Handling Multi-Dimensional Data

Advanced Technology Development Group
Gray Cancer Institute

Data, data, everybody wants more data. In our cell biology work investigating the mechanisms of cancer we find that not only do we want to see the cell as a detailed, high resolution, map in two dimensions (x, y) but that it would be great to section the cell using a confocal or two-photon microscope and explore detail in the third dimension (z). Of course we are also interested in how cells behave in time and time lapse video can be very informative, this of course adds a fourth dimension (T). Time lapse and real time video can capture slow events very well but there are fast events, such as fluorescent emission, which are also of interest, add a dimension (t) which a much higher time resolution (perhaps a few hundred picoseconds) to the system and we have now five dimensions in which to measure. That's got to be enough! Oh, our detailed (x, y) map only cover a very small portion of the sample so we now want to mosaic images together, taking a number of images in both the x and y directions, so we have two more dimensions (X, Y). That's now seven. I am sure there was an eighth … probably mosaicing in z (Z) and we haven't yet considered spectral data at different wavelengths.

A different type of data, that is in some way multi-dimensional, is parametric or vector data. In this form of data, we may have an image of (x, y) intensity data; each pixel has an intensity value. In addition, we may also have a lifetime and/or a phase value associated with each pixel. Such an image where there are a number of values or parameters for each pixel is called a multi-parametric image.

We would like to handle all the dimensions of these types of images in computer memory at the same time but also we would like to store it all in a sensible way on disk, all in one file if possible, together with information about the scales and extent of all the dimensions. We would also like to store information about how the data was acquired, and perhaps what processing has been done to the data. The following text describes how we achieved both these requirements with Dynamically Allocated Arrays (DAA) and the Image Cytometry Standard (ICS) file format. Also described is the start of a project to display and visualize these types of data.

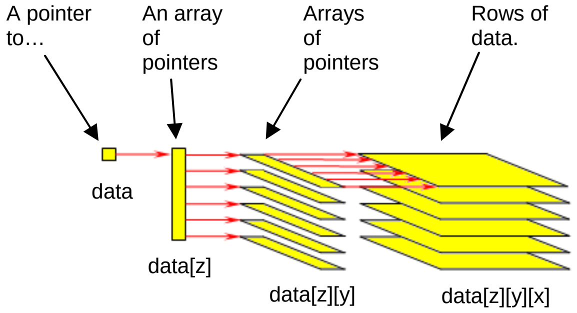
Multi-Dimensional Data in Computer Memory – DAA Structures

We made use of freely available 'C' language code for the creation and management of very useful Dynamically Allocated Arrays (DAA). This code library was available from:
http://www.numerical-recipes.com/contrib/,
see also:
http://www.embedded.com/2000/0012/0012feat4.htm.

Being based on ANSI C it can be easily incorporated into our favourite development environment, Labwindows/CVI. The code will allocate a single block of memory for the data but will also create relevant pointers to make accessing the data easy. Very easy in fact, if you want to access a value at position (x1, y1, z1) in a 3D data set just write data[z1][y1][x1]. If you are happy that this works, skip the next section. Read on if you are interested in how this is achieved.

When the DAA is set up, enough memory is allocated for the data itself and also for arrays of pointers. Let's say we have 3D data, (x, y, z), and say this is arranged so that the x values are contiguous in memory. Lets also say that the data set is 10 by 10 by 10 (i.e. there are ten x values, ten y values, and ten z values. So, in memory we have blocks (rows) of 10 values, starting with the value at (1, 1, 1) and going to (10, 1, 1), then comes the block (1, 2, 1) to (10, 2, 1), then (1, 3, 1) to (10, 3, 1) and so on. If we make an array of pointers, where each pointer points to one of these blocks, any value in a block can be accessed by pointer[1] for the first value, pointer[2] for the second etc. (pointer[x]). This array of pointers is 2 dimensional and IT will have blocks of pointers. Let's setup another array of pointers that points to these blocks of pointers. We can then access any data value by dereferencing 2 pointers: e.g. pointer[1][1] (pointer[x][y]). This second layer of pointers will be a single block in memory can then be accessed via another pointer (just one this time): let's call it 'data'.

So 'data' is the pointer to the start of the whole DAA (it is a pointer to a pointer to a pointer to a value). data[z] will select a particular plane of data (it is a pointer to a pointer to a value). data[z][y] will select a particular row of data with that plane(it is a pointer to a value). data[z][y][x] will give you the actual value (to get the pointer to any value use &(data[z][y][x]) as data[z][y] is always the start of a row).



Three dimensional data, represented on the right hand, side can be easily accessed by dereferencing 3 levels of pointers with code like: data[z][y][x].

The cost of using a structure like this is extra memory required to store all these pointers. Let's just see how you could access the data without these arrays of pointers. You would have a single pointer to the start of the block of data and you would have to calculate the position of the value you require within this block. This would look something like: value = data[10*10*z + 10*y + x]. As you can imagine, the cost of this method of access is speed. Now, imagine trying to access an 8 dimensional dataset in this way!

The usual sequence of calls is represented in the following pseudo code to set up a 3D array of unsigned short (16 bit) data:

```
//setup some variables
int d[3], asize, error_code, st[3] = {0, 0, 0};
unsigned short ***data, *free_ptr;
d[0] = image_planes;
d[1] = image_rows;
d[2] = image_cols;
asize = das(sizeof(short), 3, d, &err_code);           // calculate the size of the memory block required
free_ptr = malloc(asize);                              // allocate memory
data = (unsigned short ***) daa(sizeof(short), 3, d, st, &err_code, free_ptr, NULL); // setup the array
```

Data can be accessed with:

```
unsigned short value;
value = data [z][y][x];
```

When the DAA is finished with the memory it occupies should be freed with:

```
free (free_ptr);
```

The purpose of the ‘free pointer’ (free_ptr) then becomes clear.

