Why Color Management?

James C. King

Adobe Systems Incorporated

Abstract

It seems that everywhere you look there is some article or discussion about color management. Why all the fuss? Do I need to management my colors? We have been creating colored artifacts for a very long time and I don't think we have needed color management. So why now? Most of these discussions also refer to the ICC. What is that? These and other questions will be answered in a straightforward manner in plain English. Adobe Systems has pioneered the use of desktop computers for color work, and the author has helped Adobe pick its way down conflicting color paths with confusing road signs over the last 10 years.

Device Independent Desktop Color Management

Desktop Color Management has had a shady past. It is believed to be very difficult, hard to understand, and possibly not a good thing to attempt. I don't believe any of that and after this presentation, I hope that you will have a better, clearer understanding and like me will become a true believer.

Users are looking for color predictability and consistency. They would like a colored page to be reproducible on a variety of technologies with consistent results, even though different devices reproduce color differently. For example, when the same values are sent to two different devices (the same RGB values are sent to different displays, or the same CMYK values are sent to different CMYK printers), the resulting colors are usually quite different—unless the devices are identical and process the values identically. In order to get similar results, different values need to be sent to different devices. What's needed is a way to take the values that represent the desired color on one device, and from them produce corresponding values that reproduce the "same" color on another device. These transformations are done by using tables, equations, or other tricks. The quality of today's products is judged, in part, by how they perform these transformations.

In Figure 1 some given "source" values are shown being transformed to a modified set of values appropriate to a different "destination" device. If we were to create a transformation to map from each of N sources to each of M destinations, we would have to create M times N unique transformations as shown in Figure 2.

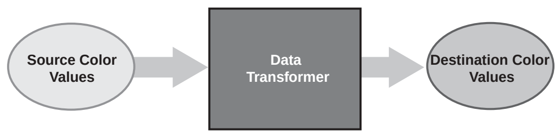

Figure 1

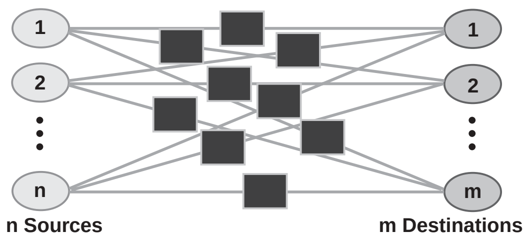

Figure 2

If we were to add a new destination device to the list we would have to add N new transforms, one for each source. Conversely, a new source would dictate that we make M new transforms, one for each destination. A common way to simplify the situation is to introduce a fixed standard color space as shown in Figure 3. One transform per source is all that is needed to relate the source to the standard and one transform per destination is all that is needed to conform to each destination. This is now an additive problem and the introduction of a new device only requires the introduction of one new transform either to or from the standard. The use of this principle is a major part of what the color management system (CMS) vendors are referring to as "device independent color[1,4]."

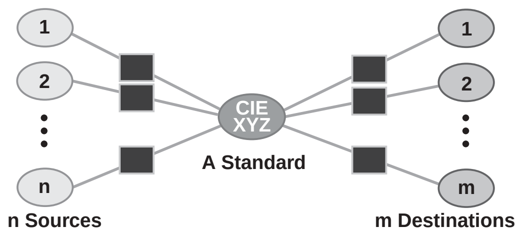

Figure 3

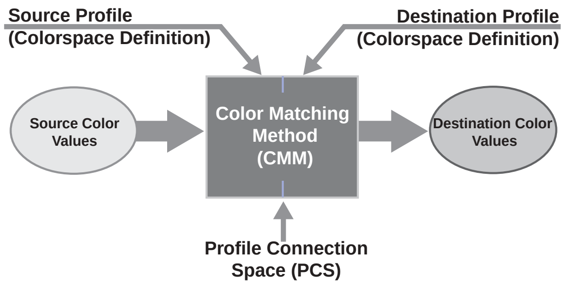
Source Profile (Colorspace Definition)
Destination Profile (Colorspace Definition)
Source Color Values
Color Matching Method (CMM)
Destination Color Values
Profile Connection Space (PCS)

Figure 4

The current transformation technology used by most of us is to have a generic transform mechanism that is customized by what we here call "colorspace definitions" as shown in Figure 4. Note that the colorspace definitions are a small amount of information compared to the large amount of data that might need to be transformed (the sources color values) such as for a large color image. The International Color Consortium (ICC) at www.color.org has standardized the "colorspace definitions" and has called them "profiles". The format chosen matches that of Apple's ColorSync®[2] Profiles and nearly all major operating system and color system vendors have agree to use that format. The transforms are implemented in what is commonly called a color matching module or color matching method (CMM). In an ICC-based color management system, the standard reference space into or out of which the color data is transformed is called a Profile Connection Space (PCS). The two PCS's in the ICC system are CIE-XYZ and CIELAB. Details of the current ICC Profile Format Specification are available on the ICC web site at www.color.org.

