Subject: Cross-correlation of two images?
Posted by Craig Hamilton on Wed, 02 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Hi IDLers: (Hmmm, just realized what that spells - entirely inappropriate,
this is the most productive newsgroup on the net!!)

I want to compute the cross-correlation of two images and have just
had a browse of the documentation. Seems that I've got to put it
together myself, either using convolv() after suitable transposition of
one of the images, or either doing it using FFTs.

I bet someone has already got a canned routine that does this.
Am I right?

Thanks for any pointers,
Craig Hamilton
cah@medeng.wfubmc.edu

Subject: Re: Cross-correlation of two images?
Posted by wbiagiot on Fri, 04 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

"Craig Hamilton" <cah@medeng.wfubmc.edu> wrote:
>
> I want to compute the cross-correlation of two images and have just
> had a browse of the documentation. Seems that I've got to put it
> together myself, either using convolv() after suitable transposition
of
> one of the images, or either doing it using FFTs.
>
> I bet someone has already got a canned routine that does this.
> Am I right?
>

Hi Craig,

I haven't tried cross correlation with images, but a cursory
examination of IDL's C_CORRELATE routine looks like it will handle 2
dimensional arrays. Isn't that what you are looking for? If the answer
is yes, I have a modified version of C_CORRELATE (available on Ron K.'s
website) which runs about 60% faster on computations with multiple lag
factors, that you might be interested in.

-Bill B.

Sent via Deja.com http://www.deja.com/
Share what you know. Learn what you don't.

Subject: Re: Cross-correlation of two images?
Posted by badastro on Mon, 07 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

In comp.lang.idl-pvwave Craig Hamilton <cah@medeng.wfubmc.edu> wrote:

> I want to compute the cross-correlation of two images and have just
> had a browse of the documentation. Seems that I've got to put it
> together myself, either using convolv() after suitable transposition of
> one of the images, or either doing it using FFTs.

> I bet someone has already got a canned routine that does this.
> Am I right?

Yup. Go to:

http://idlastro.gsfc.nasa.gov/contents.html

Under "Image Manipulation" is a list of routines starting
with 'correl*'.

Or, you could take the total of each image in each
direction (y1 = total(image1,1) & x1 = total(image1,2)) and cross correlate
the 1D vectors.

* * * * * The Bad Astronomer * * * *

Phil Plait badastro@badastronomy.com
The Bad Astronomy Web Page: http://www.badastronomy.com

Subject: Re: Cross-correlation of two images?
Posted by Carl G. Zimba on Tue, 22 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Craig,

Try the attached file. It was developed for aligning images from x-ray
microscopy but should be generic enough to use for all images.

If you want more info, e-mail me directly.

Carl

Craig Hamilton wrote:
>
> Hi IDLers: (Hmmm, just realized what that spells - entirely inappropriate,
>   this is the most productive newsgroup on the net!!)
>
> I want to compute the cross-correlation of two images and have just
> had a browse of the documentation. Seems that I've got to put it
> together myself, either using convolv() after suitable transposition of
> one of the images, or either doing it using FFTs.
>
> I bet someone has already got a canned routine that does this.
> Am I right?
>
> Thanks for any pointers,
> Craig Hamilton
> cah@medeng.wfubmc.edu