Image Cytometry Standard (ICS) File Format

After much searching for a file format that meets our requirements we settled on the Image Cytometry Standard (ICS) File Format which is highly suited to our type of work. It allows multiple dimensions to be stored all in one file together with information about the scale and extent of all the dimensions. It also allows us to store information about how the data was acquired, and what processing has been done to the data. The ‘C’ language source code for reading and writing ICS files (libics) is freely available at http://www.ph.tn.tudelft.nl/~cris/libics/. There is also a plug-in for ImageJ, allowing compatibility with these files: http://valelab.ucsf.edu/~nico/IJplugins/Ics_Opener.html. Plus, your favourite free image viewer, “IrfanView”, also supports this file format if you install the plug-ins (http://www.irfanview.com).

The ICS file format is very easy to use. The data is stored in file as it is in memory. You have to supply the information about how many dimensions, how many x, y values etc. there are, and so on. When the file is loaded again the data is put back in memory and the information you provided is again available to you. In addition to the minimum information required to successfully save and reload your data, the ICS format allows for the storage of lines of ‘history’. This can be anything but we have aimed to include a few standard items in these history lines. The following table outline the GCI standard history lines that we include with every file:

History Key	History Text
“author”	Usually the username of the person logged in
“created on”	Date time OR time date (UK format please)
“type”	“Time Resolved” or “GCI mptr” “Linear Unmixing Result”
“labels”	“x y t”, “t x y”, “x y c” (colour image), “x y w” (spectral image) etc. single letters only separated by spaces. CGI_reformatDaaData() may use these labels. Spatial labels – x y z Time – t To store the coordinates of images in mosaics use – X Y Z Time lapse – T Spectral – w Colour channel – c
“units”	“m m s” for metres in x, metres in y and time in z, etc.
“extents”	“1.0e-6 2.0e-6 3” for a 1 um by 2 um by 3 second stack, etc.
“offsets”	Usually “0 0 0”, but can be used to specify an absolute coordinate.
“dimensions”	“256 256 64” etc. The number of pixels in each dimension
Any other app specific keys	Appropriate information

The history ‘keys’ are NOT case sensitive. The text / info may be (e.g. the labels info is case sensitive).

RED = required
YELLOW = should be included
GREEN = optional

Optional, App specific keys may have information that is matched to the labels (e.g. as the units, extents and offsets etc. are). In this case the key should have a ‘*’ as the last character. This is so that the GCI_reformatDaaDataAndHistory() function of the GCI_ImageIO library knows to reformat that history line when reformatting the data (It already knows about the standard keys: units, extents, etc.).

We have implemented a static library for linking with a C compiler, “GCI_ImageIO”, which among other things, allows direct import of ICS files into DAA structures and *vice versa*.

Viewing Multi-dimensional Data with Smart Viewer

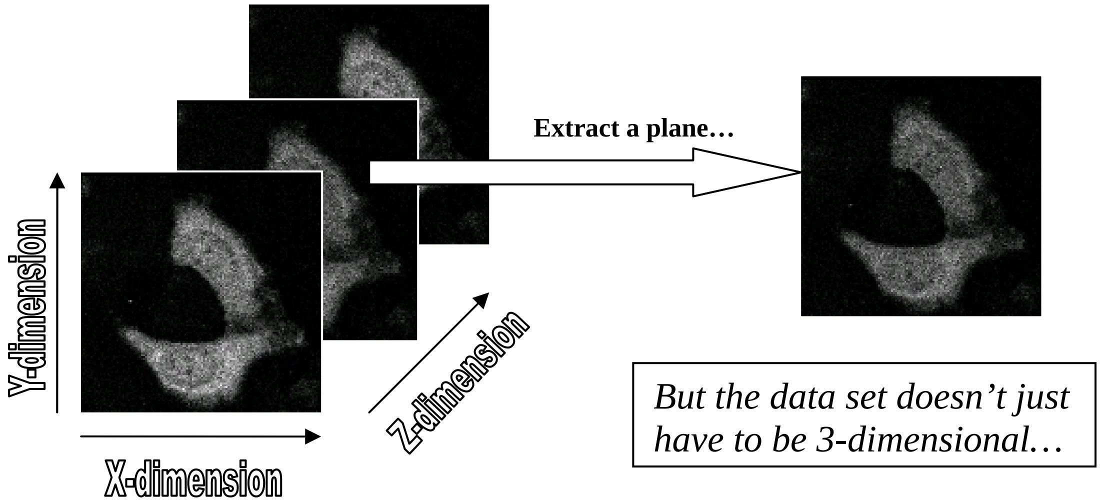
SmartViewer is a tool that we designed for the visualization of multi-dimensional data. In short, it allows the selection and display of any orthogonal 2D plane from within a particular data set, and such data can be exported as standard image formats or movie files. It can also be used to reformat, perform arithmetic, extract and delete regions from data sets.

Possible uses and features of particular interest of SmartViewer include:

