Subject: Re: What could cause disappearing array?
Posted by robintw on Wed, 20 Jan 2010 08:42:12 GMT
View Forum Message <> Reply to Message

Hi,

Thanks for the feedback. Regarding the comment: that's an error in the comment not an error in the code! Silly me!

I've done a bit more investigation into this. I tried something that Jim Pendleton suggested off-list, which was to insert the code s = n_elements(segment_image) before my for loop, and then check the number of elements throughout my for loop by using the code If (s ne n_elements(segment_image)) then stop, stopping if there's an error.

I tried that, but it didn't catch the error as when segment_image disappears it also seems that s gets reset to zero. I can't see how that happens as s isn't even in the common block.

I've done some more investigation and found that it seems to be related to a for loop that I'm using to call this function. The function below calls the TRY_PLACING_REGION (the function I'm having problems with) from a for loop. When I comment out the for loop it runs fine, but with the for loop there I get this strange disappearing act!

The code for the top function is:

```
PRO DEVEREUX_SEG, t
  COMMON SegData, seg_id, segment_image

  ENVI_SELECT, fid=fid, dims=dims, pos=pos


  WholeImage = GET_IMAGE_DATA(fid, dims, pos)

  edge_map = CREATE_EDGE_MAP(WholeImage, 15.0)

  segment_image = lonarr(dims[2] + 1, dims[4] + 1)

  seg_id = long(20)

  FOR win_size = 5, 8 DO BEGIN
    TRY_PLACING_REGION, WholeImage, edge_map, 8
  ENDFOR
END
```

and if I comment out the for loop (ie. just leave the TRY_PLACING_REGION call by itself) then it works fine.

Also, Jim wanted to know if I was doing anything unusual in the INEQUALITY_FUNCTION. As far as I know I'm not - it's just standard IDL code. I've put the function below in case that helps.

```
FUNCTION INEQUALITY_FUNCTION, array, t
  dims = SIZE(array, /DIMENSIONS)

  FOR k = 0, dims[0] - 1 DO BEGIN
    FOR l = 0, dims[1] - 1 DO BEGIN
      FOR b = 0, dims[2] - 1 DO BEGIN
        result = abs(array[k, l, b] - mean(array[*, *, b]))^2 / t[b]
        IF result GT 1 THEN return, 0
      ENDFOR
    ENDFOR
  ENDFOR

  return, 1
END
```

Thanks for all the help so far - I hope we can manage to solve this!

Best regards,

Robin

On 19/01/2010 23:05, R.G. Stockwell wrote:
>
> "R.G. Stockwell" <noemail@please.com> wrote in message
> news:hj5dpa$v8n$1@speranza.aioe.org...
>>
>> "Robin Wilson" <r.t.wilson@rmplc.co.uk> wrote in message
>> news:z7GdnYab-uiEt8vWnZ2dnUVZ8jednZ2d@pipex.net...
>>> Hi,
>>>
>>> I've got a very strange problem in one of my IDL programs. I have two
>>> nested FOR loops with some processing happening in the inside loop,
>>> and after a bit of processing one of my arrays seems to disappear.
>>>
>>> I've checked this with some strategically placed help statements,
>>> when it seems to disappear it there (comes up with the right
>>> dimensions etc with the help command) at the bottom of the loop, but
>>> by the time the loop starts again it is showing as undefined.
>>>
>>> I was about to tell you how many iterations of my loop this occurred
>>> after (by looking at the value of i and j when it crashed) but after
>>> it's crashed I find i and j to both be equal to 0 - even though it's
>>> gone through most of the loop already.

>>>
>>> I can't see anywhere in the code that I'm assigning anything to i or
>>> j. I've left the FOR loop to look after them itself, yet somehow they
>>> end up as 0.
>>>
>>> When I comment out all of the code inside the for loop then the loop
>>> runs perfectly to the end, and i and j don't get reset to zero part
>>> way through.
>>
>> So, comment out one line at a time, and see what happens. :)
>>
>> What does the output of those lines say (print, i,j, and the help
>> commands).
>> Are you saying that the array segment_image dissapears between the
>> seg_id++ statement
>> (the endfor) and the next print,i print,j,and help, statements?
>>
>> One note: that array is in a common block, who knows who else is
>> manipulating it.
>>
>>
>> One last thing: your comment and your statement don't match.
>
>
> Oops. to elaborate on that (accidently sent the message)
> Here.
>
> ; If one of the pixels had a inequality value greater than 1 then
> continue
> if inequality_result EQ 0 THEN CONTINUE
