

A Data Flow Approach to Color Gamut Visualization

Gary W. Meyer and Chad A. Robertson
Department of Computer and Information Science
University of Oregon, Eugene, Oregon 97403

Abstract

Software has been developed to help engineers visualize the gamuts of color hardcopy devices. Software modules have been written so that color gamuts and color space transformations can be explored using a scientific visualization program. Tools have been created that allow the user to accentuate the differences between gamuts and to interact with the displayed data. The data flow architecture of the program makes it easy to add more color gamuts and to investigate additional color transformations.

Introduction

A color reproduction engineer needs tools to evaluate the color gamut for a new color hardcopy device. The color reproduction technique employed in the new hardware could change the characteristics of the printer's color gamut. Because color gamuts are three dimensional objects, it is straightforward to use computer graphics to visualize the size and shape of a color gamut. By looking at a computer graphic picture, a new gamut can be compared with an old gamut or the gamut of another color reproduction device. Computer graphic hardware also allows one to precisely control the color reproduction characteristics of a color television monitor. This makes it possible to accurately render the colors of the gamuts that are displayed. Because it facilitates the production of colorimetrically correct three dimensional pictures, it is natural to use computer graphics to make pictures of color spaces.[1,2] However, it is difficult to develop such interactive computer graphic programs from scratch, and it is hard to make modifications and extensions to them once they have been completed.

Recently, visualization programs have become commercially available that allow the user to modularly build on top of an existing visualization framework.[3] All of these visualization systems employ a data flow architecture that permits the analyst to interactively define how the information is to be processed. Software components, called modules, are connected together in a network to form the visualization application. Modules take input data, process it under the control of parameters that are interactively specified, and produce output data. There are three basic types of modules. Source modules generate data by executing an algorithm or by reading an input data file. They have no upstream inputs and one or more downstream outputs. Transformation modules, which have multiple inputs and outputs, filter the data or change it into another form. These modules could scale or interpolate numeric data, create geometric data from numeric data, or produce image data from geometric data. Terminal modules are located at the ends of the network. They are responsible for writing the data to a file or sending it to a display.

We have developed source modules, transformation modules, and terminal modules for an existing scientific visualization program, AVS/Express, so as to facilitate the program's use in the solution of gamut mapping problems. We have produced a library of source, transformation, and terminal modules described below. These modules form the base that an analyst can build upon to solve his or her own gamut mapping or gamut visualization problem. This allows them to make use of a color transformation library without having to write a computer program and without having to live within the restrictions of a single monolithic color space visualization program. Because the task performed by each module is well defined and the code that represents each module is compact, the individual modules are easy to understand. This makes the system as a whole easier to use and maintain. It also facilitates the sharing of library modules and custom written modules amongst different analysts.

This paper begins by discussing the source, transformation, and terminal modules that form the heart of the color gamut visualization system. It then describes the use of these modules to create networks and presents a minimal color gamut visualization application. The paper concludes with two examples that demonstrate the application of the system to real world problems.

Source Modules

Source modules fall into the categories of generators and readers. Both categories share common properties. Source modules have no input ports and one output port. The output of all source modules is a data stream of tristimulus values which is passed into a visualization network from the output port. Encoded within the output data stream are the color space identification and parameters relevant to a particular color space.

Generator Source Modules

Generator source modules generate tristimulus data through some analytic method. This is different from the reader source modules described in the next section, which are source modules for reading data that might have been created through measurements or experiments.

The **RGBPointsGenerator** is an example of a generator source module that produces numeric data representing discrete RGB tristimulus values from the gamut of a monitor. The density of the data points within RGB space from this generator can be controlled interactively with standard, graphical user interface components. These controls set the number of numerical divisions which the module's algorithm uses in tristimulus production.

Reader Source Modules

Two different reader source modules, which read data from input files, have been developed. Each module incorporates interface components for file selection.

The **FilePointsReader** source module inputs tristimulus values from a file in any standard color space and directs them to the output port. The module identifies the color space represented in the file and encodes that information in the output data stream. Additional relevant information contained in the file can also be encoded. This information may include data such as an identifier for the illuminating light source or the illuminant's white point specification.

