Figure 5 shows a dialog opened by the [File] – [Open – Remote] – [PDBML] menu item.

Figure 5. The open-remote dialog.



When ‘Add’ button is clicked, a new parameter line is appended in the table at the center of the dialog. Figure 6 shows an example of a URL with one parameter. By clicking the ‘Add’ button the necessary number of times, you can attach multiple parameters. When ‘Remove’ button is clicked, a highlighted line in the parameter table is removed. When ‘Clear’ button is clicked, all parameters are removed.

Figure 6. Example of a URL with a parameter.



http://example.com

[✓] Attach Parameters

name	value	type
param1	value1	text

delimiter : ,

(2) 'type' column and mouse pick

The 'type' column in the parameter table is selected from the follows.

type	values	mouse pick
text	arbitrary value	does not work
atom ID	atom ID	work
atom	set of chain ID, residue ID and atom ID	work
residue	set of chain ID and residue ID	work
orig_coords	atom coordinates written in the molecule file	work
curr_coords	atom coordinates currently displayed	work

For the case of a type to which mouse pick works, selecting an atom in the screen by mouse pick sets the corresponding value to the 'value' column in a highlighted line in the parameter table. The value of the 'delimiter' field is used to delimit atom's x, y, and z coordinate.

(3) Saving and loading the configuration

By clicking the 'save' button, a file chooser dialog is appeared and the current configuration can be saved to a file. In the same way, a configuration can be loaded by clicking the 'load' button. The contents of a configuration file are in XML format as follows, whose document type seems obvious.

```
<?xml version="1.0"?>

<remote_file url="http://example.com">
  <params attach="true" coordinates_delimiter=",">
    <param name="param1" value="value1" type="text"/>
  </params>
</remote_file>
```

(4) Copy of the transform

When a file ID that currently exists is entered to the 'fit to' field, the transform of the newly opened file is set identical to that of the specified file.

(5) Execution

When the 'OK' button is clicked, a new file is loaded and the dialog is closed. When you use 'Submit' button, on the other hand, the dialog remains visible even after a new file is loaded.

5.3. Animation control dialog

The animation control dialog is visible only if [Options] - [Animation] menu is checked. When there is no animation file loaded, the animation control dialog is as Figure 7. If some animation files are opened, the animation control dialog is altered according to the number of the files. Figure 8 indicates the dialog in the presence of two files.

Figure 7. The animation control dialog in the absence of files.

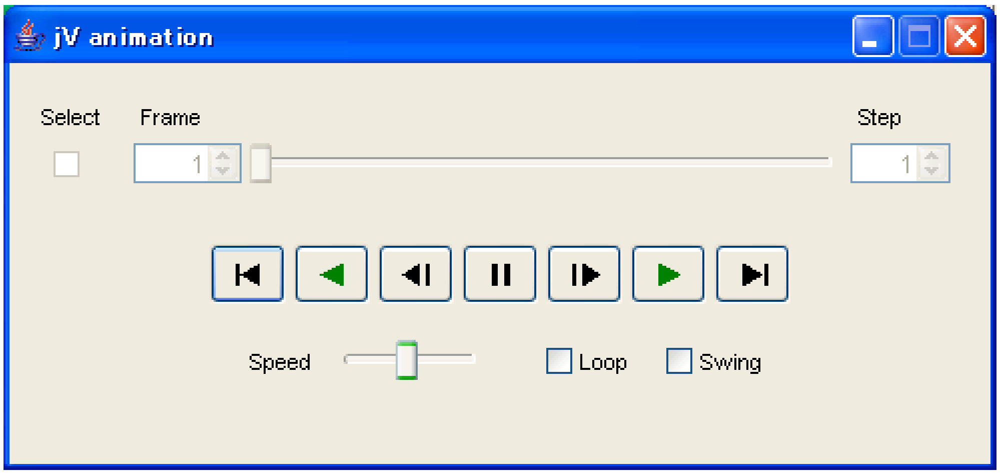
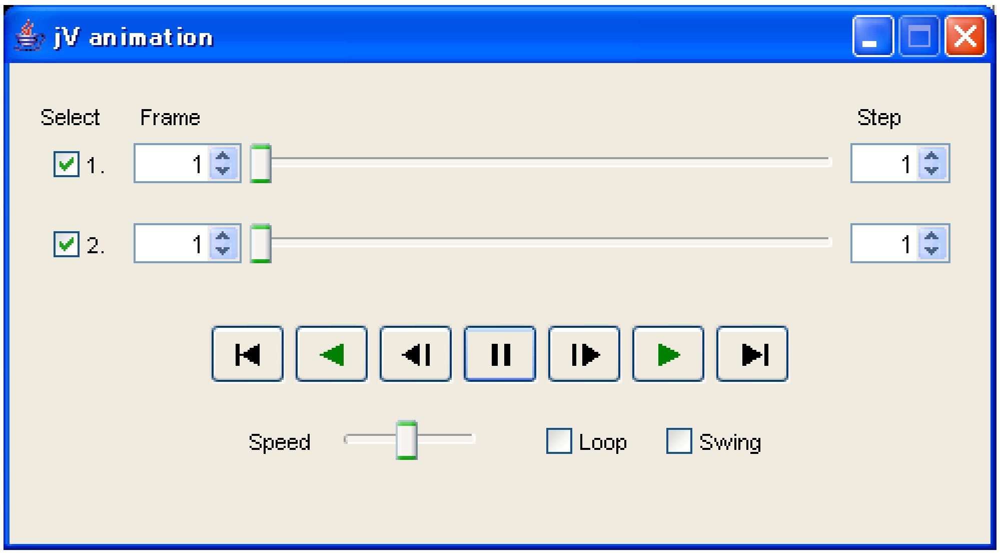


