Envision Toolbox Manual

Chapter 8

Spectrum

Interactive Classification of Images

Chapter 8 - Spectrum

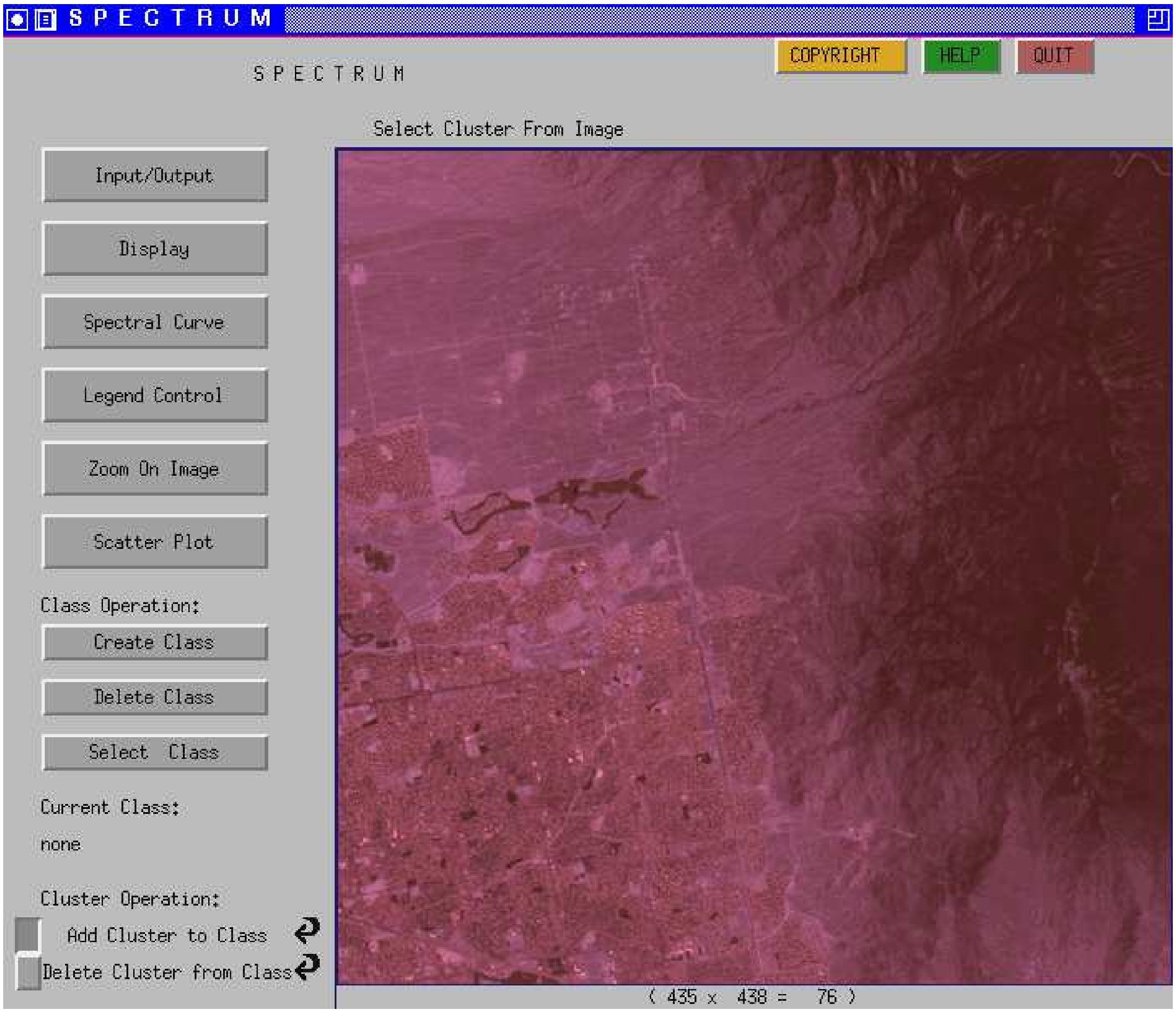


Figure 1: The spectrum application is used for interactive classification of images, such as this pre-clustered image of the northeastern Albuquerque metropolitan area ($ENVISION/data/spectrum/albq3.xv).

