Graphics Editing Outside of *Mathematica*

An overview of the uses of graphics editing tools for modifying *Mathematica* graphics, particularly for color enhancement in preparation for printing.

André Kuzniarek

Previous articles in this column have discussed options for handling graphics within *Mathematica* and the methods available for exporting them into various formats. In this article we will consider methods for editing and manipulating graphics using software unrelated to *Mathematica*. The most relevant formats for editing pictures outside of *Mathematica* are Illustrator 1.1 and the bitmap formats specific to your platform. This discussion will focus on the Macintosh, as it's the most prevalent system in use by publishing professionals. The main Macintosh bitmap formats are PICT and Bitmap PICT, though TIFFs are acceptable as well (just not readily exported out of the Macintosh version of *Mathematica*). Adobe Illustrator is obviously one of the applications we'll explore for editing pictures, the other is Adobe Photoshop. It's not my intention to endorse exclusively Adobe products; *Mathematica* happens to support the Illustrator PostScript format and Photoshop is one of the industry's standard desktop bitmap image editors. Some familiarity with these applications on the part of the reader is assumed.

Using Illustrator on *Mathematica* Graphics

Mathematica graphics are created in a custom version of PostScript. They can be made compatible with other PostScript interpreting systems by being filtered by an appropriate export procedure. As outlined in my last article [Kuzniarek 1995], an Adobe Illustrator 1.1 compatible conversion process is available as part of the front end for the Macintosh version of *Mathematica* (Copy and Convert Clipboard). You can open a graphic saved this way with Illustrator (version 1.1 or higher).

Illustrator is used for editing the actual composition details of an image, such as the typeface of plot axes and labels, line color and weight, scaling, and polygon colors. This format is handy for anyone who lacks familiarity with *Mathematica* (or access to it). For example, an author can provide a publisher with *Mathematica*-generated pictures in Illustrator format, which the publisher's art staff can then edit. The art staff need not be familiar with *Mathematica*, and the author need not worry about the exact details of the style required by the publisher.

André Kuzniarek is Senior Graphic Designer at Wolfram Research. He supervises the graphic design for much of Mathematica's documentation, along with a broad range of other design duties including special projects such as the design of the Mathematica font. He also does freelance design for various book publishers.

Begin the editing process by opening a file and clicking on the various elements with the pointer (mouse) to see how they are grouped. Axes and labels, boxes, and the plots themselves are the typical groupings (see Figures 1–3). A 3D plot is composed of polygons that can be ungrouped from each other in the unlikely event they need to be individually edited. In most cases, changes are made to the plot as a whole, rather than to individual polygons. Unfortunately, `Graphics3D` objects are saved out of *Mathematica* with ungrouped polygons, so they are a bit trickier to edit than `SurfaceGraphics` objects, as will be explained below.

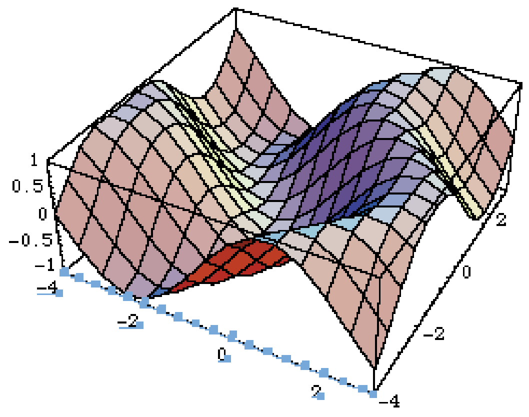


FIGURE 1. Axes and related type elements are usually grouped together in *Mathematica*-generated graphics.

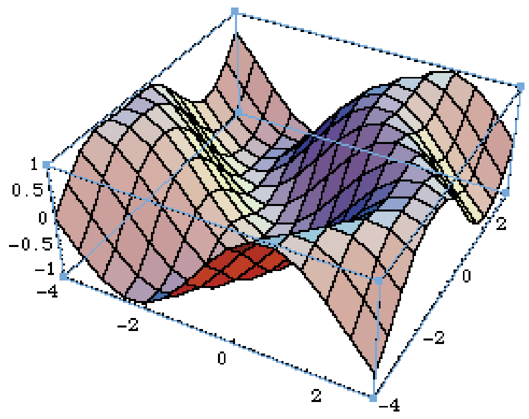


FIGURE 2. Box elements are grouped separately, in two or three units.

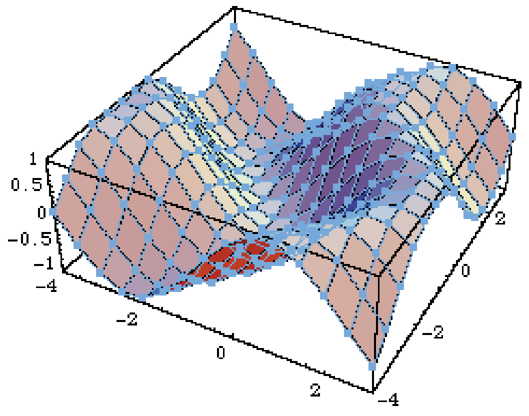


FIGURE 3. The polygons of the `SurfacePlot` are grouped as a single unit.