Figure 8. The animation control dialog in the presence of two files.



Users can select which files are controlled by their checkbox. The frame position and playback and stop of the animation can be controlled individually to each file. The playback-speed and the selection of 'loop' and 'swing' modes are common for all files.

6. Document type of the polygon file

The contents of a polygon file are organized as follows.

```
<?xml version="1.0"?>

<polygon>
  <vertices count="100" id_numbers=" use" >          ←impossible to omit
    <vertex id="1" image="x y z nx ny nz R G B"/>
    <vertex id="2" image="x y z nx ny nz R G B"/>
    ・・・
  </vertices>

  <point_array count="2" size=" 3" >                 ←possible to omit
    <point id="1" vertex="v1"/>
    <point id="2" vertex="v1"/>
  </point_array>

  <line_array count="2" width=" 2" >                 ←possible to omit
    <line id="1" vertex="v1 v2"/>
    <line id="2" vertex="v1 v2"/>
  </line_array>

  <triangle_array count="2" transparency=" 0.5" >    ←possible to omit
    <triangle id="1" vertex="v1 v2 v3"/>
    <triangle id="2" vertex="v1 v2 v3"/>
  </triangle_array>

  <quad_array count="2">                             ←possible to omit
    <quad id="1" vertex="v1 v2 v3 v4"/>
    <quad id="2" vertex="v1 v2 v3 v4"/>
  </quad_array>

  <polyline_array count="2" width=" 1" >             ←possible to omit
    <polyline id="1" vertex="v1 v2 v3 v4 v5 ・・・"/>
    <polyline id="2" vertex="v1 v2 v3 v4 v5 ・・・"/>
  </quad_array>

</polygon>
```

