Subject: Re: point inside polygon
Posted by davidf on Mon, 06 Mar 2000 08:00:00 GMT
View Forum Message <> Reply to Message

Klaus Scipal (kscipal@ipf.tuwien.ac.at) writes:

> the algorithm works fine until the point is lying on the polygon. in such a
> case the algorithm returns sometimes in and sometimes out. what i am looking
> for is an algorithm which discerns if a point is lying on the polygon and
> give only one answer either in or out but not one time in and the next time
> out.

Humm. Well, here is another one by Julie Greenwood and used
with her permission. I haven't used this one myself, but it
might be worth a try. It uses a nice coding style, which
always suggests to me that it will probably work. :-)

Cheers,

David

```
function IsPointInPolygon, xPts, yPts, XTest, YTest
;+
; NAME:
;  IsPointInPolygon
;
; PURPOSE:
;  Determine whether point (XTest,YTest) is within the boundary of
;   the polygon defined by vectors xPts & yPts.
;  Function returns 1 if point is within polygon, else returns 0
;  Use ArePointsInPolygon to test entire array at a time
;
; AUTHOR: Julie Greenwood (julieg@oceanweather.com)
;
; CATEGORY: Geometry
;
; CALLING SEQUENCE:
;  bResult = IsPointInPolygon (xPts, yPts, XTest, YTest)
;
; INPUTS:
;  xPts, yPts - vectors containing vertices of convex polygon in $
;   counterclockwise order.  See argument B to routine Triangulate.pro
;  xTest, yTest - point to test for within-ness
;
; Optional Inputs:   None
;
; OUTPUTS:
;  Function returns -1 if point is within polygon, else returns 0
```

```
;
; OPTIONAL OUTPUTS:  None
;
; KEYWORD Parameters: None
;
; COMMON BLOCKS:  None
;
; SIDE EFFECTS:   None
;
; RESTRICTIONS:
;  Polygon must be closed (first point is same as last)
;
; PROCEDURE:
;  See:  http://www.swin.edu.au/astronomy/pbourke/geometry/insidepoly /
;
; Consider a polygon made up of N vertices (xi,yi) where I ranges from
;  0 to N-1. The last vertex (xN,yN) is assumed to be the same as the
;  first vertex (x0,y0), that is, the polygon is closed.  To determine
;  the status of a point (xp,yp) consider a horizontal ray emanating
;  from (xp,yp) and to the right.  If the number of times this ray
;  intersects the line segments making up the polygon is even then the
;  point is outside the polygon. Whereas if the number of intersections
;  is odd then the point (xp,yp) lies inside the polygon.
;
; MODIFICATION HISTORY:
;  jgg - 2-Dec-1999 - Created
;-

npts = n_elements(xPts)
if (npts ne n_elements(yPts)) then begin
  print,' Error in IsPointInPolygon: vertex arrays must have same size.'
  return,PointIsIn
endif
if (npts lt 2) then begin
  print,' Error in IsPointInPolygon: polygon has less than 2 points.'
  return,PointIsIn
endif

PointIsIn = 0
for ia = 0,nPts-1 do begin
ib = (ia+1) mod nPts

sLine = xPts[ia] + $
  (yTest-yPts[ia]) * (xPts[ib]-xPts[ia]) / (yPts[ib]-yPts[ia])

bInRange = $
  (( (yPts[ia] le yTest) and (yTest lt yPts[ib]) ) or $
   ( (yPts[ib] le yTest) and (yTest lt yPts[ia]) ))
```

```
if ( bInRange and (xTest lt sLine) ) then begin

   PointIsIn = not PointIsIn

endif
endfor

return, PointIsIn

end

function ArePointsInPolygon, xPts, yPts, XTest, YTest
;+
; NAME:
;  ArePointsInPolygon
;
; PURPOSE:
;  Determine whether points (XTest,YTest) are within the boundary of
;   the polygon defined by vectors xPts & yPts.
;  Function returns array of same size and shape as XTest, containing
;   1 if point is within polygon, else 0
;  Use IsPointInPolygon to test one point at a time
;
; AUTHOR: Julie Greenwood (julieg@oceanweather.com)
;
; CATEGORY: Geometry
;
; CALLING SEQUENCE:
;  bResult = ArePointsInPolygon (xPts, yPts, XTest, YTest)
;
; INPUTS:
;  xPts, yPts - vectors containing vertices of convex polygon in $
;   counterclockwise order.  See argument B to routine Triangulate.pro
;  xTest, yTest - arrays of points to test for within-ness
;
; Optional Inputs:   None
;
; OUTPUTS:
;  Function returns array of same size and shape as XTest, containing
;   -1 if point is within polygon, else 0
;
; OPTIONAL OUTPUTS:  None
;
; KEYWORD Parameters: None
;
; COMMON BLOCKS:  None
;
```

```
; SIDE EFFECTS:   None
;
; RESTRICTIONS:
;  Polygon must be closed (first point is same as last)
;
; PROCEDURE:
;  See:  http://www.swin.edu.au/astronomy/pbourke/geometry/insidepoly /
;
; Consider a polygon made up of N vertices (xi,yi) where I ranges from
;  0 to N-1. The last vertex (xN,yN) is assumed to be the same as the
;  first vertex (x0,y0), that is, the polygon is closed.  To determine
;  the status of a point (xp,yp) consider a horizontal ray emanating
;  from (xp,yp) and to the right.  If the number of times this ray
;  intersects the line segments making up the polygon is even then the
;  point is outside the polygon. Whereas if the number of intersections
;  is odd then the point (xp,yp) lies inside the polygon.
;
; MODIFICATION HISTORY:
;  jgg - 3-Dec-1999 - Created
;-

nsizex = size(xTest)
nsizey = size(xTest)
if (total(nsizex ne nsizey)) then begin
  help, xTest, yTest
  print,' Error in ArePointsInPolygon: grid arrays must have same size.'
  return,PointsIn
endif
if (nsizex[0] ne 2) then begin
  help, xTest, yTest
  print,' Error in ArePointsInPolygon: grid arrays must have 2 dimensions'
  return,PointsIn
endif

npts = n_elements(xPts)
if (npts ne n_elements(yPts)) then begin
  help, xPts, yPts
  print,' Error in ArePointsInPolygon: vertex arrays must have same size.'
  return,PointsIn
endif
if (npts lt 2) then begin
  help, xPts, yPts
  print,' Error in ArePointsInPolygon: polygon has less than 2 points.'
  return,PointsIn
endif

PointsIn = LonArr(nsizex[1],nsizex[2])
for ia = 0,nPts-1 do begin
```

```
ib = (ia+1) mod nPts

sLine = xPts[ia] + $
  (yTest-yPts[ia]) * (xPts[ib]-xPts[ia]) / (yPts[ib]-yPts[ia])

bInRange = $
  (( (yPts[ia] le yTest) and (yTest lt yPts[ib]) ) or $
   ( (yPts[ib] le yTest) and (yTest lt yPts[ia]) ))

GotIt = where ( bInRange and (xTest lt sLine) , count)
if (count ne 0) then PointsIn[GotIt] = not PointsIn[GotIt]

endfor

return, PointsIn

end
```

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155
