JIV : A 3D Image Data Visualization and Comparison Tool

Chris Cocosco <crisco@bic.mni.mcgill.ca>

1 Inputs

1.1 Environment

JIV can be run either as a Java *applet*, either as a (standalone) Java application.

Running it as an applet requires an HTML file to invoke it; however, this can be a very simple file, containing only the `applet` tag. This applet expects a run-time parameter (in the form of an applet parameter): the URL of a config file, which contains information about which 3D data volumes to load, the layout of the user interface, initial settings of various controls, and so on.

The applet can be launched by some HTML code like this:

```
<applet height=50 width=400 archive="jiv.jar" code="jiv/Main.class">
        <param name="cfg" value="config_file">
</applet>
```

where `jiv.jar` should be the URL of the JAR file containing the JIV Java bytecode, and `config_file` should be the URL of the appropriate JIV config file (`config` can be used instead of `cfg` for the parameter name). The config file can also be supplied "inline" within the HTML file using the `inline_cfg` (or `inline_config`) applet parameter; for example:

```
<applet height=50 width=400 archive="jiv.jar" code="jiv/Main.class">
        <param name="inline_cfg" value=" ;
# config file's content here: ;
data1 : data1.raw_byte.gz ;
jiv.panel.0 : data1 ;
# ... ;
">
</applet>
```

Note that, unlike the separate config file, the inline config needs to have each line terminated with the ';' character. Both a separate config file and an inline config can be supplied at the same time: the inline config is read last, so its content takes precendence for keys defined in both places.

The HTML file is not needed when JIV is not run as an applet but as a regular Java application. In this case, the config file's URL can be supplied as a command line argument, or as the value of the `cfg` or `config` Java system properties ("environment variables"). The top-level class that needs to be run is the same (`jiv/Main.class`).

1.2 Config File

The config file is expected to be in the following format:

- Lines are separated by ';' in the inline config, and by the normal 'newline' in a config file.
- Lines that begin with `#` or `!` are comments and are ignored.
- Blank lines are ignored.
- All other lines should specify a key/value pair and be of any of the following three equivalent forms:

```
key = value
key : value
key value
```

Leading/trailing whitespace and control characters in `value` are trimmed off.

- The following escape characters are also recognized and treated as follows:
  - `\newline` : an escaped newline character is ignored, along with the spaces or tabs that follow it.
  - `\n` : expands to a newline character.
  - `\r` : expands to a return character.
  - `\t` : expands to a tab character.
  - `\uxxxx` : expands to the Java Unicode character code specified by the hexadecimal digits.

Keys which start with "`jiv.`" are JIV configuration options. Keys which do not start with "`jiv.`" give the location of the image data files (image volumes). JIV configuration options refer to data files by means of *data volume aliases*, i.e. a kind of a short name. The alias is displayed as a title at the top of each panel, hence it's a good idea to choose something descriptive and short (such that it will fit in the, possibly narrow, panel).

For a given alias "myalias", the value associated with key `myalias` is the volume image file's URL (URL-s for the individual slices are derived from the volume URL, as described in section 1.3). The value associated with key `myalias.header` is the URL of the header file associated with the image file(s). All relative URL-s are interpreted relative to the base URL of the config file, or relative to the base URL of the HTML file launching the applet if a config file is not defined (by means of the `cfg` or `config` applet parameters). If all volumes have the same header, it is acceptable to specify it only for one of the volumes; otherwise, a header should be specified for all volumes (see section 1.3 for the header file format).

The following JIV configuration options are supported:

- `jiv.sync = [true|false]`
  Sets the initial state of the *Sync all cursors* control. The default is false.

- `jiv.world_coords = [true|false]`
  If false, only voxel coordinates will be available in the user interface (however, doing this is *not* recommended practice!). The default is true.

- `jiv.byte_values = [true|false]`
  If true, all the voxel values, including the colormap range values, are presented in the user interface as byte values (0–255). If false, they are presented as fractional values (0.0–1.0). The default is false.

- `jiv.panel.N = alias`
  Specifies an *individual volume panel*, i.e. an interface panel displaying a single data volume (specified by `alias`, which should be a data volume alias declared somewhere else in the same config file). `N` should be a non-negative integer and represents this panel's number.

- `jiv.panel.N.combine = alias1 alias2`
  Specifies an *combined volume panel*, i.e. an interface panel displaying a combined view of two data volumes (specified by `alias1` and `alias2`). `N` should be a non-negative integer and represents this panel's number. The two aliases should be separated by one or more blanks (␣) or tabs (`\t`). Also, these two aliases *should* be displayed in their individual panels as well. If an alias is displayed in more than one individual panel, then the lowest numbered such panel is used as the "source" for that volume alias.

