Subject: Re: Avoiding FOR loops (version googleplex.infinity)
Posted by Tom McGlynn on Thu, 10 Apr 2008 18:49:22 GMT
View Forum Message <> Reply to Message

On Apr 10, 2:13 am, Gaurav <selfishgau...@gmail.com> wrote:
... Now it is up to you guys to decide if it
> is exactly the same as Tom's or there is some difference and as to
> which would be faster.

Nah... You're the one with the problem. Why should we do your work for you! (: Seriously, it's up to you to decide which you prefer for whatever reason. The code I wrote doesn't create as many temporary arrays, so I might expect it to be a bit faster, but if the number of matches is small then yours accesses your output array less. Perhaps that balances things. Measure it and see. I do believe that the two give different answers on the edges of the image, but which (if either) is correct depends upon details of the specification of the problem.

In any case it's a little ironic that your code illustrates one of the common IDL bugs that I mentioned in my post on the concurrent thread on that topic in this group. Can you find it?

Good luck,
Tom McGlynn

> ...Here follows my code:
>
> ##################### Code begin ###################################
> dims = size(img,/dimensions) ; getting the dimensions of input image
> output_img = bytarr(dims(0), dims(1))
>
> kernelSize = 5 ;define the kernel size (here, 5)
> padSize = FLOOR(kernelSize /2.0) ;calculate the padding to generate
> around the original image
> paddedImg = bytarr(dims(0)+2*padSize, dims(1)+2*padSize) ;initialize
> the padded image
> paddedImg(padSize, padSize) = img ;define the padded image
> tempImg = bytarr(dims(0)+2*padSize, dims(1)+2*padSize) ;will contain
> the padded, final output
>
> for x = -padSize, padSize do begin
> for y = -padSize, padSize do begin
> if x eq 0 and y eq 0 then continue ;we do not want to compare with
> the element itself
> ;the x, y loops shift the whole image array around the kernel and
> compares the shifted image with the original
> indices = WHERE(paddedImg eq SHIFT(paddedImg, x, y)) ;

> tempImg(indices) = finalImg(indices) + 1 ;increment '1' wherever an
> eqality is found
> endfor
> endfor
> output_img = EXTRAC(tempImg, padSize, padSize, dims(0),
> dims(1)) ;extract the output image from the padded output image.
> ################### end of code/algorithm ######################
>
> In my case it works wonderfully well and speeds things up by a factor
> of a few hundred times. Any further optimization would be highly
> appreciated.
>
> I guess, I ought to go for a manicure for my gnawed nails now.
>
> Cheers,
> Gaurav