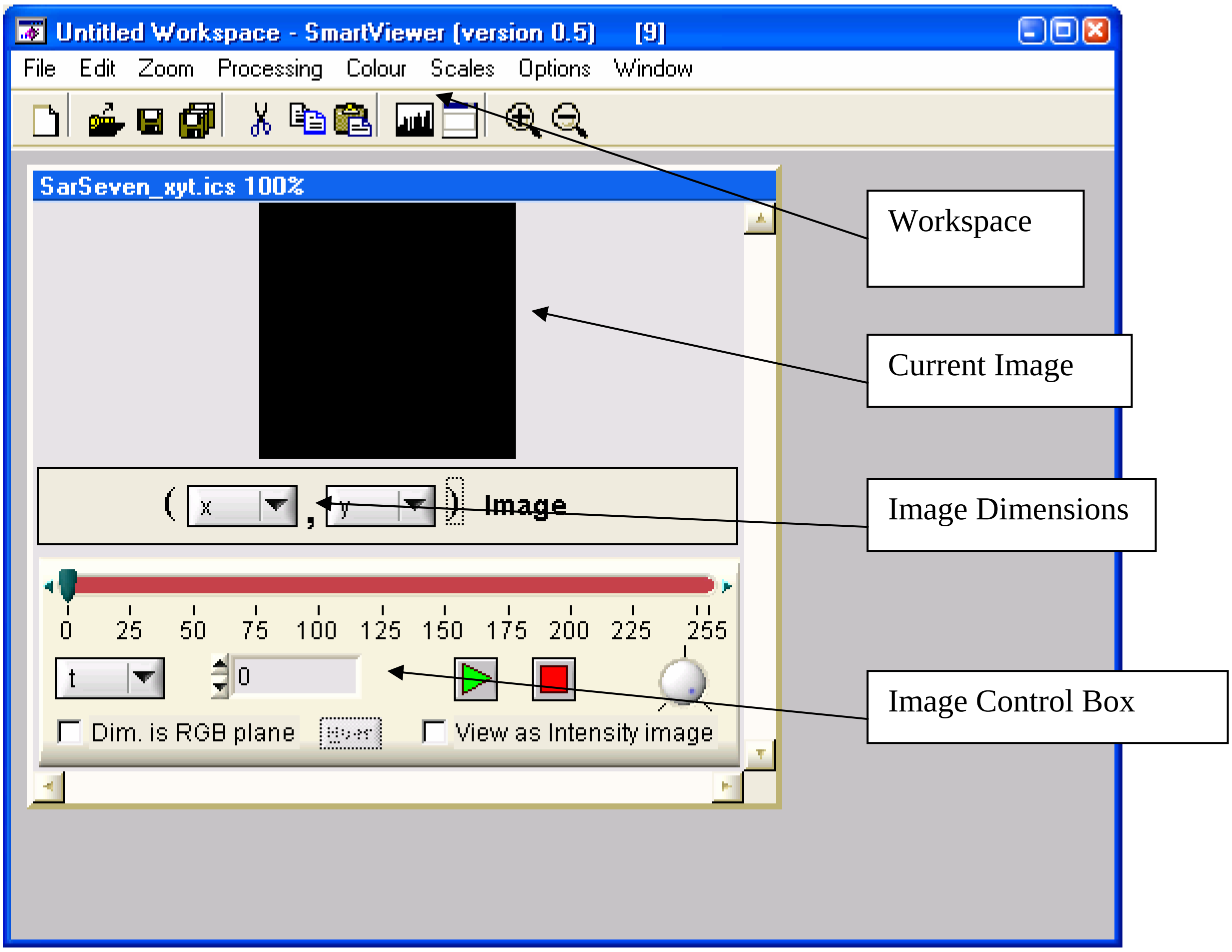
- Quick visualization and verification of generated data, particularly useful for data sets that cannot be loaded into standard imaging packages, such as ICS, Biorad PIC and SDT file formats etc.
- Output of images and movies from such data sets into standard formats (such as JPEG, Bitmap and AVI) for use in other applications or reports.
- Capability to load multi-dimensional data sets and give users some ability to view planes within the data set.
- Extraction and deletion of regions of data from within a set, to reduce file sizes and remove unwanted data or to further process specific dimensions/regions.
- Simple image processing functions, including contrast/brightness, histograms, colour tables and image arithmetic on data sets.
- **Reformatting data**, to allow alternative ways to viewing data and formatting for increased performance, described in more detail here.

As an example, a collection of images (known as a ‘stack’) can be loaded into SmartViewer, and any 2D plane (single image) from with that 3D stack can be viewed, and the combination of planes can be watched in sequence as a movie.

Once an image has been successfully loaded, it will appear in the Workspace looking something like this:



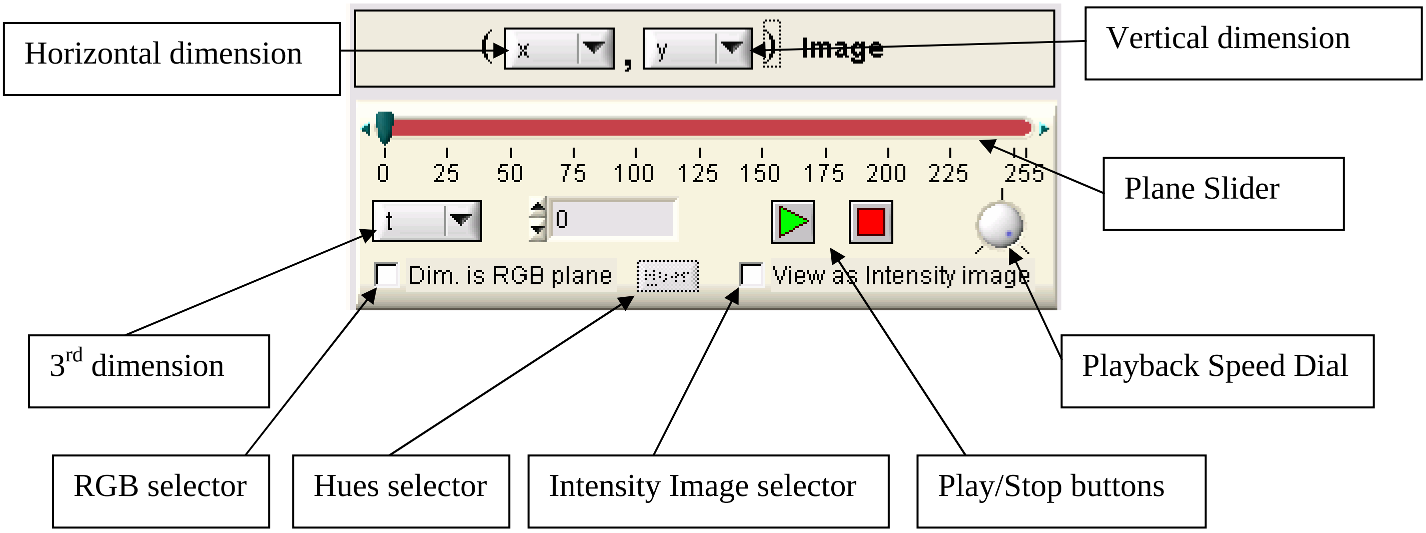
But the data set doesn’t just have to be 3-dimensional…



Note that you can load as many images into your 'Workspace' as you require (memory permitting).

If the image that was loaded was a normal 2D image, the 'Image Dimensions' and 'Image Control Box' controls will not appear. If, like this example, a data set with more than 2 dimensions is loaded, the controls will be available. The Control Box allows you to select which 2D plane from within your data set you wish to view.

