软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(9):2481−2488 [doi: 10.3724/SP.J.1001.2012.04087]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于网格中心点的点在多边形内的高效判定*

李 静+, 王文成

(中国科学院 软件研究所 计算机科学国家重点实验室,北京 100190)

Point-in-Polygon Test Method Based on Center Points of Grid

LI Jing+, WANG Wen-Cheng

(State Key Laboratory of Computer Science, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

+ Corresponding author: E-mail: lij@ios.ac.cn

Li J, Wang WC. Point-in-Polygon test method based on center points of grid. *Journal of Software*, 2012,23(9): 2481−2488 (in Chinese). http://www.jos.org.cn/1000-9825/4087.htm

Abstract: The paper proposes an efficient point-in-polygon test method based on uniform grids. In preprocessing, it constructs uniform grids and processes the center points in a sequence to determine whether they are inside the polygon, when the center points with known inclusion properties are used to quickly determine their adjacent center points. When performing an inclusion query, it first finds the cell that contains the query point and then produces a line segment from the query point to the cell's center point and counts the edges crossed by the line segment for determination. As both the preprocessing and the inclusion tests make use of known information to determine the center points or query points, the computation can run locally for acceleration and the new method can run much faster than existing methods. In most cases, it can reduce the preprocessing time complexity from $O(N^{3/2})$ to $O(N)$ for the methods based on grids, where N is the number of the polygon edges. Meanwhile, the new method can efficiently treat the non-manifold polygons with self-intersection or overlapping edges. Experimental results show that compared with other similar methods based on uniform grids, the new algorithm can speed up the preprocessing by several times and the query computation by one or several dozens of times, and can move faster than the method by convex decomposition, which has the lowest complexity $O(\log N)$ for point-in- polygon tests.

Key words: polygon; point-in-polygon test; grid; center point

摘 要: 提出一种基于均匀网格的点在多边形内的高效判定算法.它首先建立均匀网格,并从左至右依次计算每个网格单元中心点的位置属性.每个单元中心点的位置属性直接依据其左侧邻接单元已知位置属性的中心点快速获得.在判定点的位置时,确定被测点所在单元,并依据该单元中心点的位置属性判定被测点的位置属性.由于预处理和判定时均利用邻近点的已知位置属性来确定未知点位置属性,可以很好地进行局部化的计算.因此,新方法比现有方法快很多,并且其预处理时间复杂度也由同类网格算法的 $O(N^{3/2})$下降为 $O(N)$.同时,新方法可以统一处理含有自相交及重叠边的非流形多边形.实验结果表明,相比于其他基于均匀网格的方法,新方法可将预处理的速度提高几倍,将判断计算的速度提高十几到几十倍.其速度甚至优于具有该问题最低判定计算时间复杂度 $O(\log N)$的基于凸剖分的判定算法.

* 基金项目: 国家自然科学基金(60873182, 60773026, 60833007)
收稿时间: 2010-05-31; 修改时间: 2010-08-27; 定稿时间: 2011-07-01

关键词: 多边形;点包容性检测;网格;中心点
中图法分类号: TP391 文献标识码: A

多边形的点包容性检测(point-in-polygon test)是计算几何中最基本的问题之一,它在计算机图形学、地理信息系统(GIS)等领域有广泛的应用[1].该问题的描述是:给定多边形 P 和任意点 Q,判定点 Q 是否位于多边形 P 内.尽管目前已有大量的多边形点包容性检测算法,但寻找更加高效的解决方案仍然是重要的研究内容.这是因为该判定问题是很多算法的基础,如多边形裁剪和光线跟踪等,其效率在很大程度上影响着上层应用的效率.此外,应用的发展又为作为基础的判定算法提出了新的目标,如 GIS 中对洪水泛滥区域的动态监测等.

关于多边形的点包容性判断的工作很多,基本上可分为两类:一类是依靠计算某些参数得到判定结果.例如,经典的射线法(ray-crossing)[2,3]就是通过从被测点向某方向作一条射线,计算该射线与多边形的边的交点个数来进行判定.若有奇数个交点,则该点位于多边形内;否则位于多边形外.这种方法简单直观,可处理任意多边形,通常被用作与其他算法对比的基准算法.基于角度之和的算法[4,5]通过计算被测点与多边形所有边构成的夹角角度之和进行判断.若该值为 360°,点在多边形内;若为 0°,则点在多边形外.基于三角形的算法[6,7]则将一定点与多边形的每条边连接形成多个三角形,计算被测点相对于这些三角形的符号之和.若该值为 1,则点在多边形内.这类算法的实现较为简单,无需预处理和存储辅助结构;但需要进行逐边操作,计算时间复杂度均为 $O(N)$.