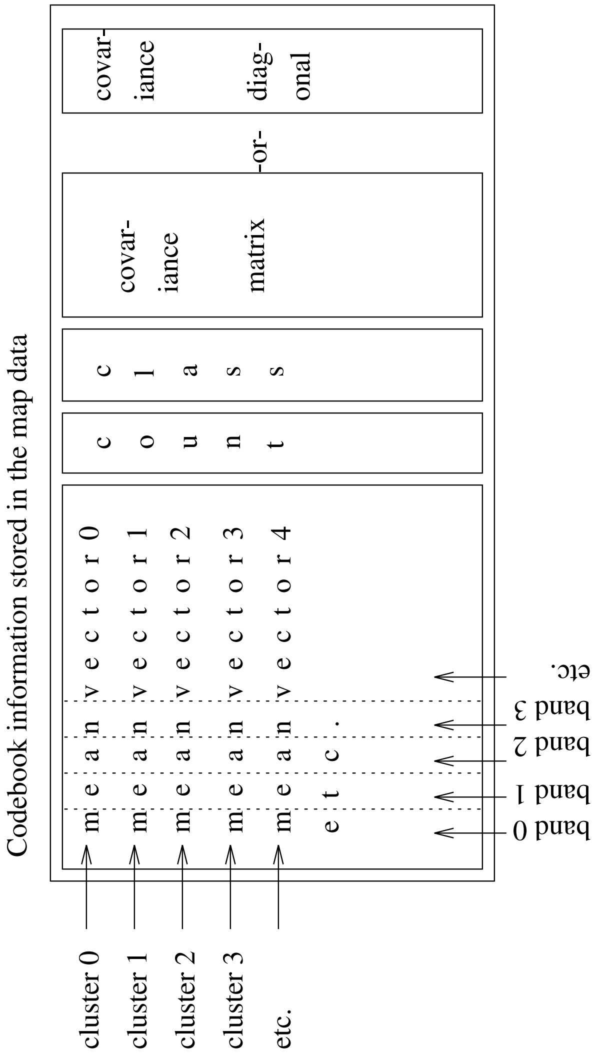
A. Introduction

Spectrum is an interactive image classification application. After an image has been clustered, spectrum is used to assign clusters to categories of interest for mapping.

You begin the classification procedure by inputting the clustered data and its associated cluster information into spectrum. An input image may be specified using the [-i] command line argument, or using the "Files" subform. Spectrum supports the same file formats as the other envision applications, but focuses on the

Khoros 2.0 VIFF and Khoros 1.0 xvimage file formats, each of which can accomodate the specially defined way in which spectrum expects the *codebook*, or associated cluster information produced during clustering, to be stored. This codebook is stored in the *map data* of the image.

In the codebook are rows of cluster center information, also called *mean vectors*; there is one mean vector for each cluster. Taken as a group, the mean vectors form *bands*, or columns of data that represent different wavelengths of spectral information. The codebook also includes cluster statistics, such as covariance matrix and cluster count.



After the clustered data is displayed in the main spectrum workspace, the Display subform allows you to choose three bands for display on the screen; one band will dictate the red values of the pixels, the next band will specify the green values of the pixels, and the third band will give the blue values of the pixels. Together, the three bands selected as red, green, and blue determine the colormap used to display the data; the resulting display is an approximation of the original image as rendered through the spectral band means for the several clusters.

Analysts with photointerpretation experience will probably choose either a band combination that gives a "color-infrared" view or a "conventional color" view. Each has advantages for interpreting particular types of landscape features. Various "indexes" such as greenness, brightness, wetness, and so on, can also be displayed if the analyst is familiar with their formulation as ratios or linear combinations of spectral bands; these equations can be input directly to spectrum, or for convenience can be defined in an ascii *display environment* file that may be provided to spectrum with the [-t] command line argument or with the use of the Files subform.

Next, the classes that will be used to categorize the features of interest in the image must be defined. The classes can be defined by clicking on the "Create Class" button of the main spectrum form, or with the use of an ascii Legend file which may be read in using the [-l] command line argument or with the use of the Files subform.

After categories for interactive classification are established, you can assign clusters in the image to a particular class by holding the mouse button down and moving the pointer over the desired area in the image ("painting"). To add a single cluster to a class, you may double click on the desired cluster.

A zoom window provides an additional degree of accuracy for examining the image. The zoom window can also be used for "painting" clusters into classes, or for double clicking on a particular cluster that is to be added to a class.

Next, the Legend subform provides red, green, and blue scroll bars that allow you to assign a special color to each class; clusters in that class will appear in that color, making them distinguishable from unassigned clusters, or from clusters that belong to other classes. Most users prefer to keep the Legend subform open as they create new classes, so that the class color can be defined as soon as the first cluster is added to the class, and the difference between clusters in one class and clusters in another class is immediately apparent.

When the classification procedure is complete, the image can be written out along with a legend file for later use.

A "spectral response curve" displays a graph of the mean vector for a cluster; the plot updates to reflect the mean vector for the cluster under the pointer as the pointer is moved across the image.

A 2D scatter plot allows you to plot selected bands as X and Y. In another method of assigning clusters to classes, you may "paint" on the scatterplot to indicate which clusters are to be assigned to a class.

At any time, statistics about the clustered image can be displayed. As the pointer is moved across the image, the values stored in the mean vector for the cluster under the pointer are automatically updated. The contents of each class can also be viewed.

The graphical user interface of spectrum contains several subforms through which you may access the different features of spectrum; these subforms may be accessed by clicking on the desired subform button that appear on the left hand side of the graphical user interface.

B. Creating Classes, Adding and Deleting Clusters

To create a class, click on the "Create Class" button. This will pop up a query object in which you can enter the name for the class. After entering the desired name for the class, click on the "Use" button. The class will be created; initially, there will be no clusters in the class.

Before assigning clusters to a class make sure that the current class is correct. Use the "Select Class" button to select the "current class". The "current class" always dictates which class is having clusters added to it, clusters deleted from it, the name changed, the color changed, etc. By default, the current class is always the most recent class to be created.

To assign clusters to a class, position the pointer in the image over the clusters which are to be added to the class. Hold the first mouse button down, and move the pointer slowly over the image; each cluster that appears under the pointer will be added to the current class. This procedure is referred to as "painting" clusters into a class.

The zoom box can also be used to add clusters into a class. Click on the "Zoom on Image" subform button to display the zoom subform. Change the "Update Mode" logical selection on the Zoom subform from "Continuous" to "Button Press". Then, using the same "painting" method in the zoom box rather than the image, indicate which clusters are to be added to the current class. The zoom subform has its own section with more details.

The scatter plot provides a third method of adding clusters to a class. Open the scatter plot subform, and use the "painting" method on the points in the plot that represent the clusters to be added to the current class. The scatter plot has its own section with more details.