One other powerful idea that most of the current systems share, is an optimization that is introduced by "smashing" the two transforms into one. The definition for what has to be done is provided in terms of the PCS, gaining the device independent benefits as noted earlier. However, no sacrifice in processing efficiencies need be made. Separating the *definition* from the *execution* is key.

Example of Desktop Color Management

Figure 5 shows an example of a color managed workflow that you might find, for example, while using Adobe® Photoshop®. Note that there are "workspace color values" or working colorspace in which the current job is kept. In order to update the display, color values in the workspace need to be converted to values in the display color space. This is constantly being done as the image is updated. Conversions from the working space to the display space requires a profile for each. The monitor profile is supplied by the operating system since the display is used across all applications and is a component of the basic machine. The working colorspace is established by the user by choosing among several available within Photoshop.

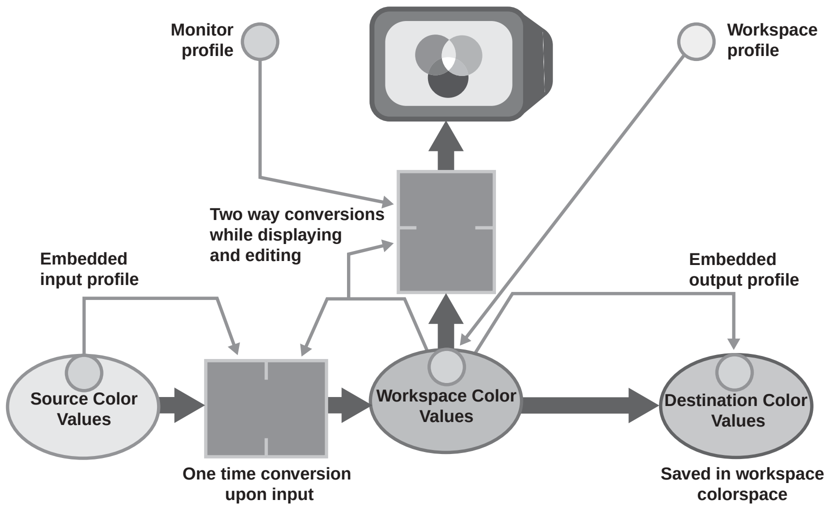

Figure 5.

If the input or source colors are in a colorspace that is other than the working colorspace then upon opening usually they will be converted into the working colorspace. This requires that the input has a colorspace profile associated with it to define the color values in the input. If the work is saved it is saved in the workspace colorspace and the profile for that space is saved with the data.

Gamut Mapping and Rendering Intents

The gamut of a colorspace is the total set of colors that can be represented within that colorspace. Typically a device CMYK colorspace will represent less saturated and fewer colors than an RGB colorspace. When the gamut of the source and destination colorspaces differ a "gamut mapping" needs to be performed. Those colors that cannot be represented in the destination need to be altered to colors that can be represented.

The ICC has defined four different general mapping categories: perceptual, saturated, and colorimetric (relative and absolute). If one is working with a company logo or any other object where it is important to preserve the colors colorimetrically, then one will want source colors to match colors in the destination (colorimetrically). For those colors that have no match some "closest possible" match should be found. This type of gamut mapping is called "colorimetric". There are two versions depending upon whether the colors are adjusted for white points or not. Usually relative colorimetric matching is done.

Experience has shown that colorimetric gamut mapping of pictures can be improved upon and so a "perceptual" method of gamut mapping is preferred for pictures. Usually these mappings make changes even to the colors that can be matched. The human visual system is more sensitive to relative color differences than it is to absolute values so changing the mapping to preserve those differences yields better looking results. When producing a business graphic or other schematic material, it may be more important to have the best solid saturated colors that a device can produce rather than to have an accurate color that the device might produce in a poorly rendered way. So for these kinds of objects one uses a "saturated" gamut mapping. There is a correspondence between gamut mapping methods and what has come to be called "rendering intents". More about rendering intents a little later. For the perceptual and saturated intents considerable latitude is allowed in the interpretation and implementation since no one has demonstrated a single best method to do the gamut mapping.