另一类则是通过将多边形分解为一些简单的几何元素并进行有效组织,或者通过建立高效的空间划分结构来加速判定操作.如梯形法[1,8]、网格法[9,10]、凸剖分法[11]、分层法[12]、层次三角形[13]等.这类算法非常适合于对同一多边形进行多次检测的应用.

凸剖分法[11]先将多边形凸剖分,并以树结构管理所得的凸多边形,判定时通过树结构快速查找可能包含被测点的凸多边形,然后以高效的凸多边形点检测算法确定被测点是否在多边形内.该方法的判定计算时间复杂度为 $O(\log N)$,是目前多边形点检测问题最低的判定计算时间复杂度.当剖分得到的凸边形个数较少时,该算法具有很高的效率.但随着凹点数目的增多,凸多边形数目也随之增多,检测时间增长.此外,凸剖分法的预处理时间也相对较长.

网格结构简单,创建迅速,在实际中得到了较多的应用.Zalik 和 Kolingerova[9]提出的 CBCA(cell-based containment algorithm)方法在预处理时将多边形的包围盒细分为均匀网格,为每个网格单元记录所有经过它的多边形的边,并将所有空单元划分为位于多边形外和多边形内两类.在点检测时,假设 Q 为被测点,$C_{[Q]}$为包含 Q 的单元.则若 $C_{[Q]}$为空单元,Q 的位置属性可由 $C_{[Q]}$直接获得;否则,需要先找到 $C_{[Q]}$内距离 Q 最近的边(或顶点),然后根据 Q 与该最近边的位置关系(或者是最近顶点的凹凸性)确定 Q 点的位置属性.该方法的期望预处理时间复杂度为 $O(N^{3/2})$,空间开销为 $O(N)$,检测时间复杂度为 $O(N^{1/2})$.但是,当最近点位于 $C_{[Q]}$之外,或者多边形顶点的两条邻边几乎共线时,CBCA 算法会出现错误.对此,Yang 等人[10]提出了 quasi-closest point 的概念,用以表示受约束条件限制的最近点,并根据 quasi-closest point 获得被测点的属性.这样,避免了当一个顶点的两条邻边几乎共线时,使用顶点的凹凸性判定而导致的错误.

尽管网格法对于某些被测点可以达到 $O(1)$的判定速度,但现有方法由于受制于某些因素而使预处理和判定速度提高有限.这两种算法[9,10]在预处理时均使用 flood-fill 算法[2]确定空单元在多边形外还是内.在扩张过程中,对于每个需要处理的空单元来说,其所有的邻居单元均需被访问一遍,以确定是否已确定位置属性以及是否需要进一步扩张.这样,大量的网格单元就会被重复访问.而在判定种子单元的位置属性时,则需要访问更多的单元.以 CBCA[9]为例,它从种子单元中分别向上、下、左、右射出 4 条射线,直至遇到边界单元或者已确定位置属性的空单元为止,选择穿越最少单元的射线并计算其与多边形的交点个数.这样,需要访问的单元总数将大大超过总单元数,从而使预处理的时间较长.此外,在判定落入非空单元的被测点时,不但需要求解点到线段的最短距离,而且还要判定点与线段的位置关系,这也使得操作变得复杂不利于提高速度.

本文提出的新算法仍然基于均匀网格结构,但与以往算法不同的是,它使用一些点的已知位置属性信息来加快被测点的判定.也就是在预处理时计算并记录每个网格中心点的位置属性(多边形内/多边形外),在判定时

通过被测点与其所在单元中心点的关系来判定该点的位置属性.高效的局部计算使预处理时间大幅下降,新方法的期望预处理时间复杂度仅为 $O(N)$.实验结果表明,新算法所需的预处理时间少,且具有很高的判定效率,比以往类似工作快几倍甚至几十倍.在处理边数较多的复杂多边形时,其速度甚至优于具有该问题最低判定计算时间复杂度 $O(\log N)$的基于凸剖分的方法[11].

1 网格中心点算法

网格中心点算法基于这样一个观察结论:给定多边形 P、一已知位置属性(P 外/P 内)的点 M 和被测点 Q,则若线段 MQ 与 P 的边的交点个数为偶数,则 Q 的位置属性与 M 的位置属性相同;否则相反.这个结论可以很容易地由射线法推出.