To modify the type, click anywhere on the baseline of a letter to select a text group. The text editing menu is easy to navigate. If you suspect that the output device that will be used to process your work does not have the fonts you're using, you can convert the type to paths, which will make them into plain graphic objects like everything else in the picture, so they won't require a matching font downloaded to a printer. However, the text might appear a bit crude at smaller sizes on low-resolution devices (such as 300 dpi printers) since the original font hints will have been sacrificed in the outline conversion procedure. (Hints are font-specific algorithms for improving the appearance of characters on low-resolution devices; they are limited to the Type 1 and TrueType font formats.) The text group need not be ungrouped to edit the content; just use the text cursor tool as in any text editing procedure.

Line weights or colors are edited in the Paint dialog box (Command-i). Two-dimensional plots, axes, and graphics primitives are obvious candidates for line adjustment. However, 3D polygon-based objects can also have the weight or color of their mesh edited in a similar way. Select the Stroke option and apply a color, or specify a line weight. There probably won't be any values showing in the dialog box options, so it may seem that a grouped unit cannot be edited, but any value you enter will indeed affect the whole group. If a graphic gets scaled up in its ultimate presentation, it's wise to lessen the line weight of the mesh to something like half a point or less (a half point can be entered as `0p0.5` or `0.5 pt`). If the lines are to stay black, you can select Overprint for black elements, which turns the default undercolor removal option in the PostScript to false (this is an extremely useful feature of Illustrator 5). The image will be processed with black sitting on top of all the other colors (instead of in place of the other colors), making color registration on a printing press much easier to handle. More explanation of this subject follows in the discussion of Photoshop and process color printing.

The procedure for mesh editing described above is simple for `SurfaceGraphics` objects, but `Graphics3D` objects consist of ungrouped polygons, and a few more steps are required to edit them as a whole. First, lock everything you don't want to change (Command-1 after making selections). Then select everything that has remained unlocked (usually just the plot

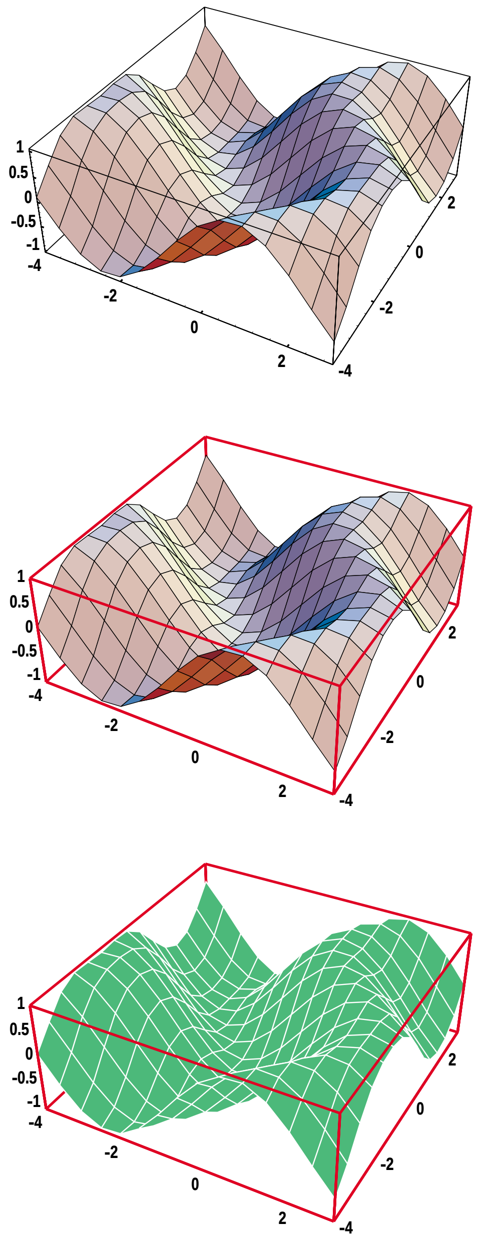


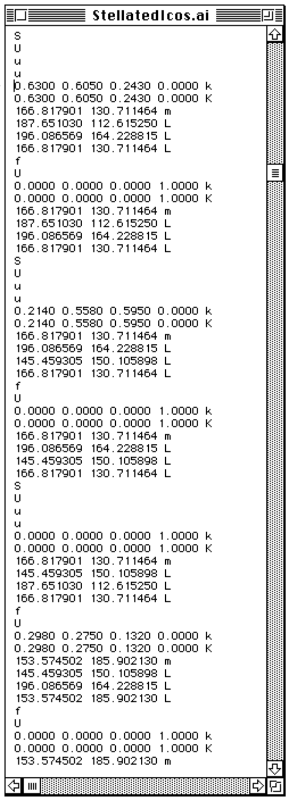
FIGURE 4. A series of changes that are simple to make with Illustrator.

itself) with Command-a and ungroup (Command-u). This releases the mesh lines from each of the individual polygons. Select any one of the mesh lines (they form L-shaped units). In the Filter menu (only available in Illustrator 5), choose Select and Stroke Color or Stroke Weight. This will search for and select all the mesh line units matching the first one selected (they should all be the same). At this point, the editing procedure described above will apply. It's useful to group the lines at this point in case they need changing later, and everything else can be unlocked again (Command-2).