The spectral data reader source module, **SpectralFileReader**, reads color sample information from a file. The data is in the form of wavelength and percent reflectance pairs. The module supports a varying number of samples per file and a varying number of wavelength measurements per sample. The spectral information for each sample is integrated to reduce it to its corresponding tristimulus values. Integration is done using one of the standard CIE light sources, source C, as the illuminant. The output tristimulus values are in the CIE XYZ color space.

Transformation Modules

A data stream of tristimulus values produced by a source module can be passed into a transformation module to be filtered or transformed. Transformation modules have one or more input ports and one or more output ports. The transformation modules of the color gamut visualization system form the largest collection within the system. Transformation modules are categorized as color space transformation modules, AVS/Express mesh builder modules, and visualization operation modules. Libraries maintaining modules in each category are found in the system.

Color Space Transformation Modules

Tristimulus values in the color gamut visualization system can be expressed in standard color spaces such as CIE XYZ, xyY, L*a*b*, L*u*v*, and L*C*H*. Conversion between these spaces is done by connecting the input port of a color space transformation module to the output port of a source module or another color space transformation module. An analyst can display tristimulus values for a gamut in the color space of interest by using these modules.

All color space transformation modules have one input port and one output port. The data type for both ports is a stream of tristimulus values. Conversion processing between color spaces is done using functions accessed through the C API of AVS/Express from a library called the Oregon Color Software. Use of the Oregon Color Software promotes accuracy in color space conversions and supports extensions for future color spaces.

The CIE XYZ color space is the foundation of the CIE color specification system. Because of this, conversions between color spaces in the color gamut visualization system generally involve modules that transform to or from CIE XYZ space. An example is the conversion from RGB values to the L*a*b* color space. An intermediate step in this process is the conversion from RGB to XYZ. The output of the module **RGBtoXYZTransform** for this conversion is directed to the module **XYZtoLabTransform** for XYZ to L*a*b* transformation. The conversion of RGB values to CIE XYZ space requires additional information. This information includes characteristics of the original primaries of the monitor. The module for this transformation presents a graphical user interface for setting the chromaticity coordinates of the monitor's red, green, and blue phosphors and the chromaticity coordinates and luminance for the white point. These values determine what colors are shown when the phosphors are combined, thereby determining the location and size of the monitor's gamut in XYZ space. This data is encoded with the tristimulus data stream that the module produces and can be used in later color space transformations.

Mesh Builder Modules

The AVS/Express mesh data type is a compact encoding of three dimensional data. It is used by the AVS/Express program for both visualization operations and conversion to a final, displayable data format. Mesh builder modules transform an incoming tristimulus data stream to the mesh data type. This mesh data type can be passed to visualization modules, or directed to a terminal module for display. The mesh builder modules of the gamut visualization system have one or more input ports which accept tristimulus values. Each mesh builder module has two output ports. One port produces the mesh data type for use by visualization modules further in the network. The second output port produces the displayable format data stream, which can be sent directly to a terminal module if no further transformations are required. Three mesh builder modules are found in the gamut visualization system libraries.

The **ColorSurfaceMeshBuilder** module creates a mesh representing a gamut as a three dimensional solid volume. The output of this builder is used to view a gamut as a solid object. Internal components of the module accept tristimulus values in any color space and convert them to the mesh format. A step in the conversion involves determining the RGB information for each tristimulus data

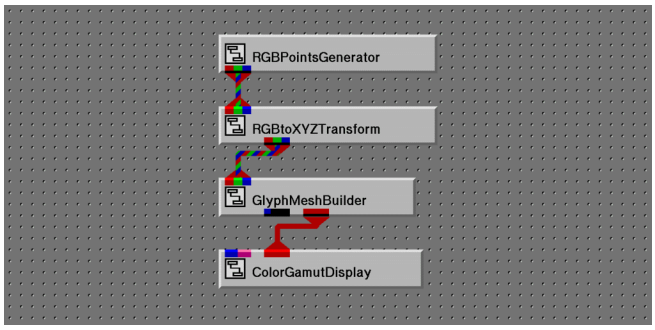

Figure 1. A simple gamut visualization network. This network was used to create Figure 2.

point so that an appropriate color value can be stored in the mesh with the point's coordinate information. This step provides the color values for the rendered surface. The **ColorSurfaceMeshBuilder** has a single, tristimulus data stream input port and the two output ports described previously.