To remove clusters from a class, change the "Class Operation" toggle from "Add Clusters" to "Delete Clusters". Then, use the "painting" method to delete clusters from the current class as was used to add clusters to the class.

Frequently, when adding clusters to a class, it is helpful to have the Legend subform open in order to see the legend describing the classes that have been created. In addition, the color in which the class appears is controlled from the Legend subform; as soon as at least one cluster is added to a class, it is helpful to set the color of that class so that the class is distinguishable from the rest of the image. The legend subform has its own section with more details.

If you know the numbers of the clusters that you would like to add to a class or delete from a class, you may specify those clusters directly, without using the "painting" method. First, be sure the current class is correct. Then, click on the button labeled, "Add Clusters" or "Delete Clusters". A pop up prompt will appear, in which you may type the numbers of the clusters to be added or deleted from the class.

C. GUI Overview

Files
Used to input and output images and legend files. Can also be used to output the colormap as it appears in the image.

Display
Used to change how map columns are displayed when additional information from a pre-clustering algorithm is provided.

Spectral Response Curve
Provides a plot giving information about the values of the map data associated with the pixel (cluster) over which the cursor is positioned.

Legend Control
Used to modify the legend of classes.

Zoom On Image
Enables the user to zoom in on the image; the zoom window can be used to add/delete clusters from a class instead of the main image workspace if desired.

Scatter Plot
An alternative way to examine clustering data; the scatterplot window can be used to add/delete clusters from a class instead of the main image workspace if desired.

Cluster Stats
Prints the values in the mean vector of a cluster, as the pointer is moved over that cluster in the image.

Class Contents
For each established class, prints the clusters that currently belong to that class.

D. Command Line Options

The full syntax for executing spectrum from the command line is as follows:

```
% spectrum
        [-i]      (infile)  Input Image
        [-l]      (infile)  Input Legend
        [-t]      (infile)  Input Display Environment File
        [-visual] (cycle)   Read Only or Read Write Visual
```

[-i {input image filename}]
This is the input file on which classification will be performed. For more information, see the section on "The Files Subform".

[-l {input legend filename}]
Images and legend files should always be written out as pairs from spectrum; together, the output image and the output legend file completely describe the classification information that was established using spectrum. For the same reason, then, partially or completely classified images read back in should always be accompanied by their associated legend file. Thus, when the [-i] option is used to input a file containing an image that has already been classified, the [-l] argument should also be used to specify the corresponding legend file. For more information, see the section on "The Files Subform".

[-visual { 1 'Read Only', or 2 'Read Write'}]
This argument dictates whether spectrum is to use a ReadOnly visual or a ReadWrite visual. A *visual* describes the characteristics of a virtual colormap that is created for use with an application on a particular screen. For a complete explanation of visuals, you are referred to Chapters 7.2, 7.3,

and 7.4 of *The XLib Programming Manual*, by Adrian Nye, published by O'Reilly & Associates. Depending on your particular workstation, you may have a DirectColor, GreyScale, PseudoColor, StaticColor, StaticGrey, or TrueColor visual. The former three visual types support two types of colorcells: *read only* and *read/write*. The color in a read only cell is set once by one application, and from then on can be shared by multiple applications, but not changed. A read/write color cell can have its color changed at any time by the application that allocated it; however, it cannot be shared by other applications. The latter three visual types are limited to read only colorcells.

By default, spectrum will allocate color cells as read only. However, if you are on a workstation that has a visual of type DirectColor, GreyScale, or PseudoColor (`% xdpyinfo` will tell you the type of visual on your workstation), you may use the [-visual] argument to instruct spectrum to use the read/write method of color allocation. There are tradeoffs that must be taken into account when setting the color allocation method; again, you are referred to *The XLib Programming Manual* for more information.

To summarize very briefly, spectrum will be able to update the colors that are displayed in the image *much* faster when a read/write color allocation method is used. However, since color cells will not be shared with other applications, this may result in the annoying "techno-flashing" phenomenon where the colormaps of spectrum and any other applications that happen to be running at the same time are installed and de-installed as you move your pointer. Thus, if *spectrum* is the only application requiring a large number of colors that you happen to be running, use of the read/write color allocation scheme is recommended. Technoflashing will increasingly become a problem as spectrum has to compete with other applications for color. If technoflashing is a problem that bothers you, it is best to leave spectrum with the read-only color allocation default behavior. However, you will notice that colors are slower to refresh, particularly the colors of classes with a large number of clusters assigned.

E. Input / Output

The Files subform allows you to read in Khoros 2.0 VIFF or Khoros 1.0 xvimage images to be classified. Input of ascii legend files is also done from the Files subform, as is input of display environment files.

Output of files is also done from the Files subform. Partially or completely classified images can be written out in Khoros 2.0 VIFF format; with each classified image, a legend file should also be written out (images and legend files should always be output as pairs). In addition, the "grouped image" output option allows you to re-assign the cluster values in a completely classified image such that each cluster value is the same as the ID number of the class that it has been assigned to. Finally, the Files subform supports output of only the codebook information, without the actual image data.

E.1. Input of Image Files

Pre-clustered input files to spectrum may be Khoros 1.0 xvimages with image data of type BYTE, or they may be VIFF 2.0 images written out by spectrum or one of the clustering programs of Khoros 2.0. Non-pre-clustered images may be of any file format currently supported by Khoros 2.0. Input files to spectrum fall into two categories:

1. Pre-clustered images with floating point codebook information stored in the map data

2. Unclustered images that have no codebook

E.1.1. Pre-clustered Images

Here, the cluster numbers of the pre-clustered image are stored in the image data of the VIFF image, and the floating point "codebook" values (additional clustering information obtained from the pre-clustering algorithms) are stored in the map data of the VIFF image. Cluster numbers serve as indices into the codebook. All columns of the map data must be of the same length, where that length is the number of unique clusters in the image. There should be one row of map data (one "mean vector") for each cluster.