Often complex pages may have several objects each associated with a different one of the classic mapping methods. The logo requiring the best possible colorimetric mapping, the picture a perceptual mapping and the graph a saturated mapping. Within the ICC method for color management, the gamut mapping is accomplished by the color conversions being controlled by data found in the source and destination profiles. Typically gamut mapping is performed on the output side when translating from the Profile Connection Space (PCS) to the destination colorspace. Different profile data is needed for different gamut mappings. A minor dilemma arises because the choice of gamut mapping or "rendering" is determined by the source objects yet effects the output profiles. We do not want to carry output profiles with our source data since that would tie that source to only one destination.

Carrying a large variety of destinations is also not a good design. So a clever trick is used. We associate one of four "rendering intent" values with each source object. They tag the source objects as to which of the basic four gamut mappings is most appropriate for that object but do not specify anything further that might involve an output or destination device.

When this source material is finally converted to a particular destination device one of four different gamut mappings can be performed on any given object provided the output profile contains the data required to do any of four different gamut mappings. So, ICC output files do require four different sets of mapping data, one for each rendering intent. (Actually, since it is possible to obtaining one colorimetric mapping from the other, only three independent sets of data are required.)

Figure 6 is a diagram of how the rendering intent chosen by the user effects the processing done by Photoshop while outputting an image to a printer. In this case, the whole Photoshop workspace is considered to be one object and it is all rendered with a single intent chosen by the user.

Figure 7 shows how color management and the various profiles and conversions are used to do what is typically called "soft proofing". Soft proofing refers to producing an image on the monitor that closely resembles the gamut-reduced image that might be produced on a gamut limited output device (think CMYK). The color values are first converted as if they were going to be sent to the printer, and then they are converted back to the monitor colorspace.

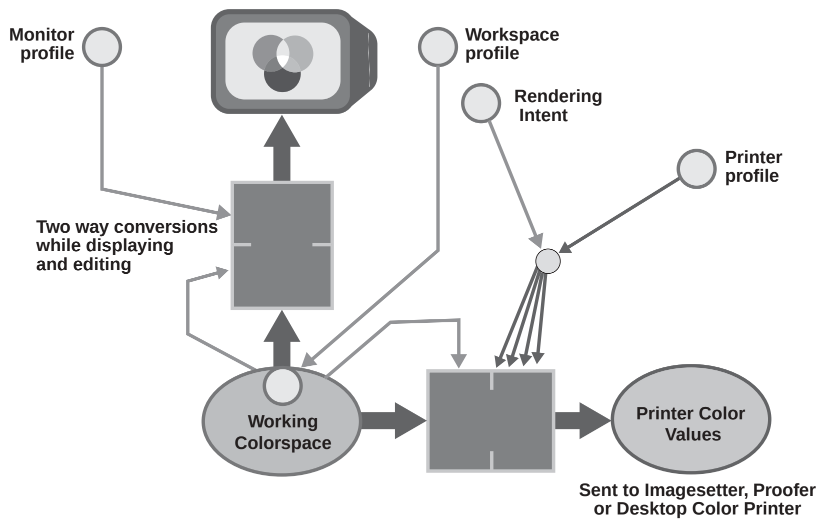
Monitor profile
Workspace profile
Rendering Intent
Printer profile
Two way conversions while displaying and editing
Working Colorspace
Printer Color Values
Sent to Imagesetter, Proofer or Desktop Color Printer

Figure 6.

During the conversion to the printer values, any important gamut mapping is performed. Then care is taken when converting those printer values to ones for the display not to do any further gamut mapping and thus preserving the reduction in gamut in the overall display.

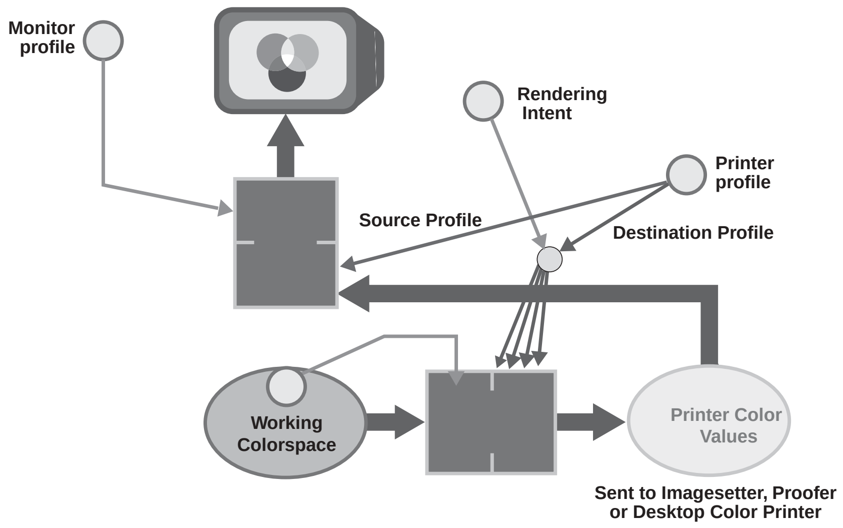

Figure 7.

