C语言中实现点在多边形内算法 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/273/2021_2022_C_E8_AF_AD_E8_A8_80_E4_B8_AD_c97_273120.htm 本文是采用射线法判断点是否在多边形内的C语言程序。这是个C语言的小算法的实现程序，本来不想放到这里。可是，当我自己要实现这样一个算法的时候，想在网上找个现成的，考察下来竟然一个符合需要的也没有。我对自己大学读书时写的代码没有信心，所以，决定重新写一个，并把它放到这里，以飨读者。也增加一下BLOG的点击量。 首先定义点结构如下：以下是引用片段： /* Vertex structure */ typedef struct { double x, y. } vertex_t. 本算法里所指的多边形，是指由一系列点序列组成的封闭简单多边形。它的首尾点可以是或不是同一个点(不强制要求首尾点是同一个点)。这样的多边形可以是任意形状的，包括多条边在一条绝对直线上。因此，定义多边形结构如下：以下是引用片段： /* Vertex list structure  polygon */ typedef struct { int num_vertices. /* Number of vertices in list */ vertex_t *vertex. /* Vertex array pointer */ } vertexlist_t. 为加快判别速度，首先计算多边形的外包矩形(rect_t)，判断点是否落在外包矩形内，只有满足落在外包矩形内的条件的点，才进入下一步的计算。为此，引入外包矩形结构rect_t和求点集合的外包矩形内的方法vertices_get_extent，代码如下：以下是引用片段： /* bounding rectangle type */ typedef struct { double min_x, min_y, max_x, max_y. } rect_t. /* gets extent of vertices */ void vertices_get_extent (const vertex_t* vl, int np, /* in vertices */ rect_t* rc /* out extent*/ ) { int i. if (np > 0){ rc->min_x = rc->max_x =

vl[0].x. rc->min_y = rc->max_y = vl[0].y. }else{ rc->min_x = rc->min_y = rc->max_x = rc->max_y = 0. /* =0 ? no vertices at all */ } for(i=1. i { if(vl[i].x min_x) rc->min_x = vl[i].x. if(vl[i].y min_y) rc->min_y = vl[i].y. if(vl[i].x > rc->max_x) rc->max_x = vl[i].x. if(vl[i].y > rc->max_y) rc->max_y = vl[i].y. } } 当点满足落在多边形外包矩形内的条件，要进一步判断点(v)是否在多边形(vl：np)内。本程序采用射线法，由待测试点(v)水平引出一条射线B(v，w)，计算B与vl边线的交点数目，记为c，根据奇内偶外原则(c为奇数说明v在vl内，否则v不在vl内)判断点是否在多边形内。