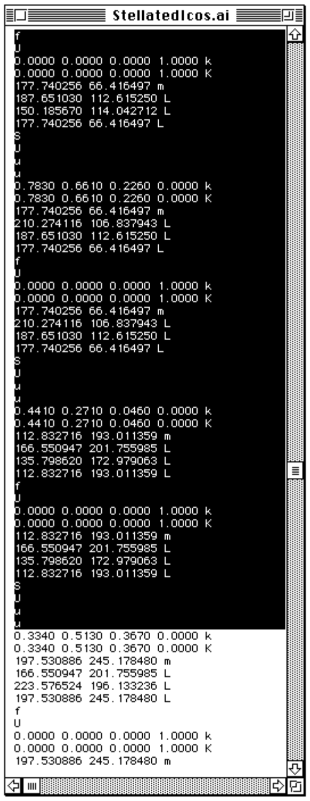
The Paint dialog box is also the place to change the polygon colors. Beware that once changed to a contiguous color, the polygons can't easily be restored to the multi-color facets of the original. It's a good idea to save the file under a different name to preserve the original before doing any editing. When saving out of Illustrator, it will ask for version types and preview options. If the intent is to reopen the picture in Photoshop for color correction, previews are unnecessary (but quite useful otherwise!).

It's important to note that in Illustrator versions 3 and below, a picture will open in Artwork mode, which looks like a skeletal line drawing (and is the only mode that allows any manipulation to be performed). To see the actual image, or the results of any changes made, select Preview in the View menu (Command-y). Artwork mode is handy in later versions when working on large graphics that take a lot of time to preview. In general, size is a factor to bear in mind, since 3D graphics can require huge amounts of memory. It's best to set the memory partition of Illustrator to the highest setting possible with it running alone on your Macintosh. Don't be surprised if a formidable looking graphic just slogs along if it only has a few megabytes available.

You may notice that 3D graphics render themselves from the background forward. It seems wasteful for all those polygons to be drawn even if they are obscured by the foreground surface of the object. In some cases, what's behind the surface might account for more than 50% of the image, which correlates directly to the memory consumed by the artwork. *Mathematica* includes a little-known `Graphics3D` option, `RenderAll -> False`, which removes the unseen background polygons. However, this option uses a rasterization process that is extremely time consuming, to the point of being impractical. Another way to reduce the size of an image file is to remove the unwanted polygons by deleting parts of the PostScript code in the Illustrator (or original *Mathematica*) file. For an Illustrator file, this trial-and-error process begins by opening the file in a text editor such as BBEdit or Word (where it's possible to select the "show all file types" option). Scroll to the beginning of the polygon list. Using the position of the scroll bar as your means of measurement, select about 40–50% of the code, leaving off at a point that matches the code at the beginning of the selection (Figure 5).


StellatedIcos.ai

```
S
U
u
u
0.6300 0.6050 0.2430 0.0000 k
0.6300 0.6050 0.2430 0.0000 K
166.817901 130.711464 m
187.651030 112.615250 L
196.086569 164.228815 L
166.817901 130.711464 L
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
166.817901 130.711464 m
187.651030 112.615250 L
196.086569 164.228815 L
166.817901 130.711464 L
S
U
u
u
0.2140 0.5580 0.5950 0.0000 k
0.2140 0.5580 0.5950 0.0000 K
166.817901 130.711464 m
196.086569 164.228815 L
145.459305 150.105898 L
166.817901 130.711464 L
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
166.817901 130.711464 m
196.086569 164.228815 L
145.459305 150.105898 L
166.817901 130.711464 L
S
U
u
u
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
166.817901 130.711464 m
145.459305 150.105898 L
187.651030 112.615250 L
166.817901 130.711464 L
f
U
0.2980 0.2750 0.1320 0.0000 k
0.2980 0.2750 0.1320 0.0000 K
153.574502 185.902130 m
145.459305 150.105898 L
196.086569 164.228815 L
153.574502 185.902130 L
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
153.574502 185.902130 m
```


