
Subject: Re: Inner Hull

Posted by [Dick Jackson](#) on Fri, 18 May 2012 22:53:24 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi,

Gianguido has a good idea about using spheres, but to go with your idea of hulls a bit...

The convex hull of a set of points is well defined, but while an "inner hull" seems plausible, looking closer, there may not always be a unique solution. I worked with Gianguido's code a bit...

- assume the "centre" to be at [0,0] (important assumption)
- imagine all points to be on a "ring" between the points' min and max radii
- flip the ring inside-out so the inner edge switches with the outer edge
- use new points to find convex hull, then flip them back

With the random points, this often looks great, resulting in an actual convex shape. Sometimes it doesn't, and if it were important, you might look for any point that makes it concave, and start removing neighbouring points from the polygon until it is fully convex. (since there may be several ways to choose these, I think there is no unique "convex inner hull") For your purposes, if the points are well-behaved enough, the first shape may be sufficient.

To see the importance of getting the centre point correct, just append "- 0.5" to the calculation of x and rerun it.

Of course, in 3-D, you'll use QHull and its returned triangle descriptors, and the "Sphere" instead of "Polar" options to CV_Coord.

PRO InnerConvexHullTest

```
!P.Multi = [0, 2, 2]
```

```
theta=randomu(s,100)*2*!pi
x=cos(theta)+randomu(s,100)*.2-.1
y=sin(theta)+randomu(s,100)*.2-.1
plot, x, y, /iso, ps=1 ; , xr=[-1,1]*1.5, yr=[-1,1]*1.5, /xs, /ys
```

```
Triangulate, x, y, triangles, hull
Plots, [x[hull],x[hull[0]]], [y[hull],y[hull[0]]], $
color='00FF00'X ; cgcolor('green')
PlotS, [0], [0], PSym=1, SymSize=2, color='00FF00'X ; cgcolor('green')
```

```
;This dumb idea ain't gonna work :-(
;newX = 1/x
;newY = 1/y
```

```
;This idea might work :-)
```

```
thetaR = CV_Coord(From_Rect=[Transpose(x), Transpose(y)], /To_Polar)
```

```

minR = Min(thetaR[1, *], Max=maxR)
thetaR[1, 0] = minR+maxR-thetaR[1, *]
newXY = CV_Coord(From_Polar=thetaR, /To_Rect)
newX = newXY[0, *]
newY = newXY[1, *]

plot, newX,newY, /iso, ps=1,title='Plotting the inverses'
Triangulate, newX, newY, triangles, hull
Plots, [newX[hull],newX[hull[0]], [newY[hull],newY[hull[0]]],$
color='0000FF'X ; cgcolor('red')
PlotS, [0], [0], PSym=1, SymSize=2, color='0000FF'X ; cgcolor('red')

plot, x, y, /iso, ps=1, $
; xr=[-1,1]*1.5, yr=[-1,1]*1.5, /xs,/ys, $
title='These are the same guys in "regular" space.'
Plots, [x[hull],x[hull[0]], [y[hull],y[hull[0]]],$
color='0000FF'X ; cgcolor('red')
PlotS, [0], [0], PSym=1, SymSize=2, color='0000FF'X ; cgcolor('red')

```

END

On Friday, May 18, 2012 9:36:13 AM UTC-7, Gianguido Cianci wrote:

```

> Can't you simplify your problem?
>
> If you want the volume of a sphere that includes all points/excludes all points, can't you just
take the point that is farthest/closest to the center, and use that as a radius?
>
> This is the dum attempt I made to flip the circle inside out...
>
> theta=randomu(s,100)*2*!pi
> x=cos(theta)+randomu(s,100)*.2-.1
> y=sin(theta)+randomu(s,100)*.2-.1
> plot, x, y, /iso, ps=1, xr=[-1,1]*1.5, yr=[-1,1]*1.5, /xs, /ys
>
> Triangulate, x, y, triangles, hull
> Plots, [x[hull],x[hull[0]], [y[hull],y[hull[0]]],$
> color=cgcolor('green')
>
> ;This dumb idea ain't gonna work :-(
> plot, 1./x,1./y,ps=1,title='Plotting the inverses'
> Triangulate, 1./x, 1./y, triangles, hull
> Plots, [1./x[hull],1./x[hull[0]], [1./y[hull],1./y[hull[0]]],$
> color=cgcolor('red')
>
> plot, x, y, /iso, ps=1, xr=[-1,1]*1.5, yr=[-1,1]*1.5, /xs,$
> /ys,title='These are the same guys in "regular" space.'
> Plots, [x[hull],x[hull[0]], [y[hull],y[hull[0]]],$
> color=cgcolor('red')

```

>
>
> -- Gianguido
>
>
> On Friday, May 18, 2012 10:22:00 AM UTC-5, Ayla P wrote:
>> Hi all,
>>
>> I'm hoping someone might have some insight into this problem i'm
>> trying to solve. Hopefully I can make my questions coherent!
>>
>> The intent is pretty simple: I have a bunch of points that make up a
>> sphere, and I want to calculate the volume of the sphere.
>>
>> The points are random and aren't in a single line that makes an edge,
>> so I want to see how much the the volume of the sphere changes if I
>> make a hull using the outer-side of the "rim" (outermost pixels)
>> versus using the inner-side of the it (not all the interior pixels,
>> but those that basically make up the inner side of the rim). I've made
>> a convex hull around the outer pixels, but now I basically want to
>> make an inner hull.
>>
>> Would it be possible to place the inner pixels on the outside and the
>> outer pixels on the inside then just retrace a convex hull and then re-
>> place the pixels into their original positions? If so, does anyone
>> have a suggestion of how to do this? If not, does anyone have an
>> alternative suggestion for how to get the inner hull?
>>
>> Thanks so much for the help!
>>
>> Ayla