Here is a closer look at the Control Box:



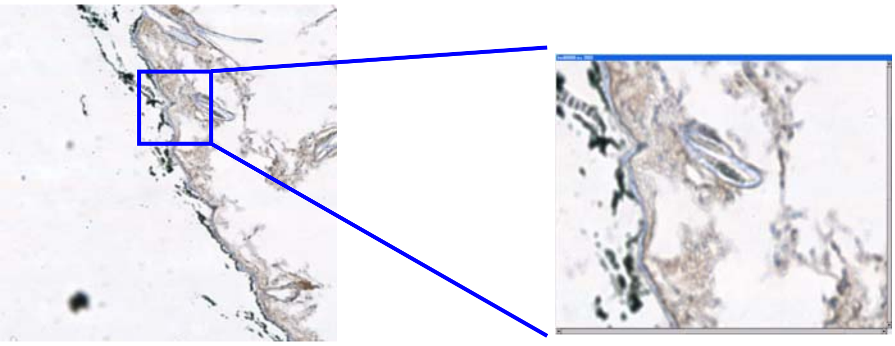
In the Control Box, the **Plane Slider** tells us that in the 3rd dimension (which is labelled '*t*' here), there are 256 individual planes to view (0 to 255). Dragging the Green knob on the slider flicks through the planes, updating the currently viewed plane displayed.

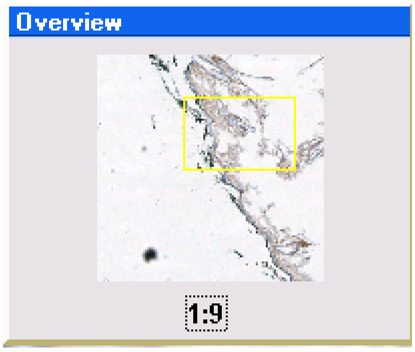
You can also use the **Play** button to watch a 'movie' of all the planes in the 3rd dimension shown consecutively. The movie will loop back to the beginning once the end has been reached. You can adjust the speed of the movie playback by adjusting the **Speed Dial**, and use the **Stop** button to finish playing the movie.

(Advanced controls such as the RGB and Intensity Image selector and the Hues button are discussed later.)

Once a particular 2D plane has been selected, you can use various SmartViewer features for analyzing that plane. For example, you can zoom in and out of an image to gain or decrease detail and to look closer at areas of interest in a large image. Use the PageUp/Page Down buttons, select the +/- magnifying glass icons from the toolbar or select Zoom→Zoom In/Out from the menu to do this.

If the image is too large to display in the current Workspace, scrollbars will appear to help you move the Field of View around the image.





To help navigate around large or magnified images, you can use the 'Overview' window to get an idea of which region of a larger image you are looking at.

In this window, the yellow box shows the current Field of View that is displayed on the current image. You can click and drag inside the Overview window to move the Field of View around quickly.

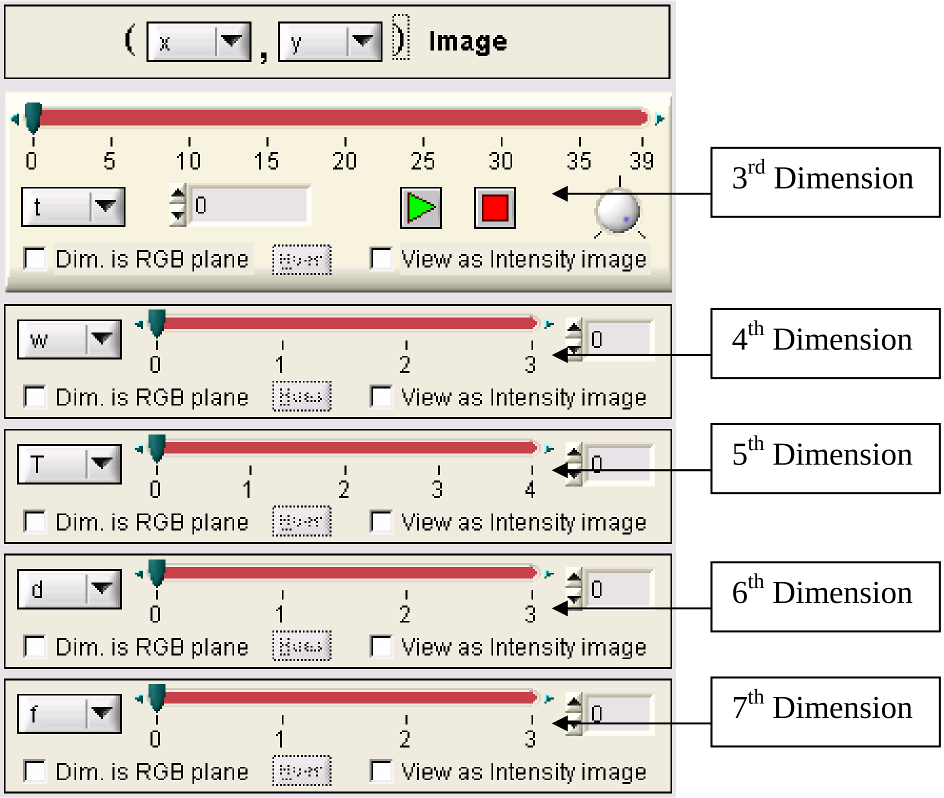
In summary, the Control Box allows control over selecting for viewing an individual 2D plane from within a data set which has more than 2 dimensions in total. A 'movie' of consecutively displayed 2D frames of the 3rd dimension can be watched.

Once loaded, you can navigate around an image and view areas of interest by zooming in and out, and using the Overview window.

Four dimensions and upwards – the Control Box expanded

If the data set that is loaded has more than three dimensions, the Control Box expands to add controls allowing for the manipulation of these dimension and selection of planes within them. Here is the Control Box where a 7-dimensional data set has been loaded.

