Subject: 24-bit image planes (was Colored MPEGs)
Posted by Struan Gray on Wed, 11 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

David Fanning, davidf@dfanning.com writes:

```
>    FOR j=0,frames-1 DO BEGIN
>       image24[0,*,*] = r(data[*,*,j])
>       image24[1,*,*] = g(data[*,*,j])
>       image24[2,*,*] = b(data[*,*,j])
>       MPEG_Put, mpegID, Image=image24, Frame=j
>    ENDFOR
```

You probably know this already, and the posted code is what I call pedagoptimal, but here's a handy speedup I like to use when creating 24-bit images from 8-bit ones. Notice that when you do an assignment:

```
image24[i,*,*] = 8_bit_image
```

you are using the least efficient way of accessing memory in IDL, since you are keeping the first index fixed and stuffing array data into the leftmost ones. You can define your image as a BYTEARR(xsize, ysize, 3) to speed things up but I find that the fastest way to do things is often to simply create the array from scratch and reform:

```
dataslice = data[*,*,j]
image24 = [r(dataslice), g(dataslice), b(dataslice)]
image24 = reform(image24, xsize, 3, ysize, /overwrite)
```

Creating the dataslice avoids the need to re-access the subarray each time, giving me a factor of two speedup. Building the image and reforming it gives a tenfold speedup on top of that. It doesn't make a huge difference with a single small image, but with bigger images or large numbers of movie frames it adds up.

If you are going to use the image with TV or other routines that accept pixel/row/plane interleaved 24-bit images you can display the image directly:

```
TV, image24, true=2
```

If not, as in the case here (MPEG_Put only accepts 3xnxm images), you have to use TRANSPOSE to swap the indexing:

```
image24 = transpose(temporary(image24), [1,0,2])
```

This has a small time penalty, but nothing like that incurred by stuffing images directly into the image planes.