Subject: Re: color value interpolation from colorbar
Posted by Peter Mason on Fri, 05 Dec 2008 04:31:41 GMT
View Forum Message <> Reply to Message

j.coenia@gmail.com wrote:
> Thanks everyone. Sorry for the delay, I had to generate the requested
> pics and create a Picasa web album. If there is a better way to post
> images to usenet forums, please let me know.
>
> Below is the Picasa link to the two images requested by Vince and
> Paolo, (1) an example frame grab of a scan, and (2) its colorbar RGB
> plotted against colorbar location:
>
> http://picasaweb.google.com/j.coenia/ColorInterpolation?auth key=H9iPrIqxX1c#
>
> To answer Jeremy's question, the colorbar length is 140 pixels or so
> (scaled here from 1 to 100 on the x axis, which is vertical in the
> scan). You can see from the plot that the colorbar sampling is
> "garbagy." There are two very bright artifacts, at approximately x=20
> and x=80. Such outliers can be tossed or smoothed out somehow I
> think. For simplicity, I just sampled the values down the vertical
> center of the colorbar, as the colorbar tends to bleed a little into
> the dark background near the edges (more errors).
>
> Jeremy's answer makes some sense to me. So is it possible to
> reasonably guess the color levels in that artery using the colorbar on
> the side of the scan? I know there is no scale on the colorbar --
> I've been instructed to assume linear gradient from 1 to 100.
> Radiologists and researchers use these colors; can the computer
> quantify them to extract more meaningful information?

Whoever made these images in the first place must have had a system, surely?
But who knows what might have happened in between printing and digitising.
For the plots, did you take a single column of colourbar pixels or did you
average across some of its width?   (Averaging, median-filtering or
something like that would clean up some of the noise.)
So anyway the objective is to find a function that maps image_colour to
data_value.   Judging from the plot, blue is sort-of linear while red and
green look like the're through some gamma.   Along the lines of
R_now=R_orig^0.9 and G_now=G_orig^1.1 (for example).   Yuk.
A brute-force solution is to use SVD, for example, to fit a polynomial to
this lot and map it to your data values.   Divide your 140-element
colourbar-derived R, G, B arrays by 255.0 and try something like this.
(This has x^2 and x^3 terms for R and G to try to map some curvature.
Maybe you don't need the x^3 terms.   Maybe you need more.   Maybe you need
a higher term for blue too.   I haven't tried this out.)
```
   A=fltarr(8,140)
   A[0,*]=1
```

```
A[1,*]=R & A[2,*]=G & A[3,*]=B
A[4,*]=R*R & A[5,*]=G*G
A[6,*]=R*R*R & A[7,*]=G*G*G
SVDC, A, W, U, V
CF=SVSOL( U, W, V, (findgen(140)+1.0)/1.4 )
```
(I might have columns and rows mixed up here. I'm only human.)
CF will have coefficients that you can use to get a data value given some image pixel's RGB. Get the image pixel's R,G,B, divide by 255, and calculate image_data_value=CF[0] + CF[1]*R + CF[2]*G + CF[3]*B + CF[4]*R*R + CF[5]*G*G + CF[6]*R*R*R + CF[7]*G*G*G.

RGB isn't a great colour space for colour matching work and you might have better luck converting to some form of HSV (hue-saturation-intensity) for this modelling. You can use IDL's COLOR_CONVERT for this. (Watch out for wrapping hues. Shouldn't be a problem for red-to-yellow hues as you can clip any hue above 330, say, to 0.) You might get away with just using intensity and saturation in this dataset (with a stronger weight on intensity), or perhaps intensity alone, as long as you know that you will only query the data values of the coloured bits of imagery (which is an obvious constraint whatever you do). If you can work with intensity alone then it'll make the modelling much simpler.

I hope this helps
Cheers
Peter Mason