Supplementary Text

Supplementary Text 1: Mathematical operators used by algorithms 018330 and 025886 (Interactive Data Language code)

MEAN

```
PRO ga_mean, in, out, w, s

  kernel = gs_mselt(w,s)

  out = gs_convolve(in,kernel,/ave)

END
```

IFLTE

```
PRO ga_iflte, in1, in2, in3, in4, out
;+
;     GA Project: If less than else
;
;-

  w1 = in1 lt in2
  w2 = in1 ge in2
  out = w1 * in3 + w2 * in4

END
```

ADDP

```
PRO ga_addp, in1, in2, out
;+
;     Add two data planes
;-

  out = in1 + in2

END
```

QTREG

```
PRO ga_qtreg,inplane,outplane,slopes,offsets,threshin
;+
;GA Project: Returns the region size (in log base 2) around each pixel for which the normalized variance per pixel standard of the square region first reaches a given threshold. Also returns planes with the linear fit slope and offset of the variance as a function of region scale for each pixel
;
Calling Sequence: ga_qtreg,inplane,outplane,slopes,offsets,thresh
;
Parameters:
;inplane = Input plane
;outplane = Output plane with log base 2 region sizes
;slopes = Output slopes of fits for variance/pixel vs. log region size
;offsets = Output offsets of fits for variance/pixel vs. log region size
;thresh = Input fractional threshold for the variance (between 0 and 1).
;If thresh is greater than 1, then the module of the value\ is used. If thresh is negative, the absolute value is used.

-thresh=abs(threshin) mod 1.0
     maxi=5
   d=maxi*total(findgen(maxi)^2.0)-(total(findgen(maxi))^2.0)
     sumx=total(findgen(maxi))
   outplane=inplane
   outplane[*]=0
   sumy=outplane
   sumxy=outplane
   for i=1l,maxi do begin
        sm=(2l^i)+1l
        mn=smooth(inplane,sm,/edge)
         var=sqrt(smooth((inplane-mn)^2,sm,/edge) > 0)
        tst=where(var lt thresh*mn,nc)
        if nc gt 0 then outplane[tst]=i
        tst=0
        sumy=var+temporary(sumy)
        sumxy=i*var+temporary(sumxy)
        var=0
    endfor
    slopes=(maxi*sumxy-sumx*sumy)/(d>1)
     offsets=abs(sumy-slopes*sumx)/maxi
    sumx=0
    sumxy=0

    return
end
```

RANGE

```
PRO ga_range, in, out, w, s

; Local Range Values (Also Called Morphological Gradient)

  ; scale input
  tmp1 = gs_discretize(in)
  tmp2 = tmp1

  mse = gs_mselt(w,s)

  ;; Dilating input

  tmp1 = gs_padimage(tmp1,w)
  tmp1 = dilate(tmp1,mse,/gray,/ulong)
  tmp1 = gs_padimage(tmp1,w,/unpad)

  ;; Eroding input

  tmp2 = gs_padimage(tmp2,w)
  tmp2 = erode(tmp2, mse, /gray,/ulong)
  tmp2 = gs_padimage(tmp2,w,/unpad)

  ;; Then take Difference
  ;; rescale output
  tmp1 = gs_discretize(tmp1,/undo)
  tmp2 = gs_discretize(tmp2,/undo)

  out = abs(tmp1 - tmp2)

END
```

ASF_CLOP

```
PRO  ga_asf_clop, in, out, N, s

; Alternating Sequential Filters (Close-Open)

; scale input
   tmp = gs_discretize(in)

   for w = 1, N do begin

      mse = gs_mselt(w,s)

 ; Close (Dilate-Erode)

      tmp = gs_padimage(tmp,w)
      tmp = dilate(tmp,mse,/gray,/ulong)
      tmp = gs_padimage(tmp,w,/unpad)

      tmp = gs_padimage(tmp,w)
      tmp = erode(tmp,mse,/gray,/ulong)
      tmp = gs_padimage(tmp,w,/unpad)

  ; Close (Erode-Dilate)

      tmp = gs_padimage(tmp,w)
      tmp = erode(tmp,mse,/gray,/ulong)
      tmp = gs_padimage(tmp,w,/unpad)

      tmp = gs_padimage(tmp,w)
      tmp = dilate(tmp, mse, /gray,/ulong)
```

```
        tmp = gs_padimage(tmp,w,/unpad)

     end

 ; rescale output
     out = gs_discretize(tmp,/undo)

END
```

DILATE

```
PRO ga_dilate, in, out, w, s
;+
; GA Project: out[n] = min( in[i] ) for all i within w of n
;-
   w=(w>0)

   mse = gs_mselt(w,s)

   ;; scale input
   tmp = gs_discretize(in)

   tmp = gs_padimage(tmp,w)
   tmp = dilate(tmp,mse,/gray,/ulong)
   tmp = gs_padimage(tmp,w,/unpad)

   ;; rescale output
   out = gs_discretize(tmp,/undo)

END
```

ERODE

```
RO ga_erode, in, out, w, s
;+
;       GA Project: out[n] = min( in[i] ) for all i within w of n

   w=(w>0)

  mse = gs_mselt(w,s)          ;morphological structure element

   ;; scale input
   tmp = gs_discretize(in)

   tmp = gs_padimage(tmp,w)
   tmp = erode(tmp,mse,/gray,/ulong)
   tmp = gs_padimage(tmp,w,/unpad)

   ;; rescale output
   out = gs_discretize(tmp,/undo)

END
```

OPEN_CLOSE

```
PRO  ga_open_close, in, out, w, s

  ;; Erode - Dilate - Dilate - Erode

  ;; scale input
  tmp = gs_discretize(in)

  mse = gs_mselt(w,s)

  ;; Open (Erode-Dilate)

  tmp = gs_padimage(tmp,w)
  tmp = erode(tmp,mse,/gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  tmp = gs_padimage(tmp,w)
  tmp = dilate(tmp,mse,/gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  ;; Close (Dilate-Erode)

  tmp = gs_padimage(tmp,w)
  tmp = dilate(tmp,mse,/gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  tmp = gs_padimage(tmp,w)
  tmp = erode(tmp, mse, /gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  ;; rescale output
  out = gs_discretize(tmp,/undo)

END
```

SOBELGRADIENT

```
PRO ga_sobel_grad, in, out
;+
:GA Project: Finds the absolute sobel gradient magnitude
;
;-

   k = [[1, 0, -1],$
       [2, 0, -2],$
       [1, 0, -1]]

     out = sqrt( gs_convolve(in,k)^2 + gs_convolve(in,transpose(k))^2)

END
```

CLOSE_OPEN

```
PRO  ga_close_open, in, out, w, s

  ;; Dilate - Erode - Erode - Dilate

  ;; scale input
  tmp = gs_discretize(in)

  mse = gs_mselt(w,s)

  ;; Close (Dilate-Erode)

  tmp = gs_padimage(tmp,w)
  tmp = dilate(tmp,mse,/gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  tmp = gs_padimage(tmp,w)
  tmp = erode(tmp, mse, /gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  ;; Open (Erode-Dilate)

  tmp = gs_padimage(tmp,w)
  tmp = erode(tmp,mse,/gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  tmp = gs_padimage(tmp,w)
  tmp = dilate(tmp,mse,/gray,/ulong)
  tmp = gs_padimage(tmp,w,/unpad)

  ;; rescale output
  out = gs_discretize(tmp,/undo)

END
```