Conversion Glitches

There are too many situations where the ICC workflows don't produce as good as results as one would hope for and rightfully expect. What accounts for these less-than-expected results? Here are 5 areas where things can be improved.:

1. Quantization

If someone mentions that your color data may have been quantized, then run the other direction. This term has been inherited from the world of analog to digital conversion where analog data has to be broken in to digital representations with fixed precision and range and hence "quantized". However, in color management the term refers to what computer scientists would more likely call "truncation" or "loss of data". It usually means dropping lower order (right most) digits in a result in order to allow the result to be represented in a certain number of binary digits.

It is somewhat unfortunate for color management that the computer industry standardized on the 8-bit byte as a basic computer storage and computational unit. With 8-bits one can represent or encode 256 distinct values. For many situations 256 levels or colors or variations per color channel matches the human visual system reasonably closely. However, if the coding of data into the 8-bit bytes is done without regard for the way that the values to be represented spread themselves out in a coding scheme, the 8-bit representation can become very lossy when representing color values. We would have seen far fewer bad color conversions had the computer industry standardized on 12-bit bytes or early color products had just use 16-bit values.

2. Gamut Mapping

Since the most notorious color gamut compressions involve reducing the total number of colors that can be represented then going back to the original values is usually impossible. What has been lost is lost. So it is best to delay any steps where gamut compression can occur to avoid reducing the gamut of the data we have. It must be done for output to gamut reduced devices but it is best to reduce your data in this way only in the last step.

3. White Point/Black Point

The International Color Consortium (ICC) is now a group of more than 60 companies working on and agreeing to a standard profile specification, and implicitly through that, agreeing to a standard way in which to perform color management and color conversion. The clarity of the agreements hasn't always been perfect and the specification has several places where the explanation and wording can be greatly improved. The ICC is working on this and will soon come out with a revised specification that is much improved. There is also other more long term work being done within the ICC to change some of the fundamental assumptions of the current architecture with hopes of making bigger advancements. This is also work in progress.

4. Bad Arithmetic

It is both surprising and expected that some programmers don't do a very good job of having the programs that they produce to good arithmetic. Any time when one does computation on a computer she has to worry about loss of significance and magnified error creeping into our computations. In the early days of computer development "numerical analysis" was a strong discipline where one learned in excruciating detail how to do computations that maintain the maximum amount of useful information. People writing color management software should at least pay a passing glance at this older establish discipline.

5. Interpolation Errors

Interpolation is a simple process of guessing a result value intermediate between two known values. For smooth functions this is a very effective way to reduce table sizes when functions are to be computed by simply looking up the proper answer in a table. For many functions the table would be required to have an impractical number of entries so tables with fewer entries are provided and the smoothness of the function is relied upon to make computation of intermediate results meaningful.

However, given that rounding and/or truncation are used to make the table values, and then averaging calculations are made to determine intermediate values it becomes clear that the table entries have to be represented to a higher precision than that required of the output. This is an observation that has generally been overlooked by the ICC and others and if corrected could result in slightly more precise results.

Compound Documents

Many of the pages or screen views that are created are made from many individual elements. They are compound documents. What if each of the contributing elements have been created with their own different colorspace? Figure 8 shows how this is typically handled in Photoshop. Photoshop maintains the idea of a working colorspace and as each contributing element is opened and included into the final document, it is converted to the working colorspace. Each input is expected to come with its own appropriate input profile and the profile associated with the working colorspace is used as the output or destination for the color transformation.