StellatedIcos.ai

```
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
177.740256 66.416497 m
187.651030 112.615250 L
150.185670 114.042712 L
177.740256 66.416497 L
S
U
u
u
0.7830 0.6610 0.2260 0.0000 k
0.7830 0.6610 0.2260 0.0000 K
177.740256 66.416497 m
210.274116 106.837943 L
187.651030 112.615250 L
177.740256 66.416497 L
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
177.740256 66.416497 m
210.274116 106.837943 L
187.651030 112.615250 L
177.740256 66.416497 L
S
U
u
u
0.4410 0.2710 0.0460 0.0000 k
0.4410 0.2710 0.0460 0.0000 K
112.832716 193.011359 m
166.550947 201.755985 L
135.798620 172.979063 L
112.832716 193.011359 L
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
112.832716 193.011359 m
166.550947 201.755985 L
135.798620 172.979063 L
112.832716 193.011359 L
S
U
u
u
0.3340 0.5130 0.3670 0.0000 k
0.3340 0.5130 0.3670 0.0000 K
197.530886 245.178480 m
166.550947 201.755985 L
223.576524 196.133236 L
197.530886 245.178480 L
f
U
0.0000 0.0000 0.0000 1.0000 k
0.0000 0.0000 0.0000 1.0000 K
197.530886 245.178480 m
```

FIGURE 5. An example of polygon code in a PostScript file and a range that can be deleted. Note the starting position of the cursor (by the arrow).

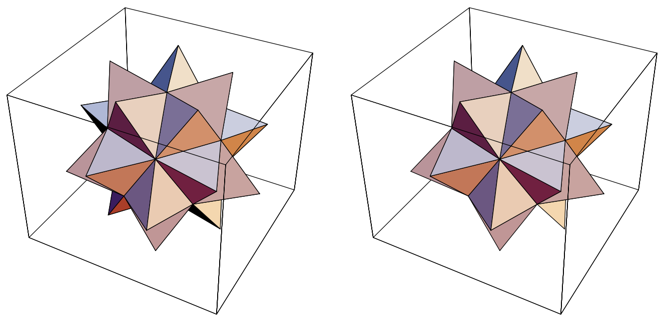


FIGURE 6. The original and edited version of a stellated icosahedron.

Delete the selected code and save the remaining code in a new file. Open this file in Illustrator and check whether any normally visible polygons have been removed. If so, repeat the editing procedure on the original file without selecting quite as much code as the first time. This process may seem cumbersome, but it can be very useful if memory conservation is critical, particularly if a graphic chokes on an older high-resolution output device (such as a Lino 100). It may not be practical if the graphic has multiple anterior layers, such as a simple torus turned a bit sideways. Figure 6 shows a simple stellated icosahedron that has been edited. A few too many polygons were removed, since four that were visible in the original are missing in the edited version, but that's easily corrected.

Another recently discovered problem lies in the Power Macintosh version of Illustrator. When opening a graphic in Preview mode, individually rendered points, such as those in a `ListPlot`, disappear (and may fail to print as well)! To correct this problem, return to Artwork mode and select a control point where one of the plot points should be. Move it to create a line, then place it back on top of the underlying point. Make sure a line width of some size is selected (which determines the dot's diameter) in the Paint dialog box and have the endpoint style set to Round. This procedure should bring the point back to life. It is not very helpful if there are a lot of points to be dealt with. In that case you'll have to open the image with a non-Power Mac version of the software until this bug is corrected.

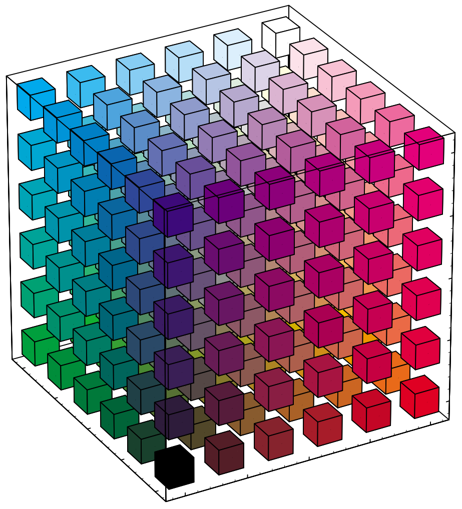
RGB to CMYK Conversion

Color correction is the process of adjusting the overall color values of a picture, particularly when switching between color models, computer platforms, or media. It's a common practice in the printing industry, where images are typically scanned from prints or transparencies and need some adjustment in the translation to digital form. The biggest issue relating to *Mathematica* graphics is the conversion from the RGB color model to CMYK. Compare the colors of a 3D plot as they appear on screen in a *Mathematica* notebook to the way they look in Preview mode opened in Illustrator. The Illustrator version will look muddier, with an overall tan or peach cast.

To explain this discrepancy, let's first look at the models. RGB stands for red-green-blue, the *additive* primary colors of light. We combine these three colors in the form of light to get white. They are the primaries used for television and computer screens, which are luminescent imagers. It's no surprise then that RGB is the default color model used by *Mathematica*. On the other hand, graphics programs like Illustrator are designed primarily for creating printed graphics. Color printing requires the use of *subtractive* primary colors: cyan, magenta, and yellow. Subtracting these three colors from an area gives white. In theory, black should result from adding the subtractive primaries together, but printing inks are far from perfect and the actual result is a muddy, brownish gray. Therefore, black is added as a fourth primary. (In CMYK, black is denoted K instead of B to avoid confusion with blue.) It's because of this fourth primary that the printing industry refers to full color printing as the four color process. Printers create four different pieces of film and plates, called "separations," for each of the CMYK primaries.