- `jiv.panel.N.coding = [gray|grey|hotmetal|spectral|\`
  `red|green|blue|mni_labels]`
  Specifies the initial color coding for panel `N`, which has to be an individual volume panel.

- `jiv.panel.N.range = L U`
  Specifies the initial lower and upper limits of the color coding range for panel `N`, which has to be an individual volume panel. `L` and `U` should be fractional (float) numbers in the range 0.0–1.0. The two should be separated by one or more blanks (␣) or tabs (`\t`).

- `jiv.download = [upfront|on_demand|hybrid]`
  Specifies the data (down-)loading method.

  In mode `upfront`, all of the data is loaded and stored in memory before the user can view and interact with any of it. This guarantees the best interactive response of the viewer, however the user has to wait for all the data to load before the JIV interface becomes available. This mode is recommended when the data is available locally, but is impractical for accessing remote data over a slow network.

  In mode `on_demand` image data is loaded one slice at a time, and only if and when the user is viewing that particular slice number. This mode is recommended for remote access over very slow networks. It minimizes the data downloads and the amount of memory required by JIV, but the interactive performance is completely dependent on the network speed.

  Mode `hybrid` combines the other two: the complete image volume is downloaded in the background, and the slices at the cursor are downloaded with priority, as soon as they are needed. This mode is the recommended for typical remote data access situations; it provides the fast startup of `on_demand` and, after the background downloading completes, the optimal interactive performance of `upfront`.

Panels are displayed left to right, sorted by their increasing number. Note that these numbers do not have to be consecutive — "gaps" in the numbering sequence are silently skipped. Note that the same alias (i.e. the same data volume) can be displayed in several individual and combined volume panels.

The config file is parsed in several passes, so the order of the key/value pairs is irrelevant. However, if several conflicting key/value definitions are given (which is bad practice, by the way), not the last definition given but a random one of them will be considered!

1.3 Data Files

JIV reads 3D image data from an image file which should contain the image intensity (gray level) data represented as unsigned bytes (8-bit). The image data are interpreted using an associated header file, which specifies the volume sampling, world coordinates (real world mm), and real image values. All the data files can be optionally compressed using `gnuzip`, in which case their names (URL-s) need to have the `.gz` suffix.

The *header file* is a text file with the same syntax as the config file (see section 1.2). The following statements are supported (where the 3 values of the right-hand side can be separated by whitespace or "`,`") :

- `size : x_size y_size z_size`
  the sizes (voxel counts) of the 3D image along the x, y, and z axes respectively.

- `start : x_start y_start z_start`
  the distance (in real world mm) from the origin to the first voxel along the x, y, and z axes respectively.

- `step : x_step y_step z_step`
  the signed distance (in real world mm) between the centers of consecutive voxels along the x, y, and z axes respectively; a negative value indicates that the file scans the volume in negative direction along that axis.

- `order : (permutation of {x,y,z})`
  the order of dimensions in the file, i.e. the order in which the volume file scans the (3D) data volume — e.g. "`order : z y x`" means that the x coordinate changes fastest (volumes using this particular dimension ordering are also known as "transverse").

- `imagerange : range_min range_max`
  the linear mapping from the byte voxel values to the real image voxel values: 0 maps to range_min, 255 maps to range_max.

The x, y, z axes and their positive direction are assumed to be (respectively): left to right, posterior to anterior, and inferior to superior of a 3D medical image. Note that the sizes, starts, and steps are always expected in x, y, z order, regardless of the file dimension order. If any of the header information is not specified, the following defaults are used (they correspond to the "ICBM" sampling and to the "Talairach" stereotaxic coordinates systems used at the Montreal Neurological Institute, McGill University):

```
size   :  181 217 181
start  :  -90 -126 -72
step   :  1 1 1
order  :  z y x
imagerange  :  0.0 1.0
```

Performance tip: Due to internal optimizations, the initial download of the image data will be faster if all the steps are positive and equal (for all dimensions and all image volumes) and the dimension ordering is `'z y x'`. However this performance difference will be hardly noticeable when using a modern Java runtime environment (JVM)...

There are two kinds of image files:

volume file: contains the complete (3D) image volume data; it is required by the `upfront` and the `hybrid` download modes.

slice file(s): contain a 2D slice of the 3D image volume; three sets of all the slices orthogonal to each of the three coordinate axes are required by the `on_demand` and the `hybrid` download modes.