The **GlyphMeshBuilder** module creates a mesh representing a gamut as a collection of individual points in three dimensional space. The **GlyphMeshBuilder** incorporates a user interface component to allow the type of marker that is displayed at each point to be chosen interactively. The different types of supported markers include points, arrows, crosses, diamonds, and other geometries. The input and output ports of **GlyphMeshBuilder** are identical to those of the **ColorSurfaceMeshBuilder**.

The **VectorDiffMeshBuilder** module is unique because it has two input ports for tristimulus data streams. This module is used to show the differences between two sets of discrete color data. The mesh that is produced by this module represents the change in colors between data sets with directed vectors. The head of each vector represents a point in one data set, while the tail represents the corresponding point in the other. The inputs to the **VectorDiffMeshBuilder** can be generated by any of the source modules, providing that both input data sets have the same number of tristimulus values. Output ports of the **VectorDiffMeshBuilder** are identical to the other mesh builders.

Visualization Modules

Visualization modules provide the functionality for utilizing scientific visualization techniques in the analysis of color gamuts. The mesh data output of a mesh builder module can be connected to the input port of a visualization module for data transformation or manipulation. Visualization modules have a single input port which accepts a mesh format data stream and one or two output ports. All modules in this category have an output port for passing the displayable data format to terminal modules. If appropriate to the visualization technique the module implements, it may also have a second port which outputs a mesh data stream. This mesh data stream, representing the manipulated data, can be sent to other visualization modules. In this way, multiple visualization modules can be used sequentially.

The **EdgeDetect** visualization module can be used to detect and display the boundaries of a gamut. This module is useful when combined with the glyph mesh builder module. Used in combination, the boundaries of a gamut can be made visible, while still allowing for individual tristimulus data points to be seen.

The **SlicePlane** visualization module will perform a slicing operation through a color surface gamut mesh. The slice appears as a two dimensional, color cross section of the gamut volume. User interface controls facilitate the control and manipulation of the location and orientation of the slice plane.

The **CutPlane** visualization module is used to cut away portions of a gamut volume allowing for interior sections to be viewed, while leaving the remainder of the gamut intact. The **CutPlane** module operates similarly to the slice plane module with controls to manipulate the cutting plane's location and orientation.

The library of visualization modules in the color gamut visualization system is the most dynamic. As new visualization techniques are developed for interpreting gamut data, modules that support the techniques are added to this library.

Terminal Modules

The final, display format data stream of a mesh builder module or a visualization module is directed to a terminal module. Terminal modules determine how a gamut appears when it is finally displayed by AVS/Express. The terminal module also allows a user to interact with the gamut once it has been rendered.

The **ColorGamutDisplay** module has one input ports and no output ports. The input port accepts display

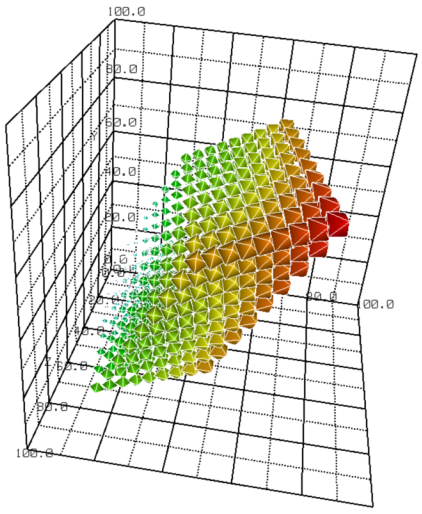

Figure 2. Monitor gamut in CIE XYZ space with tristimulus values shown as glyphs.

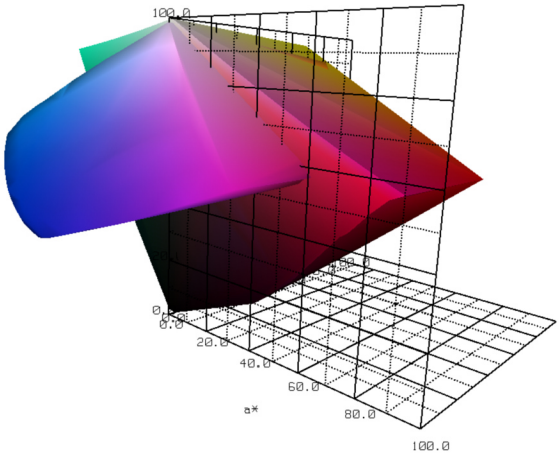

*Figure 3. Monitor gamut in L*a*b* space cut away to reveal printer gamut.*

format data from a mesh builder module or a visualization module. This port allows for multiple input connections to be made to it, providing for the display of multiple gamuts.