The ‘vertices’ element contains the sequence of vertices and the number of vertices is specified by the ‘count’ attribute. If the value of the ‘id_numbers’ attribute is ‘ignore’, the vertex id is automatically numbered starting from 1. If the ‘id_numbers’ attribute is set to be ‘use’, users can assign arbitrary numbers to each vertex id. In the ‘vertex’ element, x, y and z represent the x, y and z coordinates of the vertex, respectively, and nx, ny and nz represent the x, y and z components of the normal vector at the vertex, respectively. They are assigned real numbers. The color of each vertex is defined according to the RGB color scheme; R, G and B take the real numbers from 0 to 255. The ‘point_array’, ‘line_array’, ‘traiangle_array’, ‘quad_array’ and ‘polyline_array’ elements contain the sequence of polygons such as dots, lines, triangles, quadrangles and polylines, respectively. They can be omitted. The v1, v2, v3, v4 and v5 describe id of the vertices constructing the polygon. For these elements, the ‘transparency’ attribute that is omissible can be used to set the transparency of the image. To specify the size of dots and the width of lines, the ‘point_array’ element has the omissible ‘size’ attribute and ‘line_array’ and ‘polyline_array’ elements have the omissible ‘width’ attribute, respectively. The XML schema file that describes the type definitions and element declarations is published at http://ef-site.hgc.jp/eF-site/schema/polygon12.xsd. A simple example of a polygon is shown below. It represents a colored cube. The file is published at http://ef-site.hgc.jp/eF-site/jV/cube.xml. Before you open this polygon, you should close all files currently loaded to the application to get an appropriate viewpoint for the cube. To do this, you can use [File] – [Close] menu, or execute ‘zap’ in the command line.

```
<?xml version="1.0"?>

<polygon>
  <vertices count="24" id_numbers="use">
    <vertex id="1"  image=" 1  1  1  0  0  1 255   0   0"/>
    <vertex id="2"  image="-1  1  1  0  0  1 255   0   0"/>
    <vertex id="3"  image="-1 -1  1  0  0  1 255   0   0"/>
    <vertex id="4"  image=" 1 -1  1  0  0  1 255   0   0"/>
    <vertex id="5"  image="-1  1 -1  0  0 -1   0 255 255"/>
    <vertex id="6"  image=" 1  1 -1  0  0 -1   0 255 255"/>
    <vertex id="7"  image=" 1 -1 -1  0  0 -1   0 255 255"/>
    <vertex id="8"  image="-1 -1 -1  0  0 -1   0 255 255"/>
    <vertex id="9"  image=" 1  1  1  1  0  0   0 255   0"/>
    <vertex id="10" image=" 1 -1  1  1  0  0   0 255   0"/>
    <vertex id="11" image=" 1 -1 -1  1  0  0   0 255   0"/>
    <vertex id="12" image=" 1  1 -1  1  0  0   0 255   0"/>
    <vertex id="13" image="-1  1 -1 -1  0  0 255   0 255"/>
    <vertex id="14" image="-1 -1 -1 -1  0  0 255   0 255"/>
    <vertex id="15" image="-1 -1  1 -1  0  0 255   0 255"/>
    <vertex id="16" image="-1  1  1 -1  0  0 255   0 255"/>
    <vertex id="17" image=" 1  1  1  0  1  0   0   0 255"/>
    <vertex id="18" image=" 1  1 -1  0  1  0   0   0 255"/>
    <vertex id="19" image="-1  1 -1  0  1  0   0   0 255"/>
    <vertex id="20" image="-1  1  1  0  1  0   0   0 255"/>
    <vertex id="21" image=" 1 -1 -1  0 -1  0 255 255   0"/>
    <vertex id="22" image=" 1 -1  1  0 -1  0 255 255   0"/>
    <vertex id="23" image="-1 -1  1  0 -1  0 255 255   0"/>
    <vertex id="24" image="-1 -1 -1  0 -1  0 255 255   0"/>
  </vertices>
  <quad_array count="6">
    <quad id="1" vertex=" 1  2  3  4"/>
    <quad id="2" vertex=" 5  6  7  8"/>
    <quad id="3" vertex=" 9 10 11 12"/>
    <quad id="4" vertex="13 14 15 16"/>
    <quad id="5" vertex="17 18 19 20"/>
    <quad id="6" vertex="21 22 23 24"/>
  </quad_array>
</polygon>
```

7. Functional site information for molecules

A simple XML file is defined in jV3 in order to describe functional sites of a molecule. If an external database server that returns this type of XML file is prepared, you can let jV3 read the file through the network and obtain the functional site information. An example of the XML file is as follows.

```
<?xml version=" 1.0"  ?>

<site_list>
  <site id=" CATRES1"  db=" CATRES"  category=" catalytic"
        description=" a catalytic site defined by CATRES, Medline 98100076" >
    <region chain_id=" A"  beg_seq_id=" 100"  end_seq_id=" 100" />
    <region chain_id=" A"  beg_seq_id=" 46"  end_seq_id=" 46" />
    <region chain_id=" A"  beg_seq_id=" 116"  end_seq_id=" 116" />
  </site>
  <site id=" ASN"  db=" pdb_hetatom"  category=" binding"
        description=" ASPARAGINE binding site" >
    <region chain_id=" A"  beg_seq_id=" 48"  end_seq_id=" 48" />
    <region chain_id=" A"  beg_seq_id=" 72"  end_seq_id=" 74" />
  </site>
  <site id=" 0006529"  db=" godata"  category=" biological_process"
        description=" asparagine biosynthesis" >
    <region chain_id=" A" />
  </site>
  <site id=" 0016874"  db=" godata"  category=" molecular_function"
        description=" ligase activity" >
    <region chain_id=" A" />
  </site>
</site_list>
```