--
****

Carl G. Zimba
National Institute of Standards and Technology
Material Science and Engineering Laboratory
Polymer Division
MS 224 / B108
Gaithersburg, MD 20899
301.975.6881 VOX
301.975.4932 FAX
zimba@nist.gov
http://www.nist.gov/zimba

```
;+
; NAME:
; ZSTACK_ALIGN_IMAGES.PRO
; AUTHORS:
; Carl G. Zimba (NIST), Chris Jacobsen (SUNY - Stonybrook)
; PURPOSE:
; Alignment of two images abtained on an x-ray microscope.
; Called by ZSTACK_ALIGN.PRO.
; CATEGORY:
; Data analysis.
; CALLING SEQUENCE:
; zstack_align_images,image1,image2,xshift,yshift
; INPUTS:
; image1 = array of image 1, this is usually the FFT of the first image to be aligned
```

```
; image2 = array of image 2
; KEYWORD PARAMETERS:
; fftpass = set to indicate that the image1 has been FFT'ed
; sobel = set for sobel edge edge enhancement
; roberts = set for roberts edge enhancement
; edgegauss = number of pixels used for edge smoothing
; cm = set for alignment using center of mass of correlation function
;  if not set, alignment is done using quadratic fit of correlation function
; maxshift = maximum shift allowed during alignment
; dragbox = dimensions of box defining area used for alignment
; debug = set for degugging
; help = set to print how-to-use message
; OUTPUTS:
; xshift = alignment shifts along x-axis
; yshift = alignment shifts along y-axis
; corr_image = image of correlation function, displayed by ZSTACK_ALIGN.PRO
; corr_dims = dimensions of the maxima location of the correlation function
; COMMON BLOCKS:
; NONE
; SIDE EFFECTS:
;
; RESTRICTIONS:
;
; PROCEDURE:
;   Called by ZSTACK_ALIGN.PRO.
; EXAMPLE:
;
; MODIFICATION HISTORY:
; Modified 25-mar-1998 to deal with 24 bit color.
;   If you provide the white, red, and green colors, then
;   it is assumed that the color table is preloaded.
;
; Significantly modified on July 11, 1998.  Eliminated
; the correl option, and eliminated the /peak keyword because
; it is now the default.  Fit the three pixels about the
; peak to z=a+bx*cx^2 and z=a+by+cy^2 for subpixel location.
; Improve the center of mass peak location.
;
; Modified July 13, 1998 to let one do a sequence with 2N+1
; FFTs rather than 3N by using the /fftpass option.  If this
; option is selected, then it is expected that the "image1"
; you provide is already transformed, and the transformed
; version of image2 is returned to you (for use as a
; pre-transformed image1 in a subsequent call to ALIGN).
;
; Modified 28-aug-1998 to deal with 24 bit color by demanding
; device,decomposed=0
;
```

```
; Modified extensively procedure from ALIGN.PRO, 20feb99, CGZ
; This procedure now ONLY does alignment of two images,
;  yielding the alignment shifts, the correlation function,
;  and the dimensions of the center of the correlation function as output.
; Eliminated display routine, now done within STACK_ALIGN.PRO
; Eliminated image shift routine, now done in STACK_ALIGN.PRO,
;  where you now have a choice to keep, discard, or redo alignment
; Eliminated several keywords:
;  meanfill, medianfill, xcorimg_win, xcorimg_zoom, xplot_win, yplot_win
; Added dragbox so that correlation is only done on dragbox region
; Added corr_image, corr_dims
;  to display correlation function in ZSTACK_ALIGN.PRO
; Modified use of fftpass option
;  Now skips most of section to determine correlation peak
;  during first call of stack_align_images (without fftpass option)
; Modified center-of-mass calculation
;  Previous method found the center of mass of entire correlation function
;   including outlying regions of significant intensity.  This is a
;   common problem when dragbox is specified.
;  Method now excludes the outlying regions of intensity and calculates
;   the center of mass from just the most central region.
;
;.....................................................................
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

PRO  zstack_align_images,image1,image2,xshift,yshift,corr_image,c orr_dims, $
  fftpass=fftpass,sobel=sobel,roberts=roberts,edgegauss=edgega uss,$
 cm=cm,maxshift=maxshift,dragbox=dragbox, $
 debug=debug,help=help

;print,'stack_align_images'

@bsif_common ; added, CGZ

IF (keyword_set(help) OR (n_params() eq 0)) THEN BEGIN
