Subject: Best way to move a window
Posted by Ned Horning on Thu, 28 Feb 2002 02:03:54 GMT
View Forum Message <> Reply to Message

I am writing a progarm that move a 3 x 3 window through an image. From what I can tell IDL isn't well suited for the way I have programmed it but I can't figure out how to improve it. I expect I'm missing a fundamental concept.

The way I currently have it writen is to move the window, one pixel at a time and process it like this:

```
FOR y=1, num_tile_lines - 2 DO BEGIN
   out_tile[y*num_tile_samples] = 0
   FOR x=1, num_tile_samples - 2 DO BEGIN
     IF (((tile_data[x,y] gt 179.99999) AND (tile_data[x,y] lt
180.0001)) OR $
        ((tile_data[x+1,y] gt 179.99999) AND (tile_data[x+1,y] lt
180.0001)) OR $
        ((tile_data[x-1,y] gt 179.99999) AND (tile_data[x-1,y] lt
180.0001)) OR $
        ((tile_data[x,y+1] gt 179.99999) AND (tile_data[x,y+1] lt
180.0001)) OR $
        ((tile_data[x,y-1] gt 179.99999) AND (tile_data[x,y-1] lt
180.0001)) OR $
        ((tile_data[x+1,y+1] gt 179.99999) AND (tile_data[x+1,y+1] lt
180.0001)) OR $
        ((tile_data[x-1,y-1] gt 179.99999) AND (tile_data[x-1,y-1] lt
180.0001)) OR $
        ((tile_data[x-1,y+1] gt 179.99999) AND (tile_data[x-1,y+1] lt
180.0001)) OR $
        ((tile_data[x+1,y-1] gt 179.99999) AND (tile_data[x+1,y-1] lt
180.0001))) THEN $
        out_tile[y*num_tile_samples+x] = 0 ELSE BEGIN
```

Is there a faster way to do this?

Thanks in advance for any advice.

Ned

Subject: Re: Best way to move a window
Posted by Jaco van Gorkom on Fri, 01 Mar 2002 12:49:33 GMT
View Forum Message <> Reply to Message

"Ned Horning" <nedh@lightlink.com> wrote in message
news:uvis7ug8kfc4qtlnllpqj3223m19bo3pd3@4ax.com...

> Thanks Marc, this is pretty slick and it helps a lot. Unfortunately I
> just realized that I didn't post the second half of the window
> routine. After doing the "abs(tile_data - 180) lt 0.0001)" test I
> continue with:
>
> s= 45
> t=45
> IF (((ABS(tile_data[x+1,y+1] - 45) le s) AND (ABS(tile_data[x-1,y-1] -
> tile_data(x+1,y+1) -180) le t)) OR $
> ((ABS(tile_data[x+1,y] - 90) le s) AND (ABS(tile_data[x-1,y] -
> tile_data(x+1,y) -180) le t)) OR $
> ((ABS(tile_data[x+1,y+1] - 135) le s) AND (ABS(tile_data[x-1,y+1] -
> tile_data(x+1,y-1) -180) le t)) OR $
> ((ABS(tile_data[x,y-1] - 180) le s) AND (ABS(ABS(tile_data[x,y+1] -
> tile_data(x,y-1)) -180) le t))) THEN $
> out_tile[y*num_tile_samples+x] = 1 ELSE $
> out_tile[y*num_tile_samples+x] = 0
>
> What this does is compares opposing pixels (top and bottom, left and
> right, upper left lowerright, upper right and lower left) and I'm
> trying to see how to use the where/convolution logic for this and not
> having much luck.