The volume file URL is specified in the config file (as shown above in section 1.2); the slice files are expected at URL-s of the form: `base/orientation/slice_number.extension`, where `orientation` is one of "01", "02", or "12" — the name indicates which are the in-slice dimensions (in file dimension order) for that orientation. For example, for a "transverse" volume (order: z y x) 01, 02, 12 correspond to "sagittal" (z-y), "coronal" (z-x), and "transverse" (y-x) slice orientations.

`base` and `extension` are obtained by breaking the volume file URL at the last "`.`" (other than the suffixes `.gz` or `.bz2`); for example, all of the following:

```
/some/dir/somename.raw_byte.gz
/some/dir/somename.raw_byte
/some/dir/somename.gz
/some/dir/somename
```

result in `base` = `/some/dir/somename` .

For converting MNI-MINC data to the JIV input format, two utilities (Perl scripts) are distributed along with JIV: `minc2jiv.pl`, and `jiv.pl`.

Note: the `upfront` download mode requires the volume file(s), the `on_demand` mode requires the slice files, and the `hybrid` download mode requires both the volume and the slice files.

2 Graphical User Interface

The main JIV window is composed of one or more "panels" — columns of interface elements separated by vertical lines. Each panel has a title indicating its content and type. There are two kinds of panels:

combined data volume panel Has a title of the form "`vol_name1 <-> vol_name2`", and displays a combined view of two 3D data volumes.

individual data volume panel Has a title of the form "`vol_name`", and displays a single 3D data volume.

The two data volumes that are displayed together in a combined volume panel are always displayed in their individual panels as well.

JIV allows its main window to be resized at will (using the techniques specific to your platform and windowing environment). However, if the window is too small (especially if it's too narrow for the number of displayed panels) some interface components may overlap in a confusing way, or even become completely obscured.

2.1 Common panel features

Each panel is composed of four visible main elements, aligned in a vertical column; from top to bottom, they are:

1. "Transverse" 2D slice viewport ($Z = const$).
2. "Sagittal" 2D slice viewport ($X = const$).
3. "Coronal" 2D slice viewport ($Y = const$).
4. Panel controls area.

The x, y, z axes and their positive direction are (respectively): left to right, posterior to anterior, and inferior to superior of the head. Besides the (always visible) controls area at the bottom of the panel, additional controls are available through a panel-specific pop-up menu.

2.1.1 Slice viewport

Inside a 2D slice viewport, the following mouse actions are available:

primary-mouse-button *Moves the cursor* to a new position in slice's plane. Can also drag the cursor around.

secondary-mouse-button Combined with a vertical mouse movement, moves the cursor along the axis orthogonal to slice's plane (i.e. *changes the displayed slice*). The cursor displacement is proportional to the relative vertical mouse drag movement. This operation can also be performed using keyboard commands (see below).

Shift or Ctrl + primary-mouse-button Moves the field of view (i.e. does a *pan*) by following the mouse drag movement.

Shift or Ctrl + secondary-mouse-button Combined with a vertical mouse movement, *changes the zoom/scaling factor*. The change in the scaled image dimensions is proportional to the relative vertical mouse drag movement.

double-click primary-mouse-button *Marks* the current cursor position as the origin (first point) for the in-slice distance measurement. Also, it *enables* the distance measurement mode, if necessary.

double-click secondary-mouse-button *Disables* the distance measurement mode.

The meaning of "primary" and "secondary" mouse buttons is provided by the Java implementation used to run JIV, and is platform-dependent — e.g. on a Unix/X-Windows platform, with the common right-handed mouse configuration, primary is the left button, and secondary is either of the middle or right mouse buttons. On all platforms, the secondary button can be emulated by pressing the `Meta` or `Alt` key together with the primary button.

The following keys can be used for changing the displayed slice – that is, moving the cursor along the coordinate axis orthogonal to slice's plane by an amount of 1mm (one voxel):

- `Right`(→), `Up`(↑), or `+` : positive increment (next slice).
- `Left`(←), `Down`(↓), or `-` : negative increment (previous slice).

The in-slice distance measurement feature interactively displays the world-coordinates distance between the origin ("marked point") and the current cursor position. The value is always given in real world units (mm). The measurement origin is preserved across slice changes, same as the current in-slice cursor position is. The following keys can also be used for controlling this measurement feature:

- `d` : same as *double-click primary-mouse-button* (see above).
- `c` : same as *double-click secondary-mouse-button* (see above).