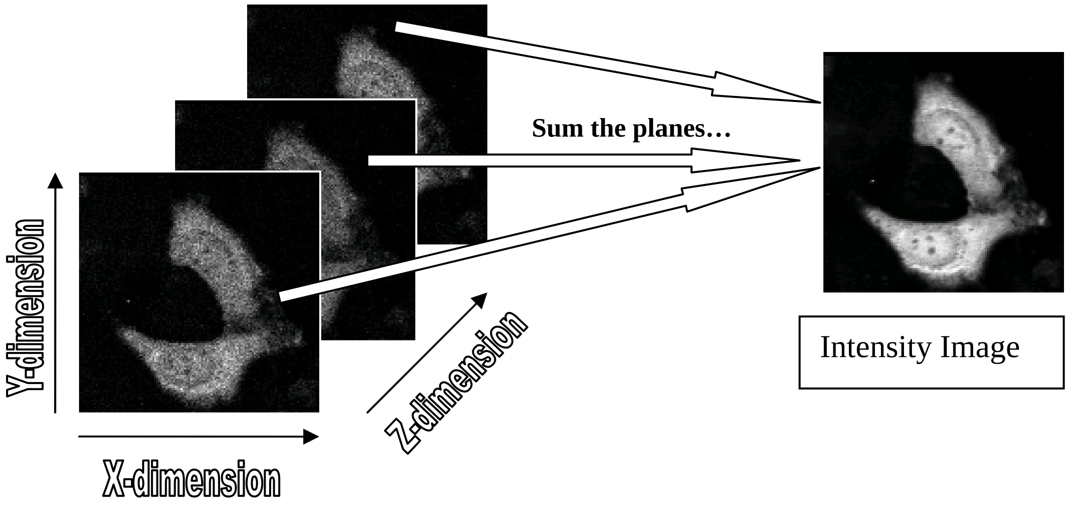
Here you can see that for each extra dimension, an extra area on the Control Box has been added. Each has its own **Plane Slider** which can be dragged to select a plane from within the data. The combination of all the sliders for a data set uniquely specifies a single plane from the data set, which is the one displayed.



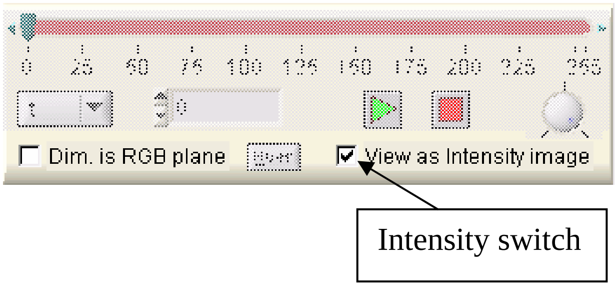
Note that you can still use the Play/Stop buttons to play a movie of the 3rd dimension, whilst dragging around the other sliders to display different regions of data.

Viewing Colour (RGB) and Intensity Images

A useful way to display and verify multi-dimensional image data is to view all the images in a particular dimension summed together to produce a single Intensity image.



To tell SmartViewer that you wish to view a particular dimension as an Intensity image, click the 'View as Intensity Image' button on the required dimension control. You will notice that the 'Play' controls dim, because we are now viewing this dimension as a single summed image. Click the button again to return to normal view.



□ Dim. is RGB plane

☑ View as Intensity image

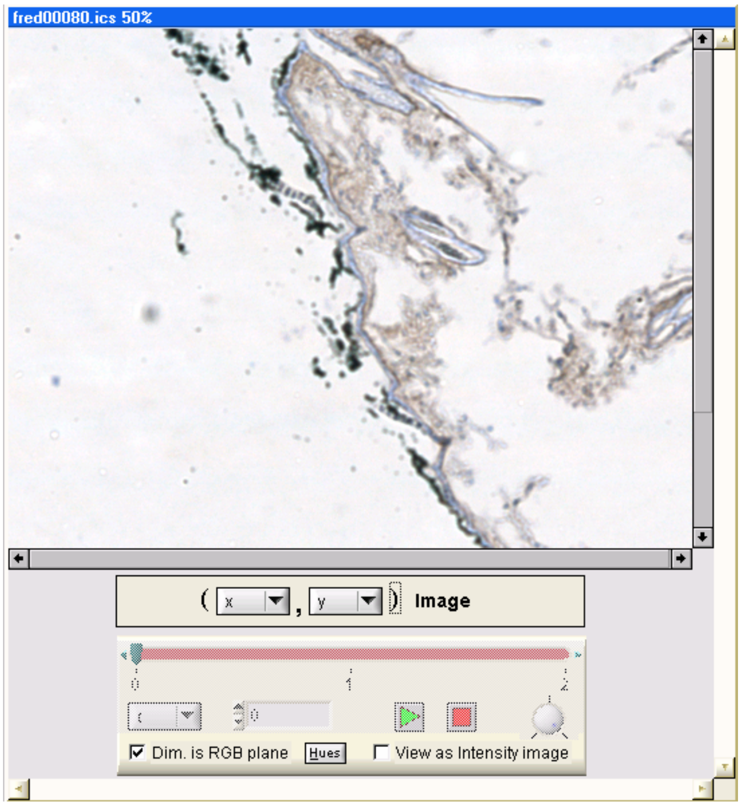
Intensity switch

Colour Images

So far, this Guide has only considered viewing and manipulating Greyscale images. SmartViewer has the ability to load and display Colour (RGB) images.

Unlike Greyscale images, which are simply made up of a single value (e.g. between black and white in most default views) for each pixel in the image, colour images are usually stored as a set of 3 values per pixel, one each for Red, Green and Blue channels which combined together provide a single colour pixel. So, a colour image can be considered as a combination of three greyscale planes: one of each red, green and blue, and a single colour image is therefore a 3-dimensional 'stack' of 'greyscale' images. SmartViewer gives you the ability to each of these planes as a greyscale image, or view their combination as a colour RGB image.

When a data set is loaded that SmartViewer can determine contains colour data, a switch is set on the user interface for the dimension that has been determined, as shown on the right.



