
Subject: Re: Converting 8-bit image + pallete to 24 bit image with alpha channel

Posted by [davidf](#) on Thu, 09 Mar 2000 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Ricardo Fonseca (zamb@physics.ucla.edu) writes:

```
> I'm trying to convert an 8-bit grayscale image to a 24 bit image with an
> alpha channel, mapping the colors of the 8-bit image through a palette.
> Right now I'm doing it like this:
>
> CT = 23
> LoadCT,CT
> TVLCT, rr, gg, bb, /get
>
> Data = ByteScl(Abs(Data), MAX = max, MIN = min)
> s = Size(Data)
> AlphaImage = ByteArr(4, s[1], s[2])
> AlphaImage[0,*] = rr(Data[*])
> AlphaImage[1,*] = gg(Data[*])
> AlphaImage[2,*] = bb(Data[*])
> AlphaImage[3,*] = 128
>
> Which works, but I was wondering if there was a more efficient way of doing
> this. Can anyone help?
```

That's about the size of it, I'm afraid. :-(

Cheers,

David

--

David Fanning, Ph.D.

Fanning Software Consulting

Phone: 970-221-0438 E-Mail: davidf@dfanning.com

Coyote's Guide to IDL Programming: <http://www.dfanning.com/>

Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: Converting 8-bit image + pallete to 24 bit image with alpha channel

Posted by [Struan Gray](#) on Fri, 10 Mar 2000 08:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

Ricardo Fonseca, zamb@physics.ucla.edu writes:

```
> AlphaImage = ByteArr(4, s[1], s[2])
> AlphaImage[0,*] = rr(Data[*])
> AlphaImage[1,*] = gg(Data[*])
```

```
> AlphaImage[2,*,*] = bb(Data[*,*])  
> AlphaImage[3,*,*] = 128
```

It is often faster to construct the array directly than to construct an empty array and fill the planes:

```
TVLCT, rr, gg, bb, /get  
data = bytscl(dist(200))  
s = Size(Data)  
  
alphachannel = make_array(size=s, value=128b)  
AlphaImage2 = [rr(Data), gg(Data), bb(Data), alphachannel]  
alphaimage2 = reform(alphaimage2, s[1], 4, s[2], /overwrite)  
alphaimage2 = transpose(alphaimage2, [1,0,2])
```

It is not as easy to see what is going on here, but on my machines it is three to four times faster.

The last transpose step is necessary because of the way IDL orders array elements in memory (as is the order of the dimensions in the reform line). With 3-channel images you can usually avoid the transpose step if you correctly use the TRUE or INTERLEAVE keyword/properties of plotting routines or image objects. I haven't used alpha channels much, and the help files are opaque, so I don't know if you can set an interleave for a four channel image - if you can, the transpose step can be omitted here too.

Struan