The XML schema file that defines the document type of the above XML file is published at http://ef-site.hgc.jp/eF-site/schema/sitelist10.xsd. When a text file 'properties.txt' exists in the same directory as the application jar file, jV3 reads the file as a configuration file for the application. The current configuration file, which is attached to the binary distribution, contains a set of URL necessary to connect to the xPSSS and eF-site system as follows.

```
# PDBML files are retrieved from the following site.
pdbml_noatom=ftp://www.pdbj.org/XML/alpha/all-noatom/
pdbml_extatom=ftp://www.pdbj.org/XML/alpha/all-extatom/
pdbml_plus=ftp://www.pdbj.org/XML/pdbmlplus/pdbml_xp/

# eF-site data are retrieved from the following site.
efsite=http://ef-site.hgc.jp/eF-site/
```

If a database server that returns the functional site file is prepared, its URL should be appended to the configuration file. Because one functional site file is assumed to exist for one PDB ID, the URL written in the configuration file depends on the PDB ID. Therefore, the URL is represented with the use of substitute characters {0}, {1}, {2} and {3} that are substituted by each column of the PDB ID sequentially. An example is as follows.

```
# works with my database.
mydb=http://myhost.jp/mydb/jv3/{0}{1}{2}{3}_jv.xml
```

Here, the property name, 'mydb' in the above example, is arbitrary. Note that a PDB ID {0}{1}{2}{3} is substituted as lowercases and available protocols are http:, ftp: and file:. For simplicity, it is assumed that only one molecule file that corresponds to the above example XML file opens in the application. Then functional site information can be displayed by the 'show' command as follows.

```
jV3> show site mydb
File 1:
  db           category
  -------------------------------
  CATRES       catalytic
  pdb_hetatom  binding
  godata       biological_process
  godata       molcular_function
jV3> show site mydb:CATRES
File 1:
  id=CATRES1  category=catalytic  region=A:100-100,A:46-46,A:116-116
      description=a catalytic site defined by CATRES, Medline 98100076
jV3> show site mydb:pdb_hetatom:binding
File 1:
  id=ASN  category=binding  region=A:48-48,A:72-74
      description=ASPARAGINE binding site
jV3> show site mydb:godata
File 1:
  id=0006529  category=biological_process  region=A
      description=asparagine biosynthesis
  id=0016874  category=molecular_function  region=A
      description=ligase activity
jV3> show site mydb:godata:molecular_function
File 1:
  id=0016874  category=molecular_function  region=A
      description=ligase activity
jV3>
```

With a similar syntax, the ‘select’ command can be used to select the relevant set of atoms. To demonstrate the operations described in this section, let us add a following line to the configuration file ‘properties.txt’.

```
ex=http://ef-site.hgc.jp/jv3site/servlet/Site?pdb={0}{1}{2}{3}
```

The above URL provides functional site information whose contents are equivalent to those of xPSSS for all PDB. To reload the configuration file, quit the application (if it runs) and restart it. Then open the molecule ‘1yec’ as an example. To do this, you can use [File] – [Open – Remote] – [PDB ID] menu item as in the section 4.2, or equivalently execute ‘load ftp 1yec’ in the command line. Subsequently, execute ‘show site ex’ and the following keywords will be listed (the contents may change in the future).

```
jV3> show site ex
File 1:
  db          category
  ---------------------
  prosite     prosite
  pdb_hetatom binding
```

Detail site information can be shown as follows.

```
jV3> show site ex:prosite
File 1:
  id=PS00290  db=prosite  category=prosite  region=L:192-198
    description=Immunoglobulins and major histocompatibility complex proteins signature.
[FY]-{L}-C-x-[VA]-{LC}-H
```

Atoms in the region ‘L:192-198’ can be selected with a similar syntax.

```
jV3> select ex:prosite
56 Atoms Selected.
```

In the same way, the following atom selections are possible.

```
jV3> select ex:pdb_hetatom
298 Atoms Selected.
jV3> select ex:
354 Atoms Selected.
```

Here the last command selects both regions specified by ‘prosite’ and ‘pdb_hetatom’ keywords, respectively.