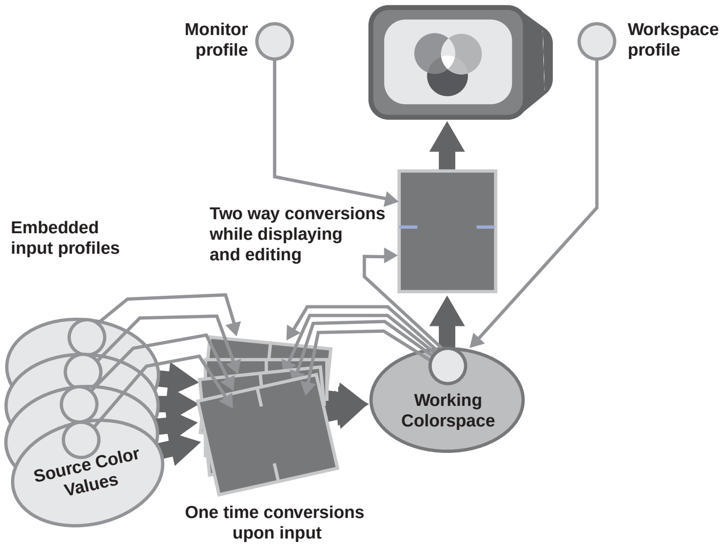

Figure 8.

In a layout application like PageMaker, a different strategy is possible. All conversions are deferred and the colorspace and associated profile are kept with each element. As the document is being displayed on the monitor many different color transformations are setup, one for each different element of the compound document. During the output of the final document all element may then be converted to a common output colorspace as shown in Figure 9. Or if the output device is a PostScript®[3] device the compound document, including all of its various input profiles can be sent on to allow the colorspace conversions to be done within the PostScript device. Notice that each element of the compound document has a rendering intent associated with it. This allows each of the elements to choose the appropriate type of gamut mapping from the output profile.

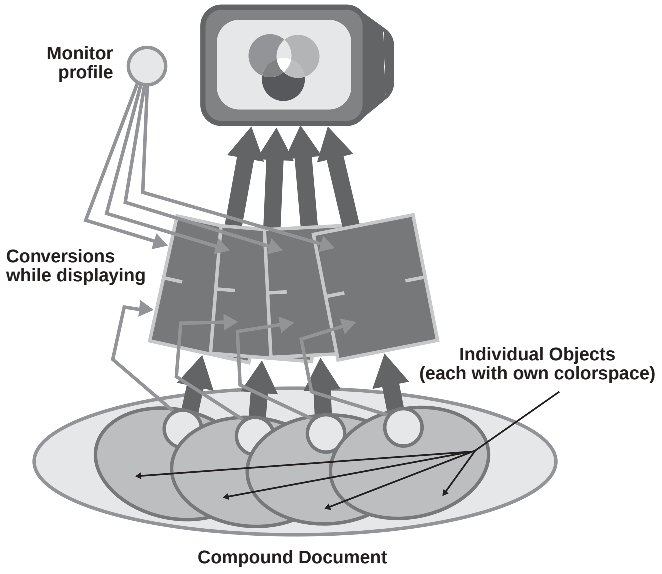
Monitor profile
Conversions while displaying
Individual Objects (each with own colorspace)
Compound Document

Figure 9.

Conclusion

At the start of this paper we tried to emphasize that color management is a requirement because devices produce color differently and require different color values to produce similar results. This part seems inevitable. However, we seem to be pushing too much of the work needed to accomplish color management onto the computer user. Let me give you an analogy. Early automobiles had a lever that the driver had to adjust that is no longer seen on automobiles today. It was called a spark advance. The exact timing of when the spark plug is to fire to cause the gasoline to explode in each engine cylinder has to be adjusted depending upon how fast the engine is running and upon how much load is being drawn from the engine. Drivers of early cars had to learn exactly how to set this spark advance as the car is accelerating, decelerating and especially when starting the engine. The fact that drivers no longer use these controls isn't that the spark does not need advancing on modern engines. It just means that engineers have figured out how to do the spark advance automatically and more accurately than the human driver can do it. We need to find which of the various things we are asking our computer users to do with respect to color management are to become spark advances. Which things can we do in our systems automatically and better than the user can?

Acknowledgements

Rob Buckley conceived of this paper and talk and urged me to do it. He also was a great help in reducing what started as an oral presentation into a written presentation. Thank you Rob.

References

1. Ron Gentile, Device independent color in PostScript, February 1993, IS&T/SPIE Int'l Symposium on Electronic Imaging, http://imaging.org.

2. Apple Computers, ColorSync. http://www.apple.com/colorsync.

3. Adobe Systems Incorporated, February 1999, PostScript Language Reference Manual, 3rd Edition, Addison-Wesley Publishing Company, 912 pages, ISBN 0-201-37922-8 (This book can be downloaded from Adobe's web site in soft copy form: http://www.adobe.com or http://partners.adobe.com/asn/developer/technotes/postscript.html.)

4. James King, On the Desktop Color System Implied by the ICC Standard, November 1998, IS&T/SID Color Imaging Conference, http://www.imaging.org.