The image scaling (i.e. zoom-up/down) is done using *nearest-neighbour* interpolation: pixels are replicated (for enlargements) or skipped (for reductions) as needed.

When the viewport dimensions change (as a consequence of the main window being resized), JIV will adjust the field of view such that it's not less than the previous field of view, while at the same time using as much of possible of the new viewport area.

2.1.2 Controls area

The controls area of each panel has, at the top, a group of three text fields that display the current X, Y, Z coordinates of the cursor in that panel. These fields also allow editing (text input) – the usual text editing commands of your platform should work. If a value out of range is given, the field will revert to its previous (valid) value. The coordinates displayed (and read in) are voxel or world coordinates, depending on the current panel setting.

Any horizontal sliders present in this area are implemented using platform-specific scrollbars, thus their behaviour in response to mouse (and maybe keyboard) actions should be similar to the other scrollbars on your computer platform (and/or windowing system).

2.1.3 Pop-up menu

The mouse and/or keyboard command that brings up the pop-up menu is the usual one for triggering context-sensitive (pop-up) menus on your particular computer platform — e.g. on Unix and on Microsoft Windows, it's usually the right (second or third) mouse button.

The following menu actions are available in all panels:

- `Coordinates type` [choice] : Changes the type of coordinates that are displayed, and read in, in the controls area at the bottom.
- `Sync all cursors` [toggle] : If `on`, the cursor positions in all panels will be kept the same. When changing this control from `off` to `on`, all cursors will be set to the cursor position of the first panel (from the left).
- `Help` [menu]: Provides access to the JIV version and copyright info (the `About` command), and to this help document (the `Help` command). If JIV is running as an applet in a web browser, the online help file should be opened in a new browser window[1]. The online help is not available if JIV is running as a standalone application.
- `Quit` : Closes the JIV window and exists the application. However, when running JIV using a web browser or an appletviewer, this action is probably not enough; in order to completely dispose of this running copy of the JIV applet you may have to move out of the HTML document that launched it, or maybe even close that browser frame/window. To make things worse, some web browsers don't release the (possibly large amount of) memory formerly used by the applet unless you completely shutdown the browser!

2.2 Individual volume panel features

The 8-bit intensity values in the data volume are displayed in the viewports using a user-controlled colormap. This is composed of a certain color-coding scheme (in between adjustable lower and upper limits), an "under" color (below the lower color-coding limit) and an "over" color (above the upper color-coding limit).

2.2.1 Controls area

To the left of the three coordinate fields, there's a read-only text field displaying the voxel (intensity) value at the cursor position.

Below the coordinate text fields, there are controls for the lower color-coding limit (left text display/entry field and lower slider) and for the upper color-coding limit (right text display/entry field and upper slider). The lower limit can never be higher than the upper limit.

The values displayed (or read in) by the voxel intensity and lower/upper color-coding limit text fields can be a fractional ones (in the range 0.0–1.0) or byte values (in the range 0–255), depending on how this instance of JIV was configured.

The current colormap is displayed as a color bar at the very bottom of the panel.

[1] provided this file is available as `doc/help/index.html` relative to the applet's code-base or document-base.

2.2.2 Pop-up menu

The following menu actions are available in the individual volume panels only:

- `Color coding` [choice] : Changes the color-coding scheme.
- `"Under" color` [choice] : Changes the "under" color (`(default)` means the same color as the one at color-coding's lower limit).
- `"Over" color` [choice] : Changes the "over" color (`(default)` means the same color as the one at color-coding's upper limit).
- `Tie colormap sliders` [toggle] : If `on`, the two color-coding limits behave like being connected together by a solid rod — adjusting one value implies changing the other one such that their difference ("distance") remains the same.

2.3 Combined volume panel features

The coloring of each of the two data volumes is the one from the individual volume panel representing that data volume. If a data volume (i.e. a volume alias) is displayed by more individual volume panels, then the left-most such panel is used as the coloring source for that data volume.

Currently, the only method available for combining ("compositing") the two data volumes is *blended*: the color of each pixel of the combined image is given by

$$color\_in\_volume\_1 \times (1 - \beta) + color\_in\_volume\_2 \times \beta$$

where β is the blend factor: a fractional value in the range 0.0–1.0 (this compositing is done in RGB color space).

2.3.1 Controls area

Below the coordinate text fields, there is a blend factor (β) slider, surrounded by two text display/entry fields: the left one for $1 - \beta$, and the right one for β. In other words, these two text fields contain the weighting factors for the two combined data volumes.