The real-world compromise of converting from RGB to CMYK is a troublesome nonlinear adventure. Reference books on color physics show three-dimensional plots of the ranges of color reproduction for the two models. These plots show that the color ranges intersect but are not identical. Blue is the most prominent victim of color translation. To view this problem directly, display the following 3D figure created by John Fultz [Fultz 1994, page 49].

```
In[1]:=  Show[
           Graphics3D[
             Table[{RGBColor[x, y, z],
               Cuboid[{x - 0.05, y - 0.05, z - 0.05},
                      {x + 0.05, y + 0.05, z + 0.05}]},
               {x, 0, 1, 0.2}, {y, 0, 1, 0.2}, {z, 0, 1, 0.2}] ],
           Axes -> False, Lighting -> False,
           ViewPoint -> {-7.8, -13.8, 8.6} ]
```

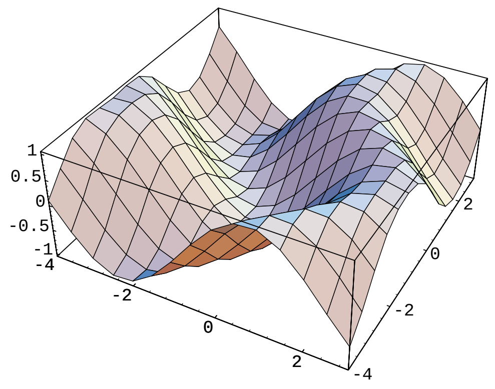


On screen you will see much brighter colors, particularly brighter and more intense blues. The printed version is a straight CMYK conversion from RGB using the algorithms present in PostScript. It is physically impossible to reproduce precisely the colors of the screen image in print. The only way to achieve a bright pure blue is to use blue ink, that is, to add a "spot" (or "nonprocess") custom color. The added printing expense of a fifth color means that this method is used only if the design is based heavily on the particular custom color.

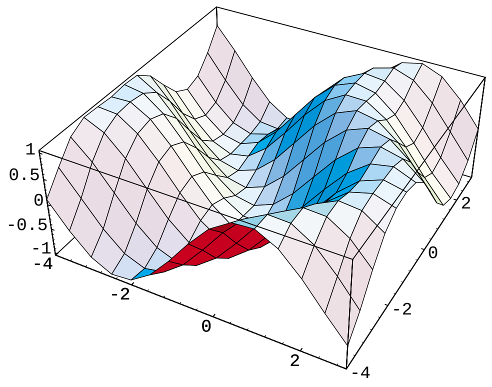
The manuals that come with Photoshop and Illustrator discuss the mechanics of color reproduction in greater detail. Color halftone screens are another issue we'll explore later.

The conversion from RGB to CMYK doesn't entirely explain the color discrepancies between graphics viewed in *Mathematica* and Illustrator. Most graphics applications on the Macintosh use methods involving color lookup tables to simulate the effects of process color printing on screen. The displayed colors are intentionally modified so the user doesn't become completely chagrined by the disparities in the printed results. *Mathematica* offers the option of using the CMYK color model to describe individual color choices or as the default for rendering entire objects. Setting the option `ColorOutput` to `CMYKColor` pre-empts the need for later CMYK conversion. The results may show improved color clarity over other RGB to CMYK conversions, but they are not necessarily any closer to what's originally seen on screen (they tend to be yellow heavy). *Mathematica* does not employ a color substitution table for simulating printed colors on screen, so the results of using the CMYK options will be startling until viewed in a printed form or in some other application that does provide substitution tables for screen rendering. Compare the following two plots on screen with the printed images here. They were exported directly into this article as Encapsulated PostScript (EPS) files. The first example was converted to CMYK through PostScript, the other was converted within *Mathematica*.

```
In[2]:=  Plot3D[Sin[x + Sin[y]], {x, -4, 3}, {y, -4, 3}]
```



```
In[3]:=  Plot3D[Sin[x + Sin[y]], {x, -4, 3}, {y, -4, 3},
           ColorOutput -> CMYKColor]
```



Color Correction in Photoshop

After CMYK conversion, the colors in a *Mathematica* graphic typically look a bit muddy since the default lighting scheme for 3D plots uses overlapped reds, greens, and blues. This problem may be alleviated with custom lighting options in *Mathematica*, or by color correction procedures using Adobe Photoshop.

