Subject: Re: image correlation
Posted by jkj on Thu, 23 Apr 2009 01:46:13 GMT
View Forum Message <> Reply to Message

On Apr 21, 4:43 am, Fabinho <fabioviann...@gmail.com> wrote:
> Good morning Brian or everyone,
> I read the topics inhttp://groups.google.com/group/comp.lang.idl-pvwave/browse _frm/thread...,
> as you recommended.
>
> I dont know why but im still having a lot of trouble with pvwave.
> Honnestly, im not an expert in programming. First I tryed to open the
> apple routine athttp://people.bu.edu/balarsen/Home/IDL/Entries/2009/4/6_Im age_registr....
> It didnt work at all! What should I do with the two routines, do I
> have to put in the same file? the routine "wheretomulti" before, and
> than the image-registration. Right? I tryed to do it, I also tryed to
> have 2 differents files in the same folder, but when i tryed to
> compile there was a lot of synthax problems. Maybe im not using the
> software correctly? It seems that the software finds the file he is
> supposed to compile, but he doesnt understand it at all.
> After i tryed to run the code that wox made, but also didnt work, i
> changed the name of the rose picture to one picture that i had, didnt
> work.
> I would be really thankful if someone are able to help me. Im working
> for a multinational company in france, my boss gave this part to me as
> a challenge!
>
> thanks
> ps: wox's code
> CODE:
> ;------------------------
> pro test
> path = Filepath(Subdir=['examples', 'data'], 'rose.jpg')
> read_jpeg, path, img, /true
> img = 0.3*Reform(img[0,*,*]) + 0.59*Reform(img[1,*,*]) +
> 0.11*Reform(img[0,*,*])
>
> kernelSize = [10,10]
> kernel = REPLICATE((1./(kernelSize[0]*kernelSize[1])), $
> kernelSize[0], kernelSize[1])
> img2= CONVOL(img, kernel, /CENTER, /EDGE_TRUNCATE)
>
> print,image_equal(img,img2,/outid)
> end;pro test
> ;------------------------
> function
> image_equal,img1,img2,npix=npix,shifttol=shifttol,Rtol=Rtol, outid=outid
> ; Image offsets or scales don't matter
> ; npix: subimage pixels for cross-correlation

```
>  ; shifttol: subimage shift tollerance
>  ; Rtol: cross-correlation tollerance
>
>  s1=size(img1,/dim)
>  s2=size(img2,/dim)
>  msize=s1[0]<s1[1]<s2[0]<s2[1]
>
>  if not keyword_set(npix) then npix=fix(msize*0.4)>10 ; 40% of the size
>  npix<=msize
>
>  if n_elements(shifttol) eq 0 then shifttol=(msize*0.01)>1 ; 1% of the
>  size
>  if not keyword_set(Rtol) then Rtol=0.9
>
>  ; Subimages in img2
>  nsub=s2/npix
>  nx=nsub[0]
>  ny=nsub[1]
>  x0=npix*indgen(nx)
>  x1=[x0[1:*],s2[0]]-1
>  y0=npix*indgen(ny)
>  y1=[y0[1:*],s2[1]]-1
>
>  ; img2 subimages in img1
>  xoff=lonarr(nsub)
>  yoff=xoff
>  xyccor=fltarr(nsub)
>  if keyword_set(outid) then img2recon=img1*0
>
>  ; Cross-correlate subimages of img2 with img1
>  for i=0,nx-1 do $
>          for j=0,ny-1 do begin
>                  sub=img2[x0[i]:x1[i],y0[j]:y1[j]]
>                  ssub=size(sub,/dim)-1
>
>                  ; Number of sub-shifts in img1
>                  noffx=s1[0]-ssub[0]
>                  noffy=s1[1]-ssub[1]
>                  ccor=fltarr(noffx,noffy)
>
>                  ; Correlate sub with img1
>                  for k=0,noffx-1 do $
>                          for l=0,noffy-1 do $
>    ccor[k,l]=c_correlate(sub,img1[k:k+ssub[0],l:l+ssub[1]],0)
>
>                  ; Sub image offset and cross-correlation
>                  mccor=max(ccor,moff)
>                  k=moff mod noffx
```

```
>               l=moff/noffx
>
>               xoff[i,j]=k
>               yoff[i,j]=l
>               xyccor[i,j]=mccor
>
>               if keyword_set(outid) then begin
>                     img2recon[k,l]=sub
>                     print,'Progress: ',(i*ny+j+1.)/(nx*ny)*100,'%'
>               endif
>         endfor
>
>  ; Check whether img2 and img1 are equal
>  bsame=total(xyccor lt Rtol,/pres) eq 0
>  bsame and= total(rebin(total(xoff,2)/ny,nx,ny)-xoff gt shifttol,/pres)
>  eq 0
>  bsame and= total(rebin(reform(total(yoff,1),1,ny)/nx,nx,ny)-yoff gt
>  shifttol,/pres) eq 0
>
>  if keyword_set(outid) then begin
>         window
>         tvscl,img1,0
>         tvscl,img2,1
>         tvscl,img2recon,2
>         tvscl,img2-img2recon,3
>
>         xyouts,0.1,0.7,'img1',/normal,color=100
>         xyouts,0.3,0.7,'img2',/normal,color=100
>         xyouts,0.5,0.7,'reconstructed img2',/normal,color=100
>         xyouts,0.7,0.7,'img1 - reconstructed img2',/normal,color=100
>         isurface,xyccor
>  endif
>
>  return,bsame
>  end;function image_equal
>
>  On 21 avr, 08:58, Fabinho <fabioviann...@gmail.com> wrote:
>
>>  thanks a lot! I will read it and get started!
>>  thks
>
>>  On 21 avr, 00:20, Brian Larsen <balar...@gmail.com> wrote:
>
>>>  I would start with a read through this post and see if that provides a
>>>  starting point.
>
>>>  http://groups.google.com/group/comp.lang.idl-pvwave/browse_f rm/thread...
>
```

>>> Cheers,
>
>>> Brian
>
>>> ----------
>>> Brian Larsen
>>> Boston University
>>> Center for Space Physicshttp://people.bu.edu/balarsen/Home/IDL
>
>

The compile errors probably mean you have some problems with the path setting in the environment. Of course, I only use IDL and have never seen PV-Wave so I don't know how you address the PATH issues in PV-Wave but it should be easy to check the docs.

-Kevin