在创建网格结构后,若 M 与 Q 位于同一单元 C 内,则 MQ 仅与单元 C 内的边进行相交测试即可判定 Q 的位置.由于每个网格单元中仅包含少量的多边形的边,因而网格中心点算法能够以很低的代价获得判定结果.M 点可以是单元 C 中的任一点,但为了计算方便且充分利用网格的规则性,我们选择单元的中心点作为 M.因此,网格中心点算法首先为被测多边形创建均匀网格,然后计算并存储每个网格单元的中心点位置属性.在判定时,找到被测点所在的单元,计算被测点与该单元中心点的连线与单元内所有边的相交数,若为偶数则被测点与中心点的位置属性相同;否则相反.下面,我们首先在第 1.1 节说明预处理过程,然后在第 1.2 节详述点包容性判定算法.

1.1 预处理

预处理过程包括创建网格和计算网格单元中心点属性两个部分.

创建网格时需要首先确定网格分辨率,它是决定网格创建效率和点检测效率的关键参数.从理论上讲,网格分辨率越高,判定时间越短.也就是总能找到一种足够细致的划分,使每个网格单元或者不包含或者仅包含一条多边形的边,从而以 $O(1)$速度得到判定结果.但网格分辨率越高,所需的预处理时间越多,占用的空间越多.而在实际应用中,需要对预处理时间、判定时间和存储空间进行综合考虑.基于此,我们采用与文献[9,10]相同的经验公式来确定网格分辨率,即设网格的宽长比为λ,则 X 和 Y 方向上单元数分别为

$$M_x = 2\left\lfloor \lambda n^{1/2} \right\rfloor, M_y = 2\left\lfloor (1/\lambda) n^{1/2} \right\rfloor.$$

在确定分辨率后,需要为每个单元计算落入其中的边,并将边指针加入相关单元.为了快速实现该过程,我们使用 DDA 算法[14],仅以加法递增计算各个边占据的网格单元.

在创建网格之后,还需要计算每个单元中心点的位置属性,也就是判定这些中心点位于多边形之内还是之外.依据前述观察结论,我们可以进行快速判定,即依据已做判定的中心点位置计算相邻的中心点属性.具体方法是:逐行自左至右遍历各个网格单元.对每个网格单元,首先从该单元中心点到左侧相邻单元中心点做一条连线 S.如果是最左侧的单元,则与左侧包围盒外部的某点(称为最左点)连线.由于位于网格外,最左点的位置属性一定是位于多边形外.然后,遍历本单元和左侧单元中的所有边,计算出与 S 有相交边的个数 r.由于算法是自左至右遍历,因此左侧单元的中心点(或最左点)的位置属性已经判断完毕.由此,若 r 为偶数,则该单元中心点位置属性与其左侧单元的中心点(或最左点)相同;否则相反.

图 1 展示了计算网格中心点位置的一个具体实例.左图为被测多边形和网格结构.其中,灰色点为网格中心点或者每行网格的最左点.有向线段为为计算中心点属性所作的连线.算法从单元 $C_{[0,0]}$开始,依次计算单元 $C_{[1,0]},C_{[2,0]},\ldots,C_{[5,0]}$的中心点位置.然后处理下一行单元,即从单元 $C_{[0,1]}$开始,向右直到单元 $C_{[5,1]}$.依此类推.在计算时,以单元 $C_{[0,2]}$为例,由于是最左侧单元,因此从该行最左点至单元 $C_{[0,2]}$的中心点做辅助连线,该线段与多边形的边仅有一个交点,因此单元 $C_{[0,2]}$的中心点位置属性与最左点相反,为多边形内.下一步处理单元 $C_{[1,2]}$,从单元 $C_{[0,2]}$的中心点至单元 $C_{[1,2]}$的中心点做辅助连线,该线段与多边形的边有 3 个交点,因此单元 $C_{[1,2]}$的中心点属性与单元 $C_{[0,2]}$的相反,为多边形外.图 1 右图展示了计算后的结果,其中,位于多边形外的中心点以空心点表示,位于多边形之内的中心点以实心点表示.

