Subject: Re: Maximum value array resampling
Posted by Richard French on Fri, 05 Aug 2005 23:57:55 GMT
View Forum Message <> Reply to Message

```
>
>  For arbitrary images with both dimensions even:
>
>  d=size(x,/DIMENSIONS) & nx=d[0]/2 & ny=d[1]/2
>   y=transpose(max(reform(transpose(reform(x,2,nx,2*ny),[0,2,1] ), $
>                        4,ny,nx),DIMENSION=1))
>
```

> How does it work? It juggles dimensions so that the indices of all the
> 2x2 sub-arrays are next to each other in memory, and then uses
> max(/DIMENSION) to collapse over them. The inner call to REFORM puts
> them adjacent to each other, but in the wrong dimension, then TRANSPOSE
> makes them adjacent in the fast-changing dimension. The rest is
> straightforward.
>
> There may be a quicker way with only one call to TRANSPOSE, but I
> couldn't find it (anyone?). Also, if you don't care to keep X, throw a
> couple of /OVERWRITE keywords for both REFORM statements, to save some
> memory and time.
>
> JD
>

On a time test, I found that this clever approach takes 18 sec for a 5000 x
4000 image, compared to the inelegant routine I posted that takes 4 seconds
to compute the indices of each element and then 7 seconds for the resampling
of each image. For lots of images, the speed gain is about 2.5x compared to
this routine. Now we just need to find a way to speed up the clever routine
and we'll be all set! I had not known about the /DIMENSION keyword to the
MAX() function - thanks for the lead on that! Are you sure there isn't a way
to use HISTOGRAM to do this?

Dick