Mathematica graphics in Illustrator format can be opened directly in Photoshop, or they can be opened after an intermediate step of opening and saving with Illustrator. The advantage of the latter process is in the Overprint option discussed above. Opening the original file saved out of *Mathematica* shows black in place of any of the other colors in the file, instead of on top of the colors. It's the default behavior of PostScript undercolor removal, which is a process similar to *Mathematica*'s `RenderAll -> False`. Undercolor removal keeps objects of one color from being printed behind objects of another color. Printing inks are transparent so the integrity of an object's facade would be compromised without undercolor removal. However, black isn't transparent, so it's advantageous to have it print over anything else (if it's supposed to be visible at all), and this is particularly true in the case of 3D *Mathematica* plots with a black mesh. Since color printing involves separate passes through a press for each of the primary colors, it becomes easier to hold registration if thin black lines print over the top of the other color separations.

Color separations, the primary colors extracted from a full color image, are saved in Photoshop to what are called "channels." Normally, you view a composite of all the channels showing the image in full color. Figures 7–9 show separate channels, which are simply grayscale images representing the placement of the primary inks when the art gets printed.

Photoshop allows you to open graphics in various modes, including Grayscale, RGB Color, and CMYK Color. The effect of undercolor removal is only present in graphics opened with the CMYK color model. CMYK mode is the best if the art ends up being printed, and it will emulate the printed results as much as possible using color substitution tables. RGB mode is the better choice if presentations are being made exclusively for on-screen viewing. It preserves

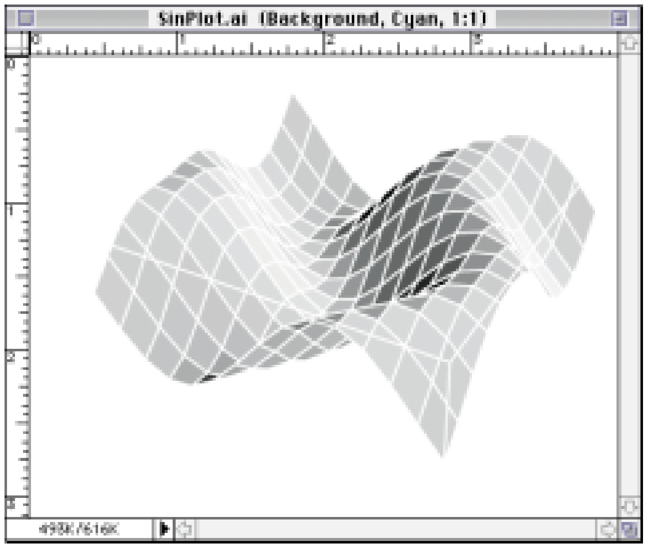


FIGURE 7. The cyan channel of a sample file saved out of *Mathematica* in Illustrator format and opened in Photoshop.

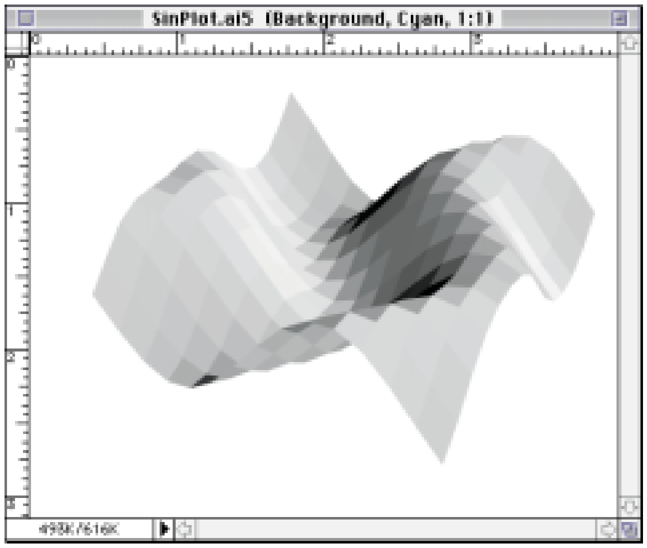


FIGURE 8. The cyan channel of the same file saved out of Illustrator 5 with overprint black enabled.

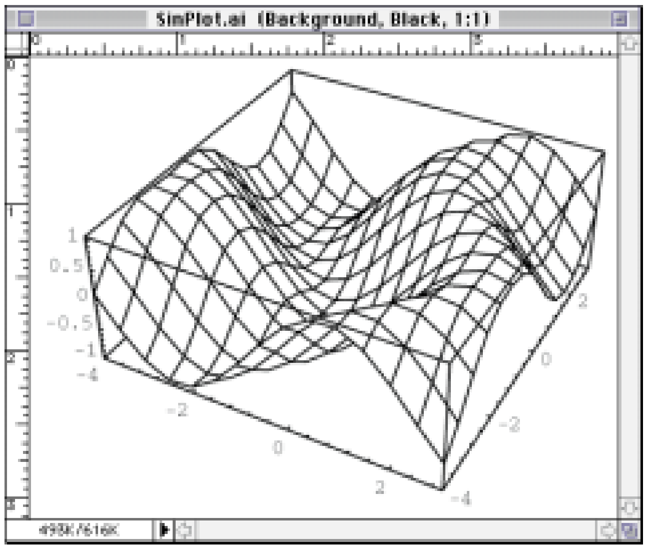


FIGURE 9. The black channel from either file.

the original colors as seen in the *Mathematica* notebook. RGB mode also has the benefit of saving memory since it only uses three channels.