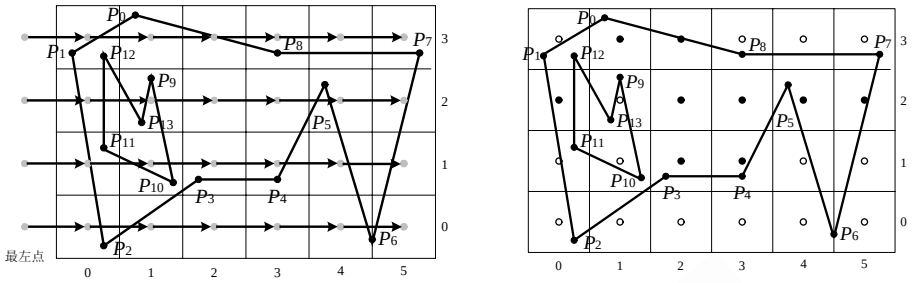


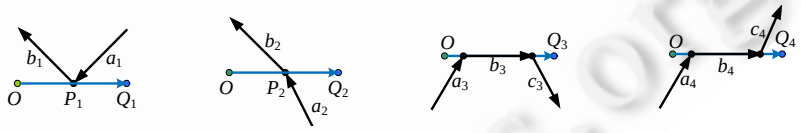
Fig.1 An example of determining the center points

图 1 计算网格单元中心点位置属性实例图

上述算法的主要基本操作是判断两条线段是否相交,这可以通过判定点与直线的相对位置关系实现.设有两条线段 e_1 和 e_2,若 e_1 的两个顶点分别位于 e_2 的两侧,并且 e_2 的两个顶点分别位于 e_1 的两侧,则认为 e_1 与 e_2 有交;否则,认为 e_1 与 e_2 无交.

判断一个点 Q 与一条边 e 关系的方法是:设边 e 的两个顶点分别为 P_0 和 P_1,待测点为 Q,计算向量 $\overrightarrow{P_0Q}$ 与向量 $\overrightarrow{P_0P_1}$ 叉积的 Z 值.若该值大于 0,则认为 Q 在 e 的右侧;否则,认为 Q 在 e 左侧.如此得到的点与边的关系分为两种,即左侧与右侧.如果点与边共线,则强制认为其位于直线的某一侧(如设为左侧).这样做的优点是避免了对奇异情况的特殊处理,它在各种奇异情况下均能得到正确的结果.下面举出几个例子作为说明,其他奇异情况依此类推.

设沿多边形的边行走时,左手侧为多边形内,右手侧为多边形外.若被测点与网格中心点的连线穿越多边形的顶点,则可能出现如图 2(a)、图 2(b)所示的情况.其中,O 为一已知位置属性的参考点,不失一般性,我们假定它位于多边形内.在图 2(a)中,按照强制设定规则,相交点 P_1 位于 OQ_1 的左侧,而多边形的边 a_1 和 b_1 的另一端也在 OQ_1 的左侧,因此 OQ_1 与 a_1 和 b_1 均无交,Q_1 与 O 的位置属性相同,皆为内部.结论正确.图 2(b)中,对于 Q_2,相交点 P_2 的位置被强制设为 OQ_2 的左侧,那么 OQ_2 与 b_2 无交,与 a_2 有交,相交数量为 1.则 Q_2 与 O 的位置属性相异,即 Q_2 位于多边形外,结论正确.若被测点与网格中心点的连线与多边形的边共线,则可能出现图 2(c)、图 2(d)所示的情况.在图 2(c)中,多边形的边 b_3 与 OQ_3 重合,按照强制设定规则,边 b_3 位于 OQ_3 的左侧,与 OQ_3 无交.OQ_3 分别与 a_3 和 c_3 相交,相交个数为 2.因此,Q_3 与 O 的属性相同,均为多边形内.而在图 2(d)中,b_4 与 OQ_4 重合.判定时,b_4 被判定为位于 OQ_4 的左侧,OQ_4 仅与 a_4 相交.因此,Q_4 与 O 的属性相反,位于多边形外.结论均正确.



(a) (b) (c) (d)

Fig.2 Examples of singular cases

图 2 奇异情况示例图

根据上述方法,待测点仅能得到两种属性,即多边形外和多边形内,不会获得位于边上的属性.但在预处理阶段,我们仅是要为每个单元取得一个已知位置的参考点,因此只要断定内外属性即可,无需断定是否位于边界上,因此不会对算法造成影响.

由于在判定每个中心点位置时,计算被局限在两个相邻单元的局部区域内,计算规模被大大降低,并且已得到判定的中心点的属性能被快速传递(相同或者相反)到相邻的中心点,因此中心点的计算能够快速完成.实验结果也证实了这一点,详见第 3 节.