☑ Dim. is RGB plane

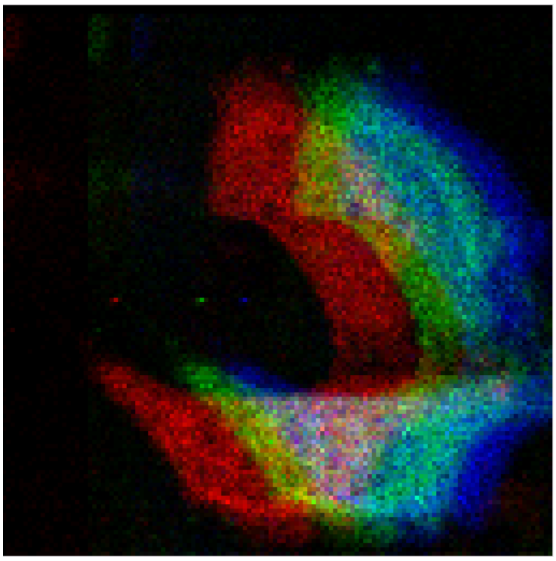
□ View as Intensity image

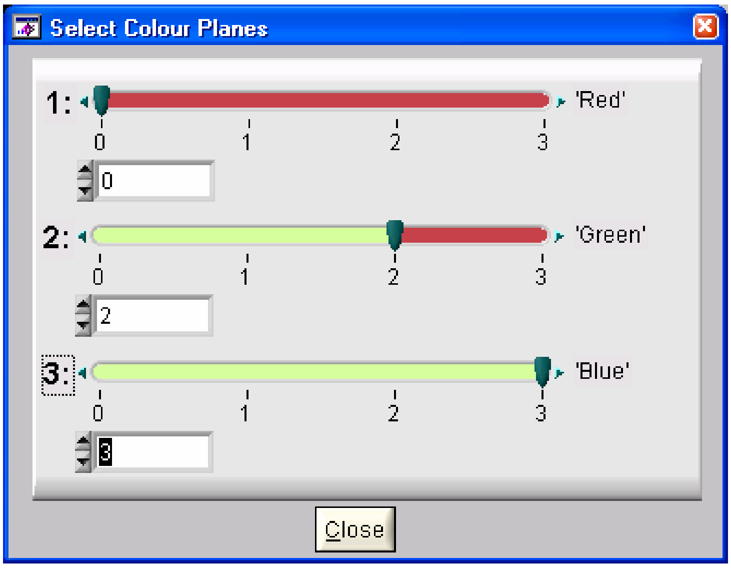
Here you can see that a colour image has been loaded, and in the Control Box, the switch labeled 'Dim. is RGB plane' has been ticked, telling SmartViewer to interpret this dimension as a colour image. Note that the 'Play' controls have been disabled, since they do not make any sense if this dimension contains colour information.

Unticking this box will allow you to view each of the three planes individually as greyscale images as usual. Also, if the data set contains 4 or more dimensions (such as a stack of images with colour information) you can tell SmartViewer to interpret any of those dimensions as containing colour information, and for example watch a colour movie of that data set.

Advanced: Selecting Hue planes

By clicking the ‘Dim. Is RGB plane’ button, you can tell SmartViewer that you wish it to interpret **any** dimension from a data set as containing colour information, even if it has more (or less) than the 3 planes expected. SmartViewer then provides the option for you to assign the R, G and B planes yourself to any 3 planes from a particular dimension. This allows the creation of some interesting and informative RGB images. For example, the image below shows a cell that is ‘moving’ from left to right across the frame by assigning the RGB planes across that dimension:




Select Colour Planes

1: 'Red' 0

2: 'Green' 2

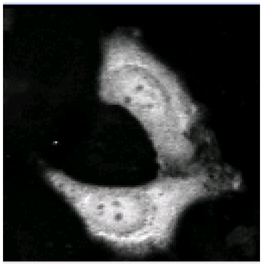
3: 'Blue' 3

To assign these planes yourself, click the Hues button, and the above panel will appear, from which you can assign the 3 colour planes.

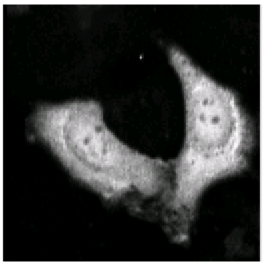
Reformatting data

SmartViewer allows you to view different planes from within a multi-dimensional data set. These planes do not have to correspond to the way that the data was originally created – the data can be reformatted so that alternate planes can be displayed.

As an example, a normal 2D image would have dimensions *[X, Y]*. Such an image can be reformatted so that it is stored as *[Y, X]*. This would have the effect of rotating the image, as shown on the right.

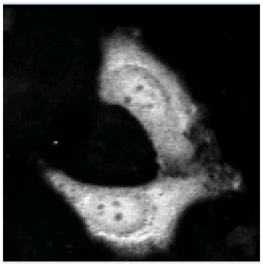


[X, Y]

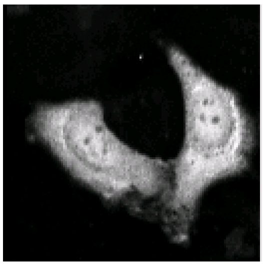


[Y, X]

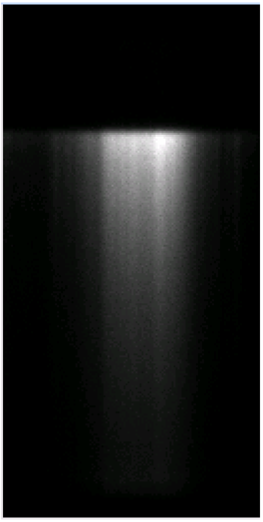
While this is of limited use for 2D images, for multi-dimensional data reformatting becomes more useful, particularly in terms of speeding up certain computational tasks. A 3D data set (our stack of 2D images) would have dimensions *[X, Y, Z]*. There are a number of different ways that a 3D data set can be reformatted, producing these resulting images (Note: these are Intensity images):



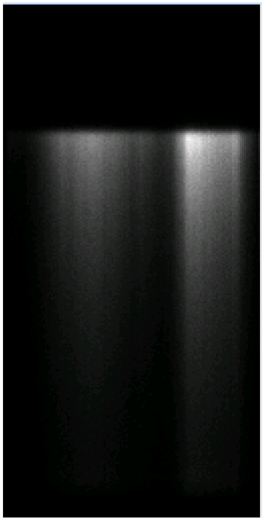
[X, Y, Z]



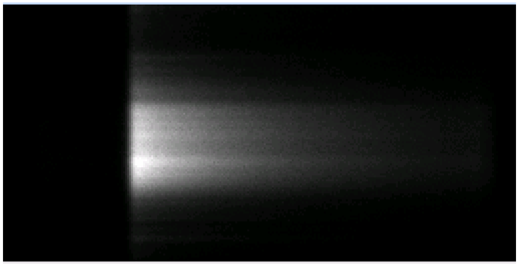
[Y, X, Z]