The image data must be of type BYTE, SHORT, or INT. The map data must be of type FLOAT or DOUBLE. With Khoros 1.0.5 images, two of the spare xvimage fields, *ispare1* and *ispare2*, are used to indicate the layout of the clustering information in the VIFF map data. In Khoros 2.0 images, the attributes "colorColnum" and "spc_map_contents" are used for the same purpose.

The minimum required map information for pre-clustered images is the cluster center vectors (mean vectors). Mean vectors may reflect any features of the clustered image that are of interest. For instance, in a geological application, there might be 6 columns of map data where the first column stored values indicating the amount of water for that cluster, the second column stored values about the type of soil for that cluster, the third column stored values about the amount of vegetation for that cluster, and so on. When **spectrum** run, three of these columns will be displayed at any given time; one will be represented as the red values in the colormap of the image, another as the green values, and the third as the blue values. At any time, you may select which of the map columns you wish to display as which color; alternatively, you may elect to display a function of the values in the map columns as red, green, and/or blue (for more details, see the section on the Display Subform).

The map information may also include a count column in which the cluster number indexes the number of occurrances of that cluster number; using other terminology, this is known as a histogram. When possible, it is highly recommended that a count column be provided; otherwise, spectrum will be forced to compute it in a process that is particularly timeconsuming with large images.

Following the count column may be a class column, in which the cluster number indexes the number ID of the class to which that cluster belongs, or -1 if the cluster has not been assigned to a class. When spectrum outputs an image, it always writes out a class column to indicate which clusters have been assigned to which classes; thus, an unclassified image usually won't have a class column, while an image that has been partially or completely classified with spectrum always has a class column.

Clustering statistics may be preserved with the inclusion of a covariance matrix (upper triangle) of the mean vectors, or a covariance diagonal of the mean vectors. Note that the presence of a covariance matrix is mutually exclusive with the presence of a covariance diagonal. Neither the covariance matrix or the covariance diagonal is currently used by spectrum, but the covariance matrix or diagonal, if present, will be preserved by spectrum so that it may be saved with the output image.

The total number of columns in the mean vectors must be provided in *ispare2* (for VIFF 1.0.5 images) or in the attribute "colorMapColnum" (for VIFF 2.0 images). If the *ispare2* value or the "colorMapColnum" attribute is zero, **spectrum** will assume that you have no codebook information (see Unclustered Images).

The Khoros 2.0 VIFF structure's "colorMapContents" attribute, equivalent to the Khoros 1.0 VIFF structure's *ispare1* integer field, is an unsigned long mask. This mask indicates contents of the codebook: the presence of a count array, a class array, a covariance matrix, or a covariance diagonal, when any of these elements are present. Provide values for "colorMapContents" or *ispare1* according to the information present in the map data, using the following values, which may be OR'ed together:

0 0 0 0 0 0 - mean vectors only
0 0 0 0 0 1 - count
0 0 0 0 1 0 - class
0 0 0 1 0 0 - covariance matrix (upper triangle)
0 0 1 0 0 0 - covariance diagonal

For example, a 5 implies that in addition to mean vectors, the input image has a covariance matrix and a count column. When the image is written out, spectrum will add or modify the class column, but leave the mean vectors, count column, and covariance matrix unchanged.

The following diagram illustrates the layout of the codebook information as stored in the map data:

```
|-----------|              |--------------------------------
|        | 2|-------       |  .  .  .  |  |  |        | c d|
|cluster ---|       ---------->.  .  .  | c| c|        | o i|
| numbers   |              |  .  .  .  | o| l|        | v a|
| stored    |              |mean vector| u| a|        | a g|
| in VIFF   |              |  .  .  .  | n| s| covar  | r o|
| imagedata |              |  .  .  .  | t| s| matrix | i n|
|           |            | |  .  .  .  |  |  |        | a a|  |
|  values   |            | |  .  .  .  |  |  |        | n l|  |
| 0 - (m-1) |            V |  .  .  .  |  |  |        | c  |  V
|-----------|            m |---------->n------------------>N  M


N = # cols in entire map
M = # rows in entire map

n = #columns in mean vectors = vector dimension
  = image->ispare2 in 1.0.5 VIFF
  = "colorMapColnum" in 2.0 VIFF

m = # mean vectors total = M
```

Notes:

1. The order of the map columns MUST be as specified in the picture above. All columns MUST be of the same size. You specify which columns are present in your map data by using the mask described above in image->ispare1 for VIFF 1.0.5 images, or in "colorMapColnum" for VIFF 2.0 images. For example, your map data might contain 5 map columns of length 255, but nothing else. In this case, N = n = image->ispare2 = "colorMapColnum" = 5; M = 255; and image->ispare1 = "colorMapContents" = 0. Or, you might have 5 map columns of length 255 and a class column. Again, in this case, n = image->ispare2 = 5, and M = 255; but now N = 6 and image->ispare2 = 2.

2. Spectrum will create a count column and a class column for you if they do not exist, which will be written out when the image is output (with "colorMapContents" set accordingly). Spectrum will

NEVER change the values in the other columns.

