Subject: Re: HELP: replace missing value with closest good pixel
Posted by paul on Wed, 14 Jun 1995 07:00:00 GMT
View Forum Message <> Reply to Message

In article <3rkser$sf3@llnews.ll.mit.edu>, knight@ll.mit.edu writes:
|>
|> I have 2-D images with flaws, i.e., missing pixels.  I want to replace each
|> missing pixel with the closest good pixel.  Does anybody have a routine to do
|> this?  For example,
|>
|>  new = replace(image,missing=99)
|>
|> where
|>  image = 2-D array
|>  missing = keyword to specify the value which is to be replaced
|>  new = image with each pixel having value 99 replaced by closest pixel
|>              with a value not equal to 99.
|>
|> Thanks for any help,
|> Fred
|>
|>
|> --
|> =Fred Knight (knight@ll.mit.edu) (617) 981-2027
|>  C-483\\MIT Lincoln Laboratory\\244 Wood Street\\Lexington, MA  02173

--

try this:

```
pro  fill_image,a,ii,niter=niter,view=view,flick=flick,omega=omeg a,tol=tol,po=po
;+
; ROUTINE:          fill_image
;
; USEAGE:           fill_image,a,ii
;                   fill_image,a,ii,niter=niter,view=view,flick=flick,$
;                              omega=omega,tol=tol,po=po
;
; PURPOSE:          fill in undefined regions of a 2-d array by interpolation
;
; INPUT:
;       a           image array with some undefined points
;       ii          index array of bad image points,
;                   E.G., ii=where(aa eq 999)
; KEYWORD INPUT
;       tol         maximum tolerance to achieve before stopping iteration
;                   (default=0.001)
```

```
;       niter       number of smoothing iterations (default=100)
;       view        if set, TV current array every VIEW iterations
;       po          if set print diagnostic print out every PO iterations
;       flick       if set, display comparison of input and output arrays
;       omega       over-relaxation parameter (default=1.0)
;                   a value of omega slightly greater than 1 may speed up
;                   convergence (too large and the iteration will fail).
;
; OUTPUT:
;       a           image array with initially undefined points replaced
;                   with values that vary smoothly in both the horizontal and
;                   vertical directions.  Initially defined points are
;                   unchanged.
;
; PROCEDURE:        In the undefined regions, the image field is assumed
;                   to have minimal curvature.  Hence the image field obeys
;                   Laplace's equation:
;
;                         div.div(a)=0
;
;                   This equation is solved by the method of successive over
;                   relaxation over the set of undefined points: the A array
;                   is unchanged in the initially defined regions.
;
;                   In index notation the iterated equation is,
;
;                        a(i,j)=(1-w)*a(i,j)+.25*w*(a(i+1,j)+a(i,j+1)+
;                                        a(i-1,j)+a(i,j-1))
;
;                   where w is the over-relaxation parameter
;
;                   (filling a 3-d array should be a simple extension of this
;                   technique)
;
; EXAMPLE:
;
;          im=dist(256)
;          im(where(smooth(randomu(iseed,256,256),5) gt .6))=-999.; bad values
;          ii=where(im eq -999.)
;          fill_image,im,ii,/view
;
; AUTHOR            Paul Ricchiazzi                         29oct92
;                   Institute for Computational Earth System Science
;                   University of California, Santa Barbara
;-
if keyword_set(niter) eq 0 then niter=100
if keyword_set(tol) eq 0 then tol=.001
```

```
sz=size(a)
nx=sz(1)
ny=sz(2)
xx=ii mod nx
yy=ii / nx
;
; set up shifted index arrays
;
xp=xx + 1 & i=where(xp ge nx) & if i(0) ne -1 then xp(i)=nx-2
xm=xx - 1 & i=where(xp lt 0)  & if i(0) ne -1 then xp(i)=1
yp=yy + 1 & i=where(yp ge ny) & if i(0) ne -1 then yp(i)=ny-2
ym=yy - 1 & i=where(yp lt 0)  & if i(0) ne -1 then yp(i)=1
ind1=xm+ym*nx
ind2=xp+ym*nx
ind3=xm+yp*nx
ind4=xp+yp*nx
;
; set over-relaxation parameter
;
if keyword_set(omega) then w=omega else w=1.0
if keyword_set(po) eq 0 then po=0
aa=a
iok=lindgen(nx*ny)
iok(ii)=-1
iok=iok(where(iok gt -1))
amin=min(a(iok))
amax=max(a(iok))
toler=tol^2*(abs(amax) > abs(amin))*n_elements(ii)
;
; iterate to solve LaPlace's equation
;
for i=1,niter do begin
  delta=.25*(aa(ind1)+aa(ind2)+aa(ind3)+aa(ind4)) - aa(ii)
  aa(ii)=aa(ii)+delta*w
  if keyword_set(view) then if i mod view eq 0 then $
     tv,bytscl(aa,min=amin,max=amax,top=!p.color)
  if total(delta^2) lt toler then goto,done
  if keyword_set(po) then if i mod po eq 0 then print,i,total(delta^2),toler
  if max(aa(ii)) gt amax or min(aa(ii)) lt amin then w=1.
endfor

print,'Warning from FILL_IMAGE: filled array values are not fully converged'

done:
if keyword_set(flick) then begin
  a(ii)=min(aa)
  flick,bytscl(a),bytscl(aa)
endif
```

a=aa
end

Paul Ricchiazzi email: paul@icess.ucsb.edu
Institute for Cumputational Earth System Science
University of California, Santa Barbara

If I have seen farther than others, it is by standing on the shoulders of giants.
Isaac Newton

If I have not seen as far as others, it is because giants were standing on my shoulders.
Hal Abelson

---

Subject: Re: HELP: replace missing value with closest good pixel
Posted by sterner on Wed, 14 Jun 1995 07:00:00 GMT
View Forum Message <> Reply to Message

knight@ll.mit.edu writes:

> I have 2-D images with flaws, i.e., missing pixels. I want to replace each
> missing pixel with the closest good pixel. Does anybody have a routine to do
> this? For example,

> new = replace(image,missing=99)

> where
> image = 2-D array
> missing = keyword to specify the value which is to be replaced
> new = image with each pixel having value 99 replaced by closest pixel
> with a value not equal to 99.

Here's a simple technique that seems to work well.

```
new = image
w = where(new eq missing)
m = median(new,3)
new(w) = m(w)
```

The median filter does a nice job filling in the missing pixels
(if the missing value is outside the range for good pixels).
The size of the filter can be adjusted for the severity of the
missing data.  Only the missing pixels are modified, the rest of the
image is untouched.

Ray Sterner  sterner@tesla.jhuapl.edu
The Johns Hopkins University  North latitude 39.16 degrees.
Applied Physics Laboratory  West longitude 76.90 degrees.
Laurel, MD 20723-6099
WWW Home page:  ftp://fermi.jhuapl.edu/www/s1r/people/res/res.html

---