Well, two options spring to mind:
- use the SHIFT() function, e.g. for left-right:
shiftleft = SHIFT(tile_data, -1, 0) & shiftright = SHIFT(tile_data, 1, 0)
expression:
(ABS(shiftleft - 90) LE s) AND (ABS(shiftright - shiftleft - 180) LE
t)

- use CONVOL() with kernels like [-1,0,1] for left-right, [[-1],[0],[1]]
for top-bottom, and 3x3 arrays for the diagonals. For left-right:
expression:
(ABS(shiftleft - 90) LE s) AND $
(ABS(CONVOL(tile_data, [-1,0,1]) - 180) LE t)

Then it should be possible to OR the above expressions directly together
and AND them into out_tile. If most of the elements get selected already
in the first half (180. being in their neigbourhood), then a loop over the
remaining elements might be a faster solution for this second half.

Getting totally confused about what this routine is actually doing,
Jaco

Subject: Re: Best way to move a window
Posted by Ned Horning on Fri, 01 Mar 2002 14:16:58 GMT

Jaco,

Thanks for the advice. It looks like I need to get more creative with CONVOL.

I am trying to detect ridges in a digital elevation model. I'm making the assumption that the sides of a ridge are sloping away from the ridgeline at roughly a 90 degree angle (i.e., if the ridge is north-south the west side of the ridge should have a west aspect) and the two sides of the ridge are roughly opposed (within a tolerance of t and s).

The reason I do the 180 check is because flat areas are assigned an aspect of 180 degrees and that really messes things up.

I also need to figure out how to calculate the mean of a moving window and work with a circular window but first I need to get this working faster.

Thanks again for your help.

Ned

On Fri, 1 Mar 2002 13:49:33 +0100, "Jaco van Gorkom" <j.c.van.gorkom@fz-juelich.de> wrote:

> "Ned Horning" <nedh@lightlink.com> wrote in message
> news:uvis7ug8kfc4qtlnllpqj3223m19bo3pd3@4ax.com...
>> Thanks Marc, this is pretty slick and it helps a lot. Unfortunately I
>> just realized that I didn't post the second half of the window
>> routine. After doing the "abs(tile_data - 180) lt 0.0001)" test I
>> continue with:
>>
>> s= 45
>> t=45
>> IF (((ABS(tile_data[x+1,y+1] - 45) le s) AND (ABS(tile_data[x-1,y-1] -
>> tile_data(x+1,y+1) -180) le t)) OR $
>> ((ABS(tile_data[x+1,y] - 90) le s) AND (ABS(tile_data[x-1,y] -
>> tile_data(x+1,y) -180) le t)) OR $
>> ((ABS(tile_data[x+1,y+1] - 135) le s) AND (ABS(tile_data[x-1,y+1] -
>> tile_data(x+1,y-1) -180) le t)) OR $
>> ((ABS(tile_data[x,y-1] - 180) le s) AND (ABS(ABS(tile_data[x,y+1] -
>> tile_data(x,y-1)) -180) le t))) THEN $
>> out_tile[y*num_tile_samples+x] = 1 ELSE $
>> out_tile[y*num_tile_samples+x] = 0
>>
>> What this does is compares opposing pixels (top and bottom, left and

>> right, upper left lowerright, upper right and lower left) and I'm
>> trying to see how to use the where/convolution logic for this and not
>> having much luck.
>
> Well, two options spring to mind:
> - use the SHIFT() function, e.g. for left-right:
> shiftleft = SHIFT(tile_data, -1, 0) & shiftright = SHIFT(tile_data, 1, 0)
> expression:
>   (ABS(shiftleft - 90) LE s) AND (ABS(shiftright - shiftleft - 180) LE
> t)
>
> - use CONVOL() with kernels like [-1,0,1] for left-right, [[-1],[0],[1]]
> for top-bottom, and 3x3 arrays for the diagonals. For left-right:
> expression:
>   (ABS(shiftleft - 90) LE s) AND $
>   (ABS(CONVOL(tile_data, [-1,0,1]) - 180) LE t)
>
> Then it should be possible to OR the above expressions directly together
> and AND them into out_tile. If most of the elements get selected already
> in the first half (180. being in their neigbourhood), then a loop over the
> remaining elements might be a faster solution for this second half.
>
> Getting totally confused about what this routine is actually doing,
>   Jaco
>
>