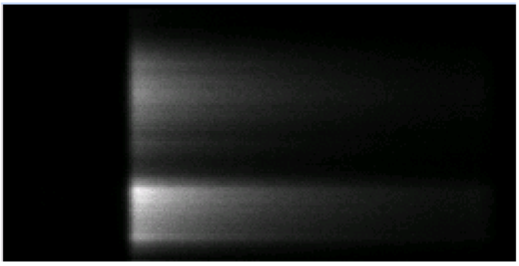
[X, Z, Y]



[Y, Z, X]

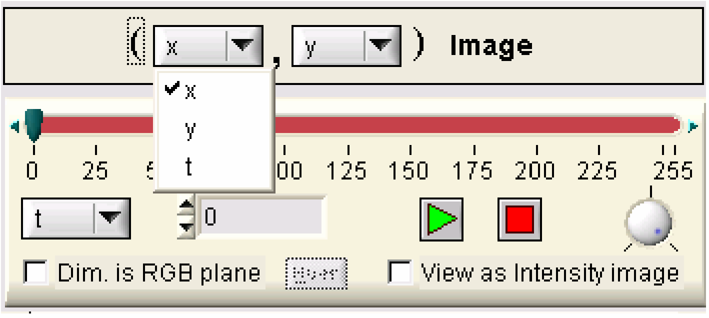


[Z, X, Y]



[Z, Y, X]

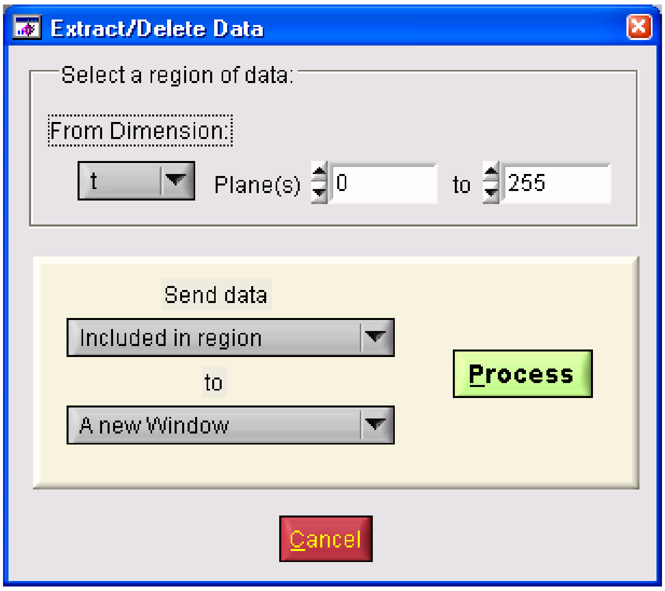
You can reformat your data by picking a required dimension from the drop-down lists available on the Control Box:



Selecting a new dimension will 'swap' those two dimensions and the data will be reformatted and displayed in the new format (for example in the image above the data is set up as *[X,Y,T]* format, and selecting *[T]* in the first dimension will reformat the data to *[T,Y,X]* format).

Any relevant History information that is stored with the image that corresponds to dimensional order (such as data extents, offsets etc.) will also be reformatted in the new image.

Deleting/Extracting Data



Extract/Delete Data

Select a region of data:

From Dimension: t

Plane(s) 0 to 255

Send data Included in region to A new Window

Sometimes it may be convenient to extract out a region of a data set to perform analysis specifically on a subset of data, perhaps to save time, disk space or compare two regions within a single data set independently. SmartViewer gives you two options – you can either **delete** data from within a set, or **extract** (copy) out a subset to view it as an entity in its own right.

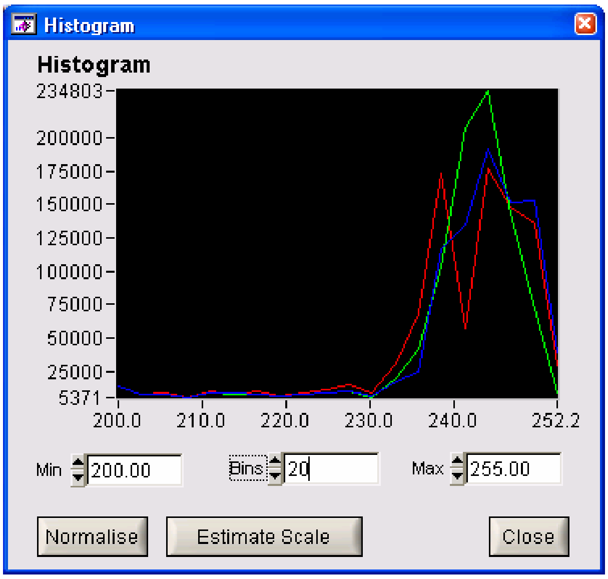
Selecting a region of data you wish to process is performed by indicating the start and end planes between which you wish to process.

You also can select which dimension it is that you wish to remove planes from by picking a dimension from the drop-down list, which will reformat the data so those planes can be accessed. For example, as well as being able to reduce the number of planes in a stack of images, you can crop all the images in the stack in both the X and Y dimensions by reformatting the data appropriately.

Other interesting features of deleting/extracting include the fact that creating a new data set with a single plane from any dimension will reduce the dimensionality of the data set by one (so deleting everything but one plane from a 4D data set will leave a 3D data set). This may be useful for producing from high-dimensional data sets extracted regions that can be loaded into other packages that do not support multi-dimensional data.

Miscellaneous

Here are a few of the other features of SmartViewer, described in brief:



Histogram

Min 200.00 Bins 20 Max 255.00

- You can view a Histogram of any image loaded in a Workspace by selecting Processing→Histogram from the menu, or clicking on the Histogram button on the Toolbar. This will display a histogram of the current image. If the image is a colour image, 3 histogram plots will be displayed – one per RGB plane coloured appropriately. You can use the Min and Max controls to set the threshold values for you image as required, and alter the binning for the histogram plot. Finally, you can use the Normalise button to automatically set the Min and Max thresholding values to the minimum and maximum values found in the data set, and Estimate Scale will attempt to find a pair of thresholding values that display the image data appropriately.