When opening a graphic, Photoshop will prompt you for values for the size and resolution of the image. These values should be chosen carefully. The size is set to what the printed graphic should be, not how it appears on screen. The resolution depends on the reproduction process. The ranges of colors and grayscales in printed artwork are simulated using haftone dots. Cyan, magenta, yellow, or black are the only colors placed on the paper (unless spot colors are used). They are combined to different degrees, creating the effect of different colors, by varying the size of the tiny halftone dots. The dots are laid out on a matrix called a screen. The resolution (in pixels per inch) should be set to double the halftone screen resolution (in lines per inch) used in the printing process. The value of the halftone resolution can be obtained from the publisher's production department or from the printing establishment. The typical line resolution employed in color printing today is 150 lines per inch. This means a bitmap graphic must be rendered at a minimum of 300 pixels per inch. Going below 300 risks making the image's pixels noticeable in the printed result. Anything more than 250% of the line resolution is redundant. Adobe recommends 250% as the ideal, but the greater the number of pixels per inch, the greater the file size of the image. In my experience, 200% of line resolution has been adequate. It's important to note that Photoshop typically requires three times an image's file size in memory (or swap space) in order to do any editing, allowing for a copy-paste cache and a temporary file cache along with the original image.

Converting resolution-independent PostScript to a specific bitmap resolution creates artifacts called "aliasing," which are most prominent in angled features. You'll see distinct stair-stepping or "jaggies." Anti-aliasing is an option that uses optical effects (possible only in grayscale or color files) to smooth out rasterized lines and edges. It's usually a good idea to select this feature, which appears in the dialog box with the size and resolution settings.

Before doing any color manipulation, you should try to calibrate your monitor to represent colors shown in Photoshop's CMYK mode as they appear on paper. Monitor color adjustment, called gamma correction, operates at a system level and affects all applications equally. Photoshop comes with a control panel device called Gamma designed for this purpose. Set it by comparing an image rasterized in CMYK mode to a printed version (or a color proof). John Fultz's triple-axis color chart (above) works well for this purpose. The gamma settings you come up with are specific to the monitor you set them with. They can be saved to a file, allowing you to turn off gamma correction if you are preparing graphics in RGB mode for screen presentation.

Photoshop 3.0 offers a number of choices for editing colors. For overall color adjustment, try editing Brightness-Contrast, Curves, Color Balance, or Hue-Saturation. The manual explains the function of these options. I recommend the Curves option as it lets you save your edits to a file, which can be called up for use on similar images. Individual colors or color ranges can be searched for and replaced. There's also a mode to convert an RGB image to Indexed Color, which creates a table of all the colors in the picture. It's limited to 256 different colors, and uses an adaption algorithm if a picture starts out with more. This mode offers the possibility of editing any of the 256 colors individually through the table. It also allows some user control in preparing graphics for conversion to GIF format (popular on the World Wide Web), which is normally limited to 256 colors. The Duotone and Tritone modes are useful if printing is limited to less than four colors. Images that are to be printed with just one color should be converted to grayscale. When converting a color image to grayscale, try adjusting the hue to a greenish cast, which can improve the midtones in the conversion process.

Individual polygons can be revised using the Magic Wand tool, which limits color selection to a specific polygon or to the image area inside the black mesh lines. This is a time consuming process but guarantees more exacting results than those possible with overall correction methods. When you choose an alternative color, the Photoshop color picker warns you if the color is outside the printable color range. Watch for the exclamation mark as you move through a color saturation selection field. Colors can also be edited on each of the separation channels. It's possible to remove all the yellow in a CYMK image, for example, without touching any of the other primaries.

The color correction procedures described above might become frustrating with stubborn graphics that just won't show improvements in a desired color range. The PostScript rasterizer built into Photoshop or Illustrator might be to blame. Improved results can often be obtained by avoiding PostScript altogether. Do this by saving a graphic out of *Mathematica* as PICT format. PICT format offers some scalability without sacrificing image clarity, but my advice is to scale your images in *Mathematica* to the necessary size before exporting. Consider the line screen that will be used in printing the final image, and note that a Mac screen is fixed at 72 pixels per inch (ppi). PICT is the default Mac screen display format, so PICT files will always be 72 ppi. You need to scale your picture up to a point where the number of pixels is double the necessary line screen resolution when the image is rendered at the desired print size. To determine the enlargement, divide the desired resolution by 72 and multiply the result by the original image size (its width or height). For example, when printing an image originally four inches wide, but with a 150 line screen, the image should be scaled to $4(300/72) = 16.666$ inches wide. After scaling the image to 16.666 inches in *Mathematica*, copy it and convert the clipboard to PICT. Open the file in Photoshop and scale the image back down to the desired four inch size, but enable the Preserve File Size option and watch the resolution jump up to 300 ppi. Convert the image to CMYK mode and see if the results are any better than what resulted from the Illustrator version. In many cases the colors will be brighter and truer to the RGB version.