    print,'align,image1,image2,xshift,yshift,image2_shifted,'
    print,'  Modifiers to input images: /sobel or /roberts, edgegauss_sigma='
    print,'    (if you simply do /edgegauss, you will get edgegauss_sigma=4.)'
    print,'  Constraints on peak finding: /cm, maxshift='
    return
ENDIF

IF (n_elements(sobel) EQ 0) THEN sobel = 0
IF (n_elements(roberts) EQ 0) THEN roberts = 0
;change to CASE structure ??
IF (n_elements(edgegauss) EQ 0) THEN BEGIN
    edgegauss_sigma = 0.
ENDIF ELSE IF (float(edgegauss) LE 1.) THEN BEGIN
    edgegauss_sigma = 4.
```

```
ENDIF ELSE BEGIN
   edgegauss_sigma = edgegauss
ENDELSE
IF (n_elements(cm) EQ 0) THEN cm = 0

svec=size(image1)
IF (svec(0) NE 2) THEN BEGIN
   print,'image1 has '+strtrim(string(svec(0)),2)+' rather than 2 dimensions'
   return
ENDIF
nx1 = svec(1)
ny1 = svec(2)

IF (n_elements(maxshift) EQ 0) THEN maxshift = fix(min(0.8*[nx1/2,ny1/2]))

svec=size(image2)
IF (svec(0) NE 2) THEN BEGIN
   print,'image2 has '+strtrim(string(svec(0)),2)+' rather than 2 dimensions'
   return
ENDIF
nx2 = svec(1)
ny2 = svec(2)

IF ((nx1 NE nx2) OR (ny1 NE ny2)) THEN BEGIN
   print,'Images must match in size (align 195)'
   return
ENDIF
nx = nx1 ; number of columns in image (as enclosed by dragbox)
ny = ny1 ; number of rows in image (as enclosed by dragbox)
xcenter = 0
ycenter = 0

;; Define dragbox dimensions, CGZ
IF ((dragbox(2) NE 0) AND (dragbox(3) NE 0)) THEN BEGIN
   xleft = min([dragbox(0),dragbox(2)],max=xright)
   ybot = min([dragbox(1),dragbox(3)],max=ytop)
   xleft = xleft>0    ; left column of dragbox
   xright = xright<(ny-1)   ; right column of dragbox
   ybot = ybot>0    ; bottom row of dragbox
   ytop = ytop<(nx-1)   ; top row of dragbox
ENDIF ELSE BEGIN  ; set dimensions of dragbox to entire image
   xleft = 0
   xright = ny-1
   ybot = 0
   ytop = nx-1
ENDELSE

;; Create fft1 and fft2 arrays - reduced in size to dragbox region
```

```
;; Reduces size of FFT calculation by only using dragbox data
fft1 = fltarr(nx,ny)
fft2 = fltarr(nx,ny)

;; We make our own copy of the input image and call it fft in anticipation
;; of what we will do to it later.

;change to CASE structure ??
IF (sobel NE 0) THEN BEGIN
 IF (not keyword_set(fftpass)) THEN BEGIN
  fft1 = sobel(image1)
 ENDIF
 fft2 = sobel(image2)
ENDIF ELSE IF (roberts NE 0) THEN BEGIN
 IF (not keyword_set(fftpass)) THEN BEGIN
  fft1 = roberts(image1)
 ENDIF
 fft2 = roberts(image2)
ENDIF ELSE BEGIN
 IF (not keyword_set(fftpass)) THEN BEGIN
  fft1 = image1
 ENDIF
 fft2 = image2
ENDELSE

IF (edgegauss_sigma NE 0.) THEN BEGIN
    IF (not keyword_set(fftpass)) THEN BEGIN
       edgegauss,fft1,edgegauss_sigma,dc1,/zero_edge
    ENDIF
    edgegauss,fft2,edgegauss_sigma,dc2,/zero_edge
ENDIF

;; Did we get the FFT of the image passed instead of the
;; image itself?
IF keyword_set(fftpass) THEN BEGIN
 fft1 = image1
ENDIF ELSE BEGIN
    fft1 = shift(temporary(fft1),nx/2,ny/2)
    fft1 = fft(fft1,-1,/overwrite)
    fft1 = shift(temporary(fft1),nx/2,ny/2)
ENDELSE

fft2 = shift(temporary(fft2),nx/2,ny/2)
fft2 = fft(fft2,-1,/overwrite)
fft2 = shift(temporary(fft2),nx/2,ny/2)

;; For these calculations, we are quite willing to mess with
;; fft1 but we don't want to modify fft2 since it will supply
```

```
;; fft1 in a subsequent call to ALIGN
fft1 = conj(temporary(fft1))
fft1 = fft2*temporary(fft1)

fft1 = shift(temporary(fft1),nx/2,ny/2)
fft1 = fft(fft1,1,/overwrite)
fft1 = shift(temporary(fft1),nx/2,ny/2)
fft1 = abs(temporary(fft1))


;; Find the maximum of the correlation function
;; prepare dimensions of box to find correlation maximum (center +/- maxshift
cleft  = (nx/2 - maxshift)>0
cright = (nx/2 + maxshift)<(nx-1)
cbot   = (ny/2 - maxshift)>0
ctop   = (ny/2 + maxshift)<(ny-1)
; maximum and minimum of correlation function within center window
max_fft1 = max(fft1(cleft:cright,cbot:ctop),max_index,min=min_fft1)
; set (xcenter,ycenter) at maximum of correlation function
xcenter = cleft+(max_index mod (cright-cleft+1))
ycenter = cbot+(max_index / (ctop-cbot+1))
IF keyword_set(debug) THEN BEGIN
    print,'Maximum value is at ['+$
      strtrim(string(xcenter),2)+','+$
      strtrim(string(ycenter),2)+']'
ENDIF

IF (keyword_set(fftpass)) THEN BEGIN
