
Subject: Re: Need hint to use FOR loops using 3-D arrays

Posted by [airy.jiang](#) on Tue, 12 Jun 2007 10:43:16 GMT

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

On 6 11 , 11 14 , alexcardesin-...@yahoo.com wrote:

> Hi,

>

> I am having a lot of trouble trying to improve a routine that I got
> from a colleague. The goal is to process a 3-D cube removing spikes,
> but it uses three nested FOR loops which makes it amazingly slow. I
> know I should try to remove the single-pix functions and try use the
> array functionalities but I do not manage to do it using a matrix.

>

> Can someone give me a hint?

>

> This is the piece of code:

> -----

> ---

> ; cube is a tridimensional matrix (256x432x256)

> size_elem=size(cube)

> ns=size_elem[1]

> nb=size_elem[2]

> nl=size_elem[3]

> level=2. ; this is a hardcoded value

>

> tot_pixels=double(ns*nb*nl)

> c1d=0

> c1n=0

>

> ; Evil LOOP

> FOR l=0,nl-1 do begin

> frame=CUBE[:,*,l)

> neighbor=fltarr(9)

> for s=0,ns-1 do begin

> for b=0,nb-1 do begin

> neighbor[0]=frame[s,b]

> neighbor[1]=frame[0 > (s-1), b]

> neighbor[2]=frame[(ns-1) < (s+1), b]

> neighbor[3]=frame[s , 0 > (b-1)]

> neighbor[4]=frame[0 > [s-1], 0 > (b-1)]

> neighbor[5]=frame[(ns-1) < (s+1), 0 > (b-1)] ; I should

> do this with a matrix, right???

> neighbor[6]=frame[s ,(nb-1) < (b+1)]

> neighbor[7]=frame[0 > (s-1),(nb-1) < (b+1)]

> neighbor[8]=frame[(ns-1) < (s+1),(nb-1) < (b+1)]

> neighbor_sort=neighbor[sort(neighbor)]

> stdev=(neighbor_sort[7]-neighbor_sort[1])/2.

> if (frame[s,b] gt neighbor_sort[4]+level*stdev) or \$

```
>      (frame[s,b] lt neighbor_sort[4]-level*stdev) then begin
>      cube[s,b,l]=neighbor_sort[4]
>      c1d=c1d+1.
>    endif
>  endfor
> endfor
>
> -----
> ---
> thanks, alex
```

Use less array operation, and less nesting. And i agree with Paolo that using the Shift function. That's my suggestion. I hope it's useful.