Many of the most powerful features of the visualization program are incorporated into the **ColorGamutDisplay** module. Several visualization operations that are common to any gamut display application are internal to the module.

Interactive gamut control functionality is provided in the **ColorGamutDisplay** module. Interaction with the displayed data is allowed in several ways. This interaction includes the ability to exhibit the numeric values for a tristimulus data point through probing operations. It also includes means for manipulating gamuts in three dimensional space.

Operations for gamut appearance manipulation are implemented in the **ColorGamutDisplay** module. Functionality for changing gamut environmental factors, such as lighting conditions and background colors, are provided. Surface properties of a gamut, such as transparency, can be controlled. Transparency is useful when displaying multiple gamuts. It allows a gamut that obscures another to be rendered transparent to permit the obscured gamut to be seen. Gamuts can be made to appear monochrome through controls of the color gamut display module. This is useful when only the size and shape of a gamut is of concern, or in multiple gamut analyses.

Additionally, the **ColorGamutDisplay** module controls secondary output of displayed data. It includes PostScript printing controls to allow for hardcopy output and for saving rendered data to files.

System Use and Functionality

The minimal color gamut visualization application includes a source module, a mesh builder module, and a terminal module. The visualization network created by these modules displays a single color gamut. Increasingly complex visualizations are created by adding more transformation modules. These additional modules are typically color space transformation modules and visualization modules.

Creating Gamut Visualization Applications

To create an application, the first step is instantiating the modules. Each of the modules is first located in the color gamut visualization system libraries. Once selected with a mouse click, it is "dragged and dropped" into the application workspace of the AVS/Express Network Editor. Next, the appropriate input and output ports of the modules are connected. Modules are connected by initially selecting the output port of one module with a mouse click operation. The connection is completed by highlighting the desired connection line. AVS/Express automatically draws connection lines from the initially selected output port to all potential input ports of all modules in the Network Editor. This feature makes it simple to identify valid connection ports.

Figure 1 shows a simple color gamut visualization application. Its principal components are a source module (**RGBPointsGenerator**) that creates the RGB tristimulus values for a monitor gamut, a transformation module (**RGBtoXYZTransform**) that converts the colors from RGB space to XYZ space, a transformation module (**GlyphMeshBuilder**) that displays a glyph at each data point, and a terminal module (**ColorGamutDisplay**) that produces the computer graphic image of the gamut. The result of creating this simple application is visible in Figure 2. The locus of diamond shaped glyphs defines the gamut of a color monitor in the CIE XYZ color space.

While it is very basic, this application displays critical features of the color gamut visualization system. It demonstrates the means for connecting modules and represents all steps in the flow of data within a gamut application: data generation, transformation, and display. The steps in the flow of data through a gamut visualization application are designed to create a simple mental model for a user. The user of the system can maintain this mental model as a basis for all gamut visualization applications that are created.

Key features of the gamut visualization system promote this mental model. The categorization of modules into source, transformation, and terminal modules, and their

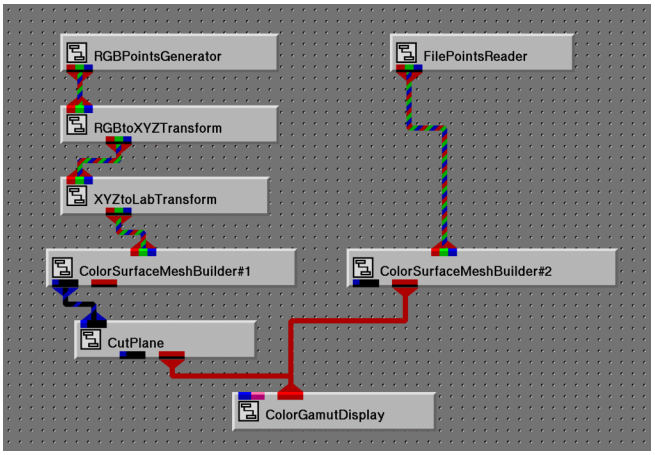

Figure 4. Gamut visualization network used to create Figure 3.

