Subject: Re: determining if a point is "inside" or "outside" a shape
Posted by Randall Frank on Fri, 22 Oct 1999 07:00:00 GMT
View Forum Message <> Reply to Message

I would suggest you read the Graphics FAQ on this issue and also check Graphics Gem (I think volume 1) for a more detailed explanation of this problem. The upshot is that there really are three core methods and many variants. In general, you can sum angles, sum signed areas or clip a line. There are good code examples of all these approaches on the net which can be coded into IDL very quickly. It also depends on how you intend to use the function. If, you are going to repeatedly test many points, you are better off using one of the sorted variants of the line clipping techniques. In general, the line clipping techniques are the fastest on the average, but have poor worst case performance without the sorting overhead. The angle sum is one of the slowest methods unless you can get creative and avoid the transcendentals (and you can). The area sum approach generally falls inbetween. In IDL code, I believe you can vectorize the latter with some setup overhead, making it the fastest for .pro code when testing multiple points with one point per call.

FWIW.

Job von Rango wrote:

> Ther is another analytic way to solve the problem:
>
> How to determine wether a point lies inside an arbitrary 2-dimensional
> polygon
> or not?
>
> IDEA:
> Look at the sum of all angles tended between all pairs of neighboured
> vertics
> the polygon and the given point.
>
> IN DETAIL:
> Assume we have the n vertices of polgons ordered at the positions:
> v_1, v_2, ... ,v_n
> The given point is denoted by p.
>
> Now connect the given point p with all n neighboured
> vertices, and get the n vectors beginning in p and ending
> in the vertices:
>
> vec_1 = vector(p_0, v_1)
> ...
> vec_n = vector(p_0, v_n)

>
> Now add all n angles tended between the 2 vectors vec_i and vec_(i+1):
>
> angle_i = angle(vec_i,vec_(i+1))
>
> and calculate the sum:
> n
> anglesum = sum angle_i
> i=1
>
> (Use vec_1 again for vec_(n+1) in the last angle_n)
>
> RESULT:
>
> _
> | 2*pi inside
> anglesum = -| if p is the polygon
> |_ 0 outside
>
> IMPORTANT:
> Take into account the correct sign of the angles
> and use the vertices in ascending order!
> Use e.g. the vector crossproduct:
>
> vec_i x vec_(i+1)
>
> in order to retrieve the absolut value |...| for the angle:
>
> |angle_i| = inv sin(|vec_i x vec_(i+1)|)
>
> The valid sign for angle_i is given by looking at the scalar product
> of the crossproduct from above and a fixed vector vec_perp, perpendicular
> to the area of the polygon:
>
> vec_perp * (vec_i x vec_(i+1))
> sign_i = ----------
> | vec_perp * (vec_i x vec_(i+1)) |
>
> Now we get the correct value for all angles:
>
> angle_i = sign_i * |angle_i|
>
> REMARK:
> Examine the case of a given point inside the polygon, but very near
> to the edge between two vertices v_i and v_(i+1). The contributing
> angle will be:
>
> angle_i = + pi - epsilon
>

> (where epsilon denotes a small positive angle.)
> If we shift p over the edge to the outside of the polygon
> (but very near again to the edge)
> the contributing angle will be:
>
> angle_i = - pi + epsilon'
>
> The switch of the sign is the reason for the discontinuity (2*pi <-> 0)
> of anglesum for points moving from inside to the outside of the polygon!
>
> ---------- --
> Job v. Rango Medizinische Klinik I
> Dipl.-Phys. D-52074 Aachen
> Pauwelsstrasse 30
> Tel. : 049 / (0)241 / 80-89832
> Fax : 049 / (0)241 / 88-88414
> Email: jran@pcserver.mk1.rwth-aachen.de

--
rjf.

Randall Frank | Email: rjfrank@llnl.gov
Lawrence Livermore National Laboratory | Office: B4525 R8019
P.O. Box 808, Mailstop:L-560 | Voice: (925) 423-9399
Livermore, CA 94550 | Fax: (925) 422-6287