E.1.2. Unclustered Images

Spectrum will also work with images that have not been pre-clustered, and therefore have no additional "codebook" information. In this case, the image to be input may or may not have a colormap associated with it. For greyscale images, spectrum will create three "bands"; the number of pixels used in the image will be counted, and three identical map columns of that size will be created with incremental values. Images with colormaps will be assumed to have three bands: red, green, and blue. RGB images are *not* supported. While unclustered images are valid input to spectrum, much of the functionality is lost when a codebook is not present.

E.2. Output of Image Files

Spectrum writes out VIFF 2.0 image files. When an image is written out with **spectrum**, the image data that was read in originally is written out untouched. The header is written out as it came in, with the addition of the "colorMapContents" and "colorMapColnum" attributes. The codebook written out will contain the original mean vectors. If there was a count column associated with the input image, that count column will remain unchanged; otherwise, the count column will be automatically generated by spectrum. In either case, the count column will be output as part of the map data, following the mean vectors. A class column will always be output, in order to save the classes and their associated that have been defined with spectrum.

Note that a legend file should also be output at the same time, as the legend information and the class column together completely define the classes at any given time. If either a covariance matrix or a covariance diagonal was provided as part of the input map data, this information will be written out untouched.

E.3. Input & Output of Legend Files

The legend file used by **spectrum** is a simple ascii file, in the following format:

```
R G B 'title of class' class_id
```

The legend file, together with the class column that is maintained by **spectrum**, completely describe the classes defined at any given time. The class consists of n entries, where n is specified in the input image with *image->ispare2* or "colorMapColnum" for VIFF 1.0.5 images or 2.0 images, respectively. The index into the class column is the cluster number; the entry it indexes is the class id of the class to which that cluster number belongs. The class id is generated internally by **spectrum**, in order of creation. For instance, suppose cluster number 11 (pixel value 11) has been assigned to the fifth class, named "vegetation". Thus, class_column[11] = 5. Suppose further that the "vegetation" class has been given the color green - RGB values of [0 255 0]. The legend file, if written out, will have the entry:

```
0 255 0 'vegetation' 5
```

Note that in order to save the image exactly as it appeared when it was written out, it is necessary to write out *both* the legend file (giving the color, title, and class id) and the image (which will specify the class column listing which cluster numbers belong in the "vegetation" class). Later, to restore **spectrum** to its previous state, it is necessary to read in *both* the legend file and the image that were written out earlier.

If a legend file is read in with one or more entries that do NOT have any associated clusters in the class column (eg, if we read in the legend file with the entry above but had not yet assigned five classes) this will be indicated by the colorboxes in the Legend display appearing in white (indicating an empty class).

E.4. Input of Display Environment Files

The display environment file is an ascii file, which has three purposes:

1. It specifies the UTM coordinates and UTM zone of the upper left hand corner of the input image, if applicable.

2. It specifies names for each of the bands in the codebook, if applicable.

3. It specifies transformations, or sets of equations defining red, green, and blue that are specifically formulated to highlight particular landscape characteristics. If provided, these transformations will then become available via the "Transformations" button of the Display subform.

The information provided in the display environment file must appear *in the format specfied here, in the order given*. Make sure that you include tic marks as in the examples.

The first information is the pixel dimensions; that is, how many units (feet, meters, etc) are represented by each pixel in the X and Y directions. This information must be given exactly as in the following example; substitute your own dimensions for "30.0":

```
pixel_dim_x: '30.0'
pixel_dim_y: '30.0'
```

Next, give the UTM coordinate of the upper left hand corner of the image. The UTM coordinate must be given exactly as in the following example; substitute your own coordinates for the ones in the example:

```
utm_00_N: '4570950.0'
utm_00_E: '258720.0'
```

To end the UTM coordinate information, provide a UTM zone, as in:

```
utm_zone: '18'
```

Next, you may specify as many band name as are applicable to the input image, in the following form.

```
M0: ' TM 1 blue'
M1: ' TM 2 green'
M2: ' TM 3 red'
M3: ' TM 4 nearIR'
M4: ' TM 5 midIR'
M5: ' TM 7 midIR'
M6: ' TM 6 Thermal IR'
```

Finally, give as many transformations as you like, specified as follows. First is the name for the transformation, followed by a colon. Then, on three separately lines are the equations for red, green, and blue, preceded by the words "Red:", "Green:", and "Blue:", as in the following example.

```
ColorIR :
Red: 'M3'
Green: 'M2'
Blue: 'M1'
```

Note that the files may be commented by putting a "#" sign in the first column of the comment line.

Two example display environment files may be found in `$ENVISION/data/spectrum/environ.1` and `$ENVISION/data/spectrum/environ.2`. The "Aliases" file provided in the `$ENVISION/repos` directory provides aliases for these two files: "environ:1" and "environ:2".

E.5. Output of A Grouped Image

When an image has been *completely* classified, ie, every cluster in the image has been assigned to a class, you may output a *grouped* image. Instead of writing out the image data exactly as it came in, the "grouped" image has the image data modified such that each cluster number is replaced with the number of the class to which that cluster was assigned. Thus, if cluster number 60 had been assigned to class one, all instances of cluster number 60 in the image would be replaced with a 1.

With grouped images, the codebook information is *not* written out, as it will no longer be applicable; only the count column is preserved.

Only completely classified images can be written out as grouped images. If you have clusters left over that need to be "dumped" into a class, use the "Catch-All" category available on the Legend Control subform.

E.6. Output of Maps Only

Use this feature to write out the codebook of the classified image without the associated image data. The codebook is written out exactly as it would be when writing out an output image.