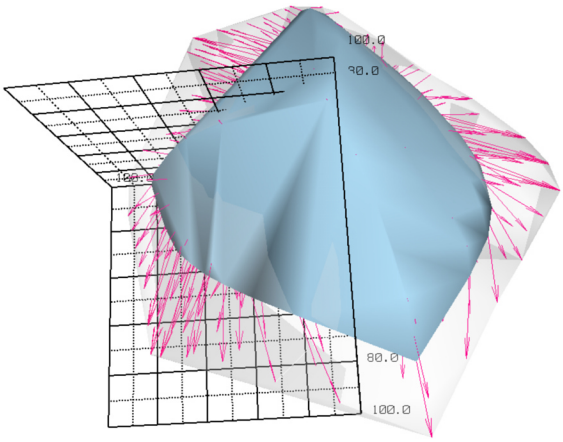

*Figure 5. Vector plot in L*a*b* space between printer gamut on plain paper (solid surface) and printer gamut on glossy paper (transparent surface).*

division into libraries within the color gamut visualization system make module location and identification straightforward. Also, the operation of connecting modules is simplified in several ways. As mentioned, when a module's port is selected for connection, AVS/Express automatically draws all possible, valid connection lines to ports on all modules currently in the Network Editor's application workspace. Additionally, the ports on modules within the color gamut visualization system have been color-coded to identify the type of connection they support. This allows for immediate identification of the appropriate location of a module in an application. Module ports which accept tristimulus data streams are identified with hatched red, green, and blue colors. Mesh data ports have hatched blue and black coloring. Ports which carry color identification and normalization data have magenta and blue hatched coloring. Input and output ports which accept displayable data are pure red. Lastly, the gamut visualization system will not allow connections to be made between ports of incompatible types. Connection lines will never be displayed by AVS/Express between ports that are incompatible.

Interacting with Displayed Data

Operations for interacting with displayed data in the gamut visualization system are intuitive. Gamuts are manipulated directly in the viewing window through mouse operations. Support is included for standard manipulation functions such as rotation, translation, and scaling, as well as more specific functions, such as object picking and probing.

Rotating, scaling, and transforming gamuts are all done with similar mouse operations. Each is accomplished by clicking the displayed object of interest with the mouse cursor and dragging it in the direction in which change is desired. Rotation is the default operation of mouse interaction in the AVS/Express data viewer window. The other two types of standard manipulation are chosen by selecting a toggle button to indicate which function is desired. An example manipulation would be to rotate a gamut around its horizontal axis. The gamut is first selected with a mouse click on its surface and then dragged in an upward or downward direction. Speed of rotation is set relative to the speed of the mouse drag movement. Scale and transformation operations behave similarly.

The interactive techniques of picking and probing are done with a combination of mouse and keyboard controls. Picking allows for individual display elements to be selected. It is a necessary operation for performing functions such as appearance manipulation. For example, to make a gamut transparent, it must first be selected through picking. Once the gamut has been picked, the object editor in AVS/Express can be used to set appropriate surface properties. Probing allows a user to display the numeric values for a tristimulus data point by selecting an area of the displayed data with a mouse click.

Practical Gamut Visualizations

A main motivation in the design and creation of the color gamut visualization system was to provide means for creating real world gamut visualizations. Practical visualizations may involve analyzing the display of multiple gamuts, or the display of the movement of colors between two sets of measurements. Means for analysis of color data in this way was previously unavailable to engineers. It is a strength of the gamut visualization program.

Multiple Gamut Display

Multiple gamuts can be displayed simultaneously with the system. Gamuts from different reproduction devices can be compared, or gamuts from the same device under different conditions can be examined. By observing the location of the surface of one gamut relative to that of another, an engineer can identify colors from one device that cannot be reproduced by the second.