Color Hardcopy Output Options

There are a number of color hardcopy imaging options available today to computer users when not too long ago there were almost none. The oldest color output technology is probably inkjet. Hewlett-Packard is the leader in providing inexpensive color inkjet printers in the 300 dpi range, but Epson has recently brought to market a highly praised 720 dpi inkjet printer. On the high end, Iris inkjet output, which has been available for quite a while, provides near-photographic results because of its extremely high resolution. It's about the most expensive option available, but is also becoming obsolete. Inkjet printing offers bright colors, but with tiny splattering artifacts that can affect clarity. Inkjet printouts are also fairly delicate and susceptible to smearing.

Color laser printers are just starting to gain acceptance, but they tend to create dark colors. They offer sharp imaging as well as maintenance conveniences. Canon Laser Copy (CLC) systems are a kind of toner-based laser printing technology, but more refined than office-grade color laser printers. CLC systems are available through service bureaus and copy shops, and provide excellent output when working from images on disks. They still tend towards darker output, but they offer the best quality for the price, especially when printing multiple copies.

Dye sublimation printing produces results that most closely emulate the image seen on screen. It's a moderately expensive option, but less costly than high-resolution inkjet printing, with results that might be considered equal or better. Dye sublimation printing is sometimes used in the printing industry as a substitute for film-based color proofs. Its imaging capabilities are flexible enough for accurate calibration for matching printed results, but it doesn't show the image as it will exist on the final film, and might be considered risky as a printer's proof. It's quite possible for a graphic to print to this proofing medium without error, but then fail in the film stage because of a missing font or some other complication related to the film output device.

Matchprint (or similar) proofs from film are the most accurate proofing method for offset printing. These proofs are made from the film that will be used to expose the printing plates, so what you see is what you'll get (except for some variation in color intensity that can be controlled on press). Film is processed through a PostScript-based raster image processor (RIP) to create an extremely high resolution bitmap. Beware that some older film output systems might have trouble with complex *Mathematica* graphics because of the many PostScript control points. Film imagers are capable of different resolutions, typically 1200, 2400, and 3200 dpi. Higher resolutions offer greater color range and help alleviate color banding problems when high-resolution line screens are required. Scitex systems used to be the standard prepress devices for processing computerized graphics into film. PostScript clone RIPs make it possible for them to handle *Mathematica* graphics, but also incur a greater possibility of processing error. True PostScript interpreters are the best choice for film imaging of *Mathematica* graphics.

Page Layout Tools

Mathematica users can choose from many page design tools for incorporating graphics into layouts with text. The notebook front end offers a good online presentation paradigm, but it's a bit lacking in printing and publication formatting options. TeX might be familiar to *Mathematica* users because of its math formatting capabilities, but user friendly tools like FrameMaker, Quark Xpress, and PageMaker are worth considering.

The learning curve for TeX is very high (although Blue Sky's TeXtures alleviates that a bit). FrameMaker is next in line, being a rich, option-laden application that takes a while to learn. Quark is easier, but only if you discard their highly confusing documentation and use a third-party reference. PageMaker, the desktop page layout tool that defined the desktop publishing market, is the most intuitive tool of the bunch. It used to be the least flexible, but has recently been upgraded to a level of sophistication that almost rivals Quark.

Why is Quark the most popular graphics production tool? Mainly because of its object placement precision, its flexibility in image handling, and its sophisticated color production options. Printing establishments consider it the software of choice (along with Illustrator and Photoshop). FrameMaker is considerably less sophisticated about these issues, and TeX almost completely ignores them. In fact, the only graphic format supported by TeX is EPS. If you want to use TeX to handle bitmap images, you'll need a TIFF to EPS converter, or any image processing application that can handle the conversion of a specific bitmap format.

FrameMaker offers index, cross reference, bibliographic, and notation tools that rival TeX's. These sort of authoring tools are not as handy or well implemented in Quark and PageMaker. FrameMaker also offers excellent platform portability, since it's available on Windows, Macintosh, NeXT, and X Windows, and provides its own interchange format.

Of all these tools, TeX is the least expensive, since it's mostly in the public domain. However, specialized manpower is likely to be required in its use so it doesn't usually end up being free. All the other applications tend to be priced similarly in the $500 range, so it's wise to compare prices through various dealers.

References

Fultz, John. 1994. Using color in graphics. *The Mathematica Journal* 4(4): 48–53.

Kuzniarek, André. 1995. Putting pictures elsewhere. *The Mathematica Journal* 5(1): 62–66.

André Kuzniarek
Wolfram Research, Inc., 100 Trade Center Drive, Champaign, IL 61820-7237
`andre@wri.com`

The electronic supplement contains the notebook `Comparing Color Models`.