F. Display Capabilities

The "Display" subform allows you to normalize the colormap of the image. It also lets you control which bands or functions of bands are displayed as Red, Green, and Blue. In addition, you can use this subform to change the names assigned to each band.

F.1. Colormap Normalization

The colormap can be normalized if desired. Set the "Normalization Policy" logical selection to TRUE if you want the colormap to be normalized.

Then, set the "Type Of Normalization" to "Global" or "Local". With local normalization, the normalization procedure is performed locally using the maximum and minimum values of each of the red, green, and blue bands. For example, if the minimum and maximum values in the red column are 0 to 1, and the minimum and maximum values in the green column are 100 to 200, the red column will be normalized such that the value 0 stays 0, but the 1 value becomes 255, while the green column will be normalized such that the 100 value becomes 0, and the 200 value becomes 255. For global normalization, the normalization procedure is performed globally over all of the three columns displayed as red, green, and blue. Thus, with global normalization, if the minimum and maximum values over all the red, green, and blue columns are 1 to 200, all three bands will be normalized such that 1 becomes 0 and 200 becomes 255.

Finally, use the "Normalization Method" toggle to specify the method of normalization you prefer. When normalization is to be performed, this attribute defines the algorithm which is used in normalization. Normalization may be a simple stretch over the values that may be displayed on your workstation, or any of three variations on a standard deviation normalization may be used. For a simple normalization of values to within displayable intensity bounds, use "0 <= values <= Max Colors". Alternatively, images can have contrast increased or reduced using on of the standard deviation normalizations. "-1*(std.dev) <= values <= 1*(std.dev" will normalize values to within 1 standard deviation, "-2*(std.dev) <= values <= 2*(std.dev)" will normalize values to within 2 standard deviations, and "-3*(std.dev) <= values <= 3*(std.dev)" will normalize values to within 3 standard deviations.

F.2. Changing the Bands Displayed as Red, Green, Blue

By default, an image with a colormap has the first band defining the red values of the colormap, the second band defining the green values of the colormap, and the third band defining the blue values of the colormap, producing the results that you would expect for an image with a colormap, but without a codebook.

Clicking on "Change Red Map Col", "Change Green Map Col", or "Change Blue Map Col" will pop up a list of the bands stored in the codebook of the input file. You may then choose the band that is to provide the colormap values for red, green, or blue.

F.3. Applying Functions of Bands (Transformations)

There are a variety of "indexes" used to translate spectral information to particular types of landscape features. For example, you might define the red intensity as being dictated by the value in the fifth band, green as computed with the value (band 4 - band 3)/(band4 - band3), and blue as dictated by the second band. In this case, the "RED", "GREEN", and "BLUE" string selections may be used to define the function to be applied in order to produce the values that will be used as red, green, and blue. Functions must have *only* variable *M*; following each *M* must be a number *starting at zero* and ranging to *N-1*, where *N* is the number of bands available in the image. In our example, the equations would be:
Red: M4
Green: (M3-M2)/(M3+M2)
Blue: M1

Equation evaluation follows the standard rules of precedence; evaluation proceeds from left to right, and use of parentheses is fully supported.

As implied by the above example, the red, green or blue functions can be set to represent the band which is to be used to specify the red, green, or blue values of the pixels in the image, by suppling *Mx* as the "function" where "x" is the number of the band - 1 (remember that numbering begins at 0). This is equivalent to using the "Change Red Map Col", "Change Green Map Col", and "Change Blue Map Col" buttons.

You can also set the "RED", "GREEN", and "BLUE" functions to a constant, if desired. For example, if you wanted the red values of all the pixels in the image to be 200, you could enter "200" in the "RED" function.

F.4. Color Transformations

"Transformations", or sets of equations for red, green, and blue, formulated specifically to get particular types of landscape features, can also be supplied in a Display Environment File for convenient access. For transformations that are used often, use of the Display Environment File is recommended over manual entry from the spectrum GUI. For more information about the Display Environment File, see the section on the Files subform. When transformations are supplied in the Display Environment file, the "Transformations" button on the Display subform will be activated.

Clicking on the "Transformations" button will pop up a list of the transformations defined in the Display Environment file. Select the desired transformation from the list, and it will be immediately applied to the image. Note how this is a much more convenient method of applying frequently used transformations than entering each function for red, green, and blue by hand.

G. Legend Control

The "Legend" subform allows you to modify the legend of classes.

At the top of the legend subform is a set of action buttons. With these buttons, you may empty a class, create a "catch-all" class, "hide" a class by suppressing the display of the color with which it is associated, and "show" a class that has been previously hidden.

Next is a set of red, green, and blue scroll bars that are used to specify the color which represents a class. The clusters belonging to a class will be displayed in this color.

Finally, the Legend subform contains a scrollable list of the classes which exist in the current legend. Each "legend entry" in the list which consists of three items.

1. On the left is a colorbox which reflects the color in which the class is displayed. To change the current class, click the first mouse button in the color box.
2. At the top are the RGB values that define the color in which the class is currently displayed.
3. On the bottom is a button labelled with the name that identifies the class. Click the first mouse button on this label to change the name.

G.1. Changing the Color of a Class

First, select the current class by clicking the first mouse button in the colorbox associated with the desired class. Then, move the RGB scroll bars above the legend to until the colorbox of the class appears in the desired color. At the same time, all clusters that are members of the selected class will reflect the same color change in the image. By changing the color of the class, the locations of clusters in that class become readily apparent. The RGB values defining the color in which a class is displayed will be updated in the legend entry.