 IF (cm NE 0) THEN BEGIN ; align using center of mass of correlation function
  cm_threshold = min_fft1 + (0.5*(max_fft1-min_fft1)) ; mean intensity of correlation function
  cm_image = fft1
  cm_indices=where((fft1 GE cm_threshold),n_cm)

  cm_array = cm_image
  cm_array(*,*) = 0
  cm_array(cm_indices) = 255

  ; cm_indices can contain regions outside of central correlation peak
  ; which must be excluded before calculating center of mass
  ; This is particularly true if dragbox is used
  ; So exclude any cm_indices which are a significant distance away from center
  ; Create an array of points whose correlation intensity is above mean threshold
  ; i.e, cm_indices = a linear array of indices corrsponding to points in fft1
  ;     above cm_threshold
  ; Large differences between successive points in cm_indices correspond to
  ; boundary between areas of fft1 above and below cm_threshold
  ; Select central region of correlation function based on these boundaries
  ; cm_x =  x-coordinate of data above cm_threshold
```

```
 ; cm_y = y-coordinate of data above cm_threshold
 ; cm_dist = distance from center of data above cm_threshold
 ; cm_array = binary array representing central region of correlation function above cm_threshold
 cm_x = cm_indices mod nx
 cm_y = floor(cm_indices/ny)
 cm_dist = sqrt( (cm_x - xcenter)^2 + (cm_y - ycenter)^2 )

 ; Find biggest changes in distance array by subtraction
 ; between cm_dist and +/-1 shifted cm_dist
 shift_p1 = shift(cm_dist,+1)
 shift_p1(0) = cm_dist(0)
 shift_m1 = shift(cm_dist,-1)
 shift_m1(n_elements(shift_m1)-1) = cm_dist(n_elements(cm_dist)-1)
 max_dummy = max((shift_p1 - cm_dist),min_index)
 min_dummy = min((cm_dist - shift_m1),max_index)
 IF ((max_dummy - min_dummy) GT (2*maxshift)) THEN BEGIN
  ; value of 2*maxshift is empirical and speculative
  IF (max_index GT min_index) THEN BEGIN
   cm_array = cm_indices(min_index:max_index)
   cm_indices = cm_array
  ENDIF
  IF (max_index LT min_index) THEN BEGIN
   cm_array = cm_indices(max_index:min_index)
   cm_indices = cm_array
  ENDIF
  ; above conditionals used to avoid problem if min_index < max_index
  ; seems to only be a problem with first call of stack_align_images
  ; which is probably eliminated by restructure use of /fftpass option
 ENDIF

 cm_array = cm_image
 cm_array(*,*) = 0
 cm_array(cm_indices) = 255

 IF (n_cm GE 4) THEN BEGIN
  xpositions = cm_indices mod nx
  ypositions = floor(cm_indices/nx)
  inverse_mass_total = 1./total(fft1(cm_indices))
   xcenter=total(xpositions*cm_image(cm_indices))*inverse_mass_ total
   ycenter=total(ypositions*cm_image(cm_indices))*inverse_mass_ total

  IF keyword_set(debug) THEN BEGIN
   print,'Center of mass is at ['+$
    strtrim(string(xcenter,format='(f10.2)'),2)+','+$
    strtrim(string(ycenter,format='(f10.2)'),2)+']'
  ENDIF
 ENDIF
ENDIF ELSE BEGIN ; align using quadratic fit of maximum of correlation function
```

```
  xpts = xcenter+[-1,0,1]
  ypts = fft1((xcenter-1):(xcenter+1),ycenter)
  tri_fit,xpts,ypts,xfit,xpeak
   xpts = fft1(xcenter,(ycenter-1):(ycenter+1))
  ypts = ycenter+[-1,0,1]
  tri_fit,ypts,xpts,yfit,ypeak
  xcenter = xpeak
  ycenter = ypeak
  IF keyword_set(debug) THEN BEGIN
   print,'Quadratic center is at ['+$
    strtrim(string(xcenter,format='(f10.2)'),2)+','+$
    strtrim(string(ycenter,format='(f10.2)'),2)+']'
  ENDIF
 ENDELSE

ENDIF ELSE BEGIN ; (NOT keyword_set(fftpass))
; print,'not doing alignment'
ENDELSE

;  Define dimensions for correlation image, CGZ
IF ((dragbox(2) NE 0) AND (dragbox(3) NE 0)) THEN BEGIN
; print,'made it to kilroy'
 xleft = min([dragbox(0),dragbox(2)],max=xright)
 ybot = min([dragbox(1),dragbox(3)],max=ytop)
 cycenter = ybot+ycenter
 cxcenter = xleft+xcenter
 corr_image = fltarr(n_cols,n_rows)
 corr_image(xleft:xright,ybot:ytop) = fft1
 corr_dims = [cxcenter,cycenter]
ENDIF ELSE BEGIN
 corr_image = fft1
 corr_dims = [xcenter,ycenter]
ENDELSE

fft1 = 0
xshift = xcenter - double(nx/2)
yshift = ycenter - double(ny/2)

;; Save fft2 for use as fft1 for a subsequent call to ALIGN
;IF keyword_set(fftpass) THEN BEGIN
 image2 = fft2
;ENDIF

return
END
```

File Attachments

1) zstack_align_images.pro, downloaded 101 times