Subject: Re: avoiding for loops
Posted by steinhh on Thu, 12 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

In article <364a6d1a.35772978@news1.alterdial.uu.net>
lbryanNOSPAM@arete-az.com (Lisa Bryan) writes:

> I've had such good luck recently with y'all that I thought I'd send
> another one. Although this is a little less esoteric. I need to
> apply a median filter all the way to the edges of an image. So I came
> up with putting an edge around it the width of the filter then
> selecting out the original region. Sort of what convol does
> automatically.
>
> The following code is what I came up with. It works, but it is clunky
> as heck. I know some of you guys are wizzes at getting rid of loops.
> Care to lend a hand?
>
> p.s. the data I'm working on array 4000x512 arrays not 100x100!

Ok. Though posting code on the net may expose bugs!

Your lines

```
>      temp(i,azrange-1+n:azrange-1+2*n) = temp(i,azrange +n - 2)
```

and

```
>      temp(shotrange-1+n:shotrange-1+2*n,i) = temp(shotrange +n - 2,i)
```

do have some flaws.

The "temp(n,n) = bottshade" statement has initialized all values
inside temp(n:shotrange+n-1,n:azrange+n-1), so here you're
overwriting one row/column too much. That in itself is ok, but
that probably led you to take the value from the wrong column/row
(e.g. azrange+n-2 instead of azrange+n-1). If I'm not completely
lost, lines should read

```
     temp(i,azrange+n:azrange-1+2*n) = temp(i,azrange +n - 1)
and  temp(shotrange+n:shotrange-1+2*n,i) = temp(shotrange +n - 1,i)
```

But now, for a version without loops. This is indeed my favourite
loop killing mechanism:

```
  [.....]
  temp = fltarr(shotrange +2*n,azrange + 2*n)
  temp(n,n) = bottshade

  ;; Expand and copy first/last column of *bottshade* into place

  temp(0,n) = rebin(bottshade(0,*),n,azrange)
  temp(shotrange+n,n) = rebin(bottshade(shotrange-1,*),n,azrange)

  ;; Expand and copy bottom/top row of *temp* into place

  temp(0,0) = rebin(temp(*,n),shotrange+2*n,n)
  temp(0,azrange+n) = rebin(temp(*,azrange+n-1),shotrange+2*n,n)

  ;; Done!
```

I haven't timed it, but it should save you a lot of time. There
are other ways of doing this (matrix multiplication) but the
principle is the same: "Fan out" a column or row so it becomes a
2-d array that only needs to be patched in at the right place.

Personally I don't like the matrix multiplication method, but
that's mostly for esthetic reasons. Although I do remember we did
some timing tests on stuff like this ages ago, I don't remember
which one was the fastest - I would guess it's architecture
dependent as well. The difference wasn't all that big, though.

Regards,

Stein Vidar