Spectrum has [-visual] option that allows you to use a ReadWrite visual rather than a ReadOnly visual. When the [-visual] option is set to ReadWrite, the process of updating the class colors in the image will be significantly faster. The drawback of using a ReadWrite visual, however, is that it may cause spectrum to "technoflash", or swap colormaps as you move your pointer in and out of the spectrum GUI.

G.2. Changing the Name of a Class

Each class is uniquely identified by a name. This name should be descriptive of the nature of the class, such as "Desiduous Forest", "Coniferous Forest", "Wetlands", and so on. To change the title of a class, simply click on the title button associated with that class. A prompt widget will pop up, in which you may enter the new name.

G.3. Emptying a Class of Clusters

At some time, you may wish to delete all clusters from a particular class, but at the same time to preserve the class itself for later use. To empty a class of clusters, click the first mouse button on the action button labelled, "Empty Class(es)". A list widget will pop up, listing all existing classes. Select the class(es) to be emptied, and click on "Use". Another pop-up prompt will appear, asking you to confirm your decision. When the action is confirmed, the class(es) selected will be emptied of clusters.

G.4. Creating a Catch-All Class

After the desired amount of classification has been accomplished, you may wish to dump all remaining clusters into a "catch-all" class. Click the first mouse button on the action button labelled, "Catch-All Class". A pop-up prompt will appear, asking you to confirm your decision. When the action is confirmed, every cluster that does not belong to one of the currently existing class will be made a member of the Catch-All Class.

G.5. Hiding a Class

Often it is useful to see only a subset of the classes displayed in the image, with the others "invisible". Click on the action button labelled, "Hide Class(es)". A list widget will pop up, listing all classes that are currently displayed. Select the class(es) to be hidden, and then click on "Use". All specified classes will then "disappear". Hidden classes will still be available for other operations, but the clusters in the hidden classes will now be displayed in background (or original) colors.

G.6. Showing (Re-Displaying) a Class

Once a class is hidden using the "Hide Class(es)" option, it is inevitable that sooner or later you will want to re-display the hidden class in its associated color. For this situation, use the "Show Class(es)" action button. A list widget will pop up, listing all classes that are currently hidden. Select the class(es) to be redisplayed, and then click on "Use". All specified classes will then "reappear", and all clusters belonging to the redisplayed class will again appear in the color associated with that class.

G.7. Transferring Clusters from One Class to Another

All clusters in an existing class class may be transfered to another class using the "Transfer Class(es)" action button. When you click on this action button, you will first be presented with a list widget containing the titles of all classes that are not empty. Select from this list one or more "Source" class whose clusters are to be transferred to another class. Click "Use" when ready. Next, another list widget will pop up, containing titles of all classes with the exception of those chosen from the first list. Select one "Destination" class to which all clusters of the "Source" class (es) will be moved.

H. The Zoom Window

The Zoom subform allows you to zoom in on the image. By default, the zoom window is in "Continuous" mode, which means that the contents of the zoom window are continually updated as you move the pointer across the image. You may also use the zoom box instead of the image to select clusters to be added to or deleted from a class. The "Zoom On Image" subform contains:

1. An "Update Mode" button, which allows you to control whether clusters are added to or deleted from a class using the zoom box or the image.

2. A float selection with which to change the zoom factor.

3. The zoom window itself.

H.1. Setting the Zoom Factor

To set the zoom factor, simply enter the desired number in the parameter box of the float selection labelled, "Zoom Factor:". Hitting <cr> will set the zoom factor. The change in the zoom factor will be immediately reflected in the zoom box.

H.2. Using the Zoom Window to Add or Delete Clusters

First, click on the "Update Mode" button to change the zoom update mode from Continuous to ButtonPress. This will cause spectrum to change from using the image workspace to using the zoom window to add or delete clusters. Next, click the mouse in the image to control the area of the image which is displayed in the zoom box. Now, follow the same procedure to add or delete clusters as usual, by "painting" in the zoom box.

To revert to the original procedure of adding and deleting clusters using the main image workspace, change the "Update Mode" back to "Continuous".

I. The Scatter Plot

The "Scatter Plot" subform allows you to create a 2D scatter plot with X and Y coordinates dictated by two of the bands in the codebook. You may zoom in on the plot for greater accuracy; you may also use the scatter plot in either its normal or zoomed forms to add or delete clusters from the current class. The scatter plot displayed may be proportional or non-proportional.

When the scatter plot is displayed, every cluster in the map columns will be plotted; however, some points may overlap others -- it may be necessary to use the "Zoom" option to see all the points. The scatter plot will be presented in different colors; unassigned clusters will appear in the same color as they do in the image, while assigned clusters will appear in the color that was specified for their class using the Legend subform.

An additional feature of the scatter plot is the "roving" marker. This feature highlights the point on the scatter plot that represents the cluster located directly under the cursor when the cursor is in the image window. It is sometimes difficult to see; look closely to see the roving marker.

Under the scatter plot are two labels which are continually updated as you move the pointer in the plot. The first label indicates the plot position, or the (x,y) values of a point in the plot as dictated by the bands that are currently displayed as X and Y. The second label indicates the cluster number associated with the points in the plot, and lists class membership (if any).

I.1. Bands Displayed as X, Y

A pair of action buttons are provided, labeled "Band Displayed as X", and "Band Displayed as Y". Use these buttons to specify which bands displayed as the X and Y coordinates of the points in the plot. Clicking on one of the buttons will pop up a list of all the available bands; select the band to be used as X or Y, and the plot will be updated accordingly. By default, the X coordinates come from band 0 and Y coordinates come from band 1.