Figure 3 shows the simultaneous display of the gamut of a color monitor and the gamut of a color printer in CIE

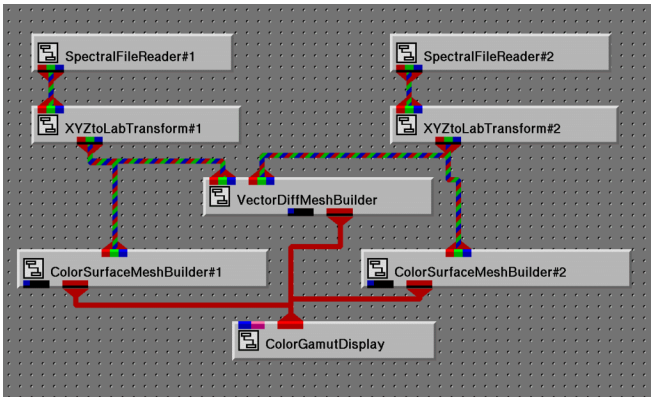

Figure 6. Gamut visualization network used to create Figure 5.

L*a*b* space. The monitor gamut has been cut away to reveal the printer gamut it encompasses. The module network used to create this visualization is seen in Figure 4. The network in Figure 1 has been expanded to include a second source module (**FilePointsReader**) to read in the measured tristimulus values for the printer. The tristimulus values created by the **RGBPointsGenerator** module are passed through the **XYZtoLabTransform** module in addition to the **RGBtoXYZTransform** module from the first example in order to convert the colors from RGB space to L*a*b* space. Displayable surfaces are constructed for the monitor and printer gamuts by **ColorSurfaceMesh Builder#1** and **ColorSurfaceMeshBuilder#2** respectively. Finally, a visualization module called **CutPlane** is used to cut open the hull of the monitor gamut to reveal the printer gamut.

No bound exists on the number of gamuts that can be included in a network and displayed at one time. Appearance manipulation techniques, combined with the use of surface mesh and glyph mesh builders, provide myriad ways to visualize multiple gamuts.

Gamut Difference Display

The ability to display the difference between two gamuts through vectors is a needed tool. Vectors represent the movement of a data point between two locations. With color data, they can show the change between colors. A color reproduction engineer might need to determine the difference in the output of a color printer with different ink sets. Another situation may require analyzing data showing a printer's output on different paper types. Alternatively, the effects of aging on the inks used in a color printer may need to be determined.

The **VectorDiffMeshBuilder** provides a solution to all of these problems. Used in combination with other mesh builder modules and visualization modules, examining differences between data sets can be an extremely powerful technique for gamut analysis. The solid volumes of two gamuts can be rendered with the difference between their corresponding data points displayed as vectors. In this way, an analyst can quickly see the size, shape, and extents of each gamut, as well as the exact path of color movement between them. A visualization of this type, and its construction, is shown in Figures 5 and 6 respectively. This visualization shows the difference between output on two paper types, plain and gloss, for a color printer.

Conclusions

We have developed a tool to help engineers evaluate the color gamut that is produced by a new color reproduction device. The resulting gamut visualization program includes source modules that generate both gamut boundaries and tristimulus values for standard color reproduction technologies such as color television monitors and color printers. Transformation modules have been written to facilitate the transformation of color data between standard color spaces such as CIE XYZ, CIE Yxy, L*a*b*, Luv, and YIQ. Terminal modules have been developed to provide convenient methods for visualizing the differences between data sets and for interacting with the displayed information. The gamut visualization program can be easily extended to include source modules for new color reproduction technologies and transformation modules for non-colorimetric data manipulations. The modularity of this gamut visualization system facilitates its use between individuals, and it could eventually form the basis for a standard way of communicating data between color analysts.

Acknowledgements

This research was supported by the Hewlett Packard Company.

References

1. G. W. Meyer and D. P. Greenberg, Color Education and Color Synthesis in Computer Graphics, *Color Research and Application*, **11**, S39-S44, (1986).
2. G. W. Meyer, L. S. Peting, and F. Rakoczi, A Color Gamut Visualization Tool, *IS&T/SID Color Imaging Conference*, 197-201, (1993).
3. C. Upson, T. Faulhaber, D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. vanDam, The Application Visualization System: A Computational Environment for Scientific Visualization, *IEEE Computer Graphics & Applications*, 30-42, (1989).