- You can use the Cut/Copy/Paste image controls to copy images into and between Workspaces. These work exactly like the same controls is MS Word etc., and are useful for making separate copies of data sets before processing them separately. Select Edit→Copy/Cut/Paste from the menu or use the toolbar icons. Also, the Copy command will set the currently displayed image onto the Window Clipboard, so that you can then paste the image quickly into other applications such as MS Word, Paint etc.

- In the same manner, you can Copy and Paste Image Scales between images to propagate thresholding values across or between Workspaces. Selected Scale→Copy/Paste Scale from the menu. You can also save and recall Scales from this menu.

- You can adjust the Contrast and Brightness for an image upwards or downwards to make an image easier to see by selecting Colour→Contrast/Brightness from the menu.

- You can also apply different colour tables to you images (not colour images) by selecting Colour→Apply from the menu and selecting a colour table. There are a variety of tables provided, including Seismic, Rainbow and Red, Green and Blue.

- You can perform simple Arithmetic calculations (Add, Multiply, Divide and Subtract) on multi-dimensional data sets. However, the two data sets that you select must have **identical** dimensionality. Select Processing→Arithmetic from the menu, and then pick the two data sets from the drop-down menus that you wish to perform the arithmetic on. Note the warning message on the panel that informs that the History data for the data set that will be created from the processing will be carried across from the first image selected for processing. Therefore, if there is some History information that you would like to keep with the new data set, make sure that the appropriate value is set for Image 1 before processing.

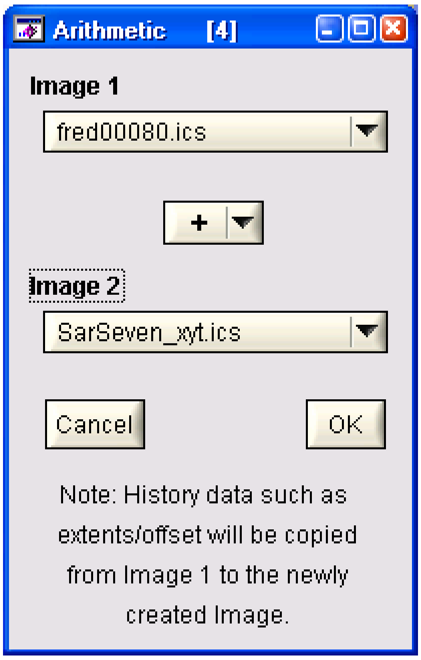


Image 1: fred00080.ics

+

Image 2: SarSeven_xyt.ics

Note: History data such as extents/offset will be copied from Image 1 to the newly created Image.

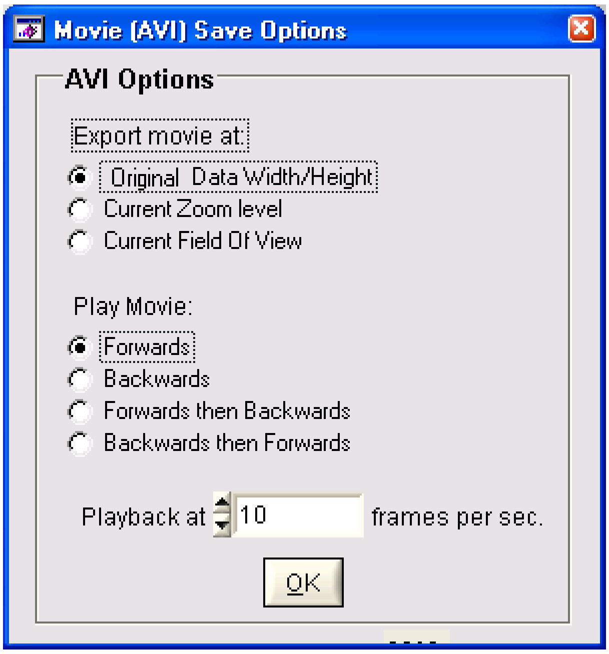
Finally: Saving data to file

You can save a copy of the data you have opened/modified to an ICS file, to which will be added the appropriate History entries describing the data set. To save your data as an ICS file select File→Save File (shortcut Ctrl-S) from the menu.

For use in other applications, you can export data in different file formats. At any time you can save the currently viewed 2D plane as a single image in the common format JPEG, Windows Bitmap (BMP) and as a single-frame AVI movie. To do this, select File→Export Plane as 2D image from the menu and choose the desired format from the file popup that appears.

You can also export the entirety of the 3rd dimension (the ‘Play’ dimension) as an AVI movie. This will create (in high-resolution/interpolation) the movie that is seen when the Play button is pressed. To do this, select File→Export as AVI movie from the menu.

There are a number of options that can be selected for saving to AVI files. To choose them click File→AVI Movie options from the menu. The following ‘options panel’ will appear:



Movie (AVI) Save Options

AVI Options

Export movie at:

● Original Data Width/Height
○ Current Zoom level
○ Current Field Of View

Play Movie:

● Forwards
○ Backwards
○ Forwards then Backwards
○ Backwards then Forwards

Playback at 10 frames per sec.

For a measure of control over the AVI file produced, you can select the output file to be displayed at:

The Original Data Width and Height, which will produce a movie at the original size of the data.
The Current Zoom level, which will produce an AVI at the current zoom level of the data.
The Current Field of View. This will output a movie file of exactly what is displayed when the play button is depressed, so that if you wish to zoom in on a particular section of data and just output a movie of that section, select this option.

You can also choose whether the movie should play forwards, backwards, forwards then backwards or backwards then forwards.

Finally, you can choose in Frame per Second, the playback speed of the movie when it is viewed in another application.

Now, even with all these features, it has to be admitted that some form of brain transplant may be needed to fully appreciate seven-dimensional images. Nevertheless, these tools are not just useful for viewing images as they can be effective in determining personal integrity: if you demonstrate what the package can do to a colleague or visitor for the first time and they use expressions like "I see" or even worse "I understand", you can be pretty sure that they are not telling the truth. If on the other hand they say "show me this again" or "what does this mean?" they will be productive collaborators! An 'in between' expression such as "that's interesting" is probably acceptable and will make you feel good….