I.2. Functions of Bands Displayed as X, Y

A pair of string selections are available, labeled "Function displayed as X", and "Function Displayed as Y". Use these selections to enter a function of the bands which will be displayed as the X and Y coordinates of the points in the plot. When specifying a function, you must ALWAYS use "M#" to specify the map column number. Remember that numbering of map columns begins with 0. For example, "M3" would actually stand for the 4th column, and the specification, "M0 + M3 / M4" would indicate that you wanted to take each value of map column 3, divide it by the corresponding value in map column 4, and add the result to the corresponding value in map column 0.

I.3. Zooming on the Plot

To zoom in on the plot, enter the desired zoom factor in the Zoom Factor parameter box, and hit <cr>. This will set the zoom factor. Next, click on the "Set Zoom" button; this will allow you to "rubberband" a rectangle about the area to be zoomed in on. To rubberband the zoom area, move the pointer into the image, click on one corner of the desired zoom area. Holding the button down, move the pointer to the opposite corner of the area on which you want to zoom; a rectangle will "rubberband" after the pointer. When the desired area appears within the rubber band, release the pointer. After the zoom area has been established with the rubber band, the plot will be redrawn, containing only the points in that area. To go back to the original plot containing all the points, click on the "Reset" button.

I.4. Adding & Deleting Clusters from a Class Using the Scatter Plot

First, use the "Select Class" button on the main spectrum form to select the current class. Next, use the "Cluster Operation" toggle on the main spectrum form to indicate whether the scatter plot will be used to add clusters to the current class, or delete clusters from the current class.

Now, move the pointer into the plot window. Holding the button down, use the usual "painting" motion to add or delete clusters.

J. The Spectral Curve

The "Spectral Response Curve" subform provides you plots representing the mean vectors associated with clusters. A different 2D plot is drawn for each cluster; the plot displays the cluster center information stored in the mean vector as the X values, and the band number for each cluster center value as the X values.

When the "Update Mode" logical selection is set to "Continuous (single plot)", the spectral response curve is constantly updated to display the information associated with the cluster currently under the pointer as you move the pointer over the image or over the zoom box.

When the "Update Mode" logical selection is set to "Button Press (multiple plot)", a new plot for each cluster will be drawn every time you click the button on a cluster in the image. This capability allows you to compare the mean vectors for different clusters. The "Clear" button may be used to clear the plot window.

K. The Cluster Statistics Subform

This subform displays cluster statistics, which are updated to refer to the cluster over which the pointer appears as the pointer is moved across the image. Cluster statistics displayed include the cluster number, the cluster count, and class membership (if any). Then, the values in the mean vector for the cluster are displayed, each preceded by the name of the band in which the mean value is stored.

L. The Class Contents Subform

The Class Contents subform provides a convenient way to see at a glance which clusters are assigned to which classes. The subform contains a workspace with a viewport; the current contents of each class are continually updated as clusters are added and deleted from classes, and as new classes are created or deleted.

M. The AutoClassify Capability

The "AutoClassify" button on the main spectrum form provides access to a special feature which allows you to automatically classify an image. When used intelligently, this feature can be used with very good results, while at the same time dramatically reducing the amount of time and effort necessary to satisfactorily classify an image.

The inclusion of the auto classification feature in spectrum was made possible by Jim White, Pat Kelly, Julio Barros, and Dave Modl, of Los Alamos National Labs. The algorithm used for the automatic classification procedure will not be described here; it is covered in the following references:

J. Barros, J. French, W. Martin, P. Kelly, and J. White. Indexing Multispectral Images for Content-Based Retrieval. University of Virginia Technical Report CS-94-40. In *SPIE Vol 2368 Proceedings of the 23rd AIPR Workshop on Image and Information Systems: Applications and Opportunities*. Washington, DC. October 12-14, 1994.

P.M. Kelly. An Algorithm for Merging Hyperellipsoidal Clusters. *Technical Report LA-UR-94-3306*, Los Alamos National Laboratory, Los Alamos, NM, 1994.

Important Note: The auto classification capability can only be used with pre-clustered images having a codebook stored in the map data. Furthermore, the map data *must* include the covariance matrix obtained during the clustering algorithm that was used to cluster the image. Attempts to use the auto classification feature with input images that do not contain a covariance matrix in the map data will result in an error message.

To use the auto classification capability, you must first establish the full set of classes to which clusters may be assigned. This may be done either by using the "Files" subform to read in a predetermined legend file, or by repeatedly using the "Create Class" button on the main form of spectrum to create the desired number of classes with appropriate names. After establishing the classes to which clusters are to be assigned, you must give the auto classification algorithm some "starting information". For each class that has been created, assign some clusters to the class; for best results, it is recommended that more than one "starting" cluster be added to each class. If a class exists for which no clusters in the image apply (for example, in the case of a predefined legend file which has been read in), you may leave the class empty. Be very careful when assigning the "starting" clusters to the classes; the more accurate the "starting information", the more successful the automatic classification procedure will be.

When you are confident that you have provided an accurate "starting point", click on the "AutoClassify" button. Be patient while the auto classification procedure runs; the procedure may take as little as 10 seconds with small(512x512), byte images, or as long as 15 minutes with large (2000x2000), short images. In any case, you may be confident that the auto classification procedure will be completed in a significantly shorter amount off time than the equivalent manual procedure would require.

Table of Contents