1.2 判定算法

在建立网格结构并记录单元中心点位置属性后,就可以借助它们快速确定被测点的位置.方法是:首先,找到待测点 Q 对应的网格单元 $C_{[i,j]}$,其中,i,j 分别为单元 X,Y 方向上的坐标.然后,做一条从 Q 到网格单元 $C_{[i,j]}$的中心点 $M_{[i,j]}$的连线 $QM_{[i,j]}$(称为测试边).遍历 $C_{[i,j]}$中所有的边,记录与 $QM_{[i,j]}$有相交的边的个数.如果相交边数为偶数,则待测点 Q 与单元中心点 $M_{[i,j]}$的位置属性相同;否则,待测点 Q 与 $M_{[i,j]}$的位置属性相反.

以图 3 为例,图中 Q_1,Q_2,Q_3 和 Q_4 为被测点.首先,我们定位 Q_1 和 Q_2 在单元 $C_{[1,2]}$中,Q_3 和 Q_4 在单元 $C_{[3,1]}$中.然后,分别将 Q_1,Q_2 与单元 $C_{[1,2]}$的中心点 $M_{[1,2]}$相连;Q_3,Q_4 与单元 $C_{[3,1]}$的中心点 $M_{[3,1]}$相连.$Q_1M_{[1,2]},Q_2M_{[1,2]}$与单元 $C_{[1,2]}$中边的交点数分别为 2 和 1,且 $M_{[1,2]}$位于多边形外,因此,Q_1 与 $M_{[1,2]}$的属性相同,为多边形外;Q_2 与 $M_{[1,2]}$的属性相反,为多边形内.同理,$Q_3M_{[3,1]},Q_4M_{[3,1]}$与单元 $C_{[3,1]}$中边的交点数分别为 0 和 1,且 $M_{[3,1]}$位于多边形内,因此,Q_3 与 $M_{[3,1]}$的属性相同,为多边形内;Q_4 与 $M_{[3,1]}$的属性相反,为多边形外.

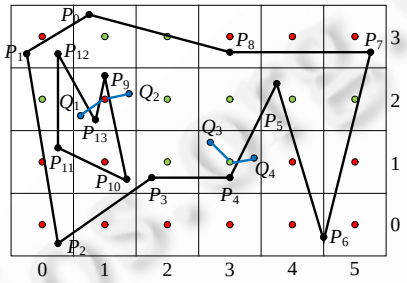
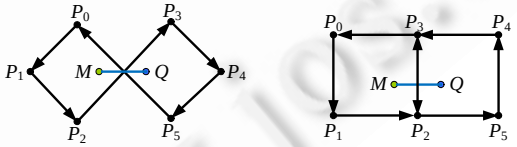


Fig.3 Examples of inclusion test by our method

图 3 点包容性判定实例图

判定算法中的基本操作仍是判定两条线段是否相交,判定方法与预处理时使用的相同.需要指出的是,在某些应用中,需要断定被测点是否位于多边形的边界上.对此,判定两条线段是否相交的过程应稍作修改,使之能够返回位于边界上的结果.即当判定测试边 QM 和多边形的边 P_0P_1 是否相交时,若向量 $\overrightarrow{P_0Q}$ 与向量 $\overrightarrow{P_0P_1}$ 叉积的 Z 值为 0(在实际的数值计算时,可设一个很小的值 ε,以其值的绝对值小于 ε 作为等于 0 的判定),并且 P_0 和 P_1 位于 QM 的两侧时,Q 位于多边形边界上.得到此结果后,判定算法无需后续计算,立即返回.

由于检测时仅需要求线段相交点个数,因此网格中心点算法对于自相交、重叠边等非流形情况能够等同地处理.以图 4 为例,图 4(a)中,多边形的边 P_5P_0 和 P_2P_3 自相交,中心点 M 位于多边形内,测试边 QM 与 P_5P_0 和 P_2P_3 均相交,交点数为偶数,Q 位于多边形内.在图 4(b)中,边 P_2P_3 和 P_3P_2 重叠,中心点 M 位于多边形内,测试边 QM 与 P_2P_3 和 P_3P_2 均相交,交点数为偶数,Q 位于多边形内.



(a) Self-Intersection (b) Overlapping edge

(a) 自相交 (b) 重叠边

Fig.4 Treatment of the non-manifold polygons with self-intersection or overlapping edge

图 4 对自相交、重叠边等非流形情况的等同处理

2 复杂度分析

根据文献[9]中对 CBCA 方法的分析,网格单元数的复杂度为 $O(N)$,空间复杂度为 $O(N)$.由于本文方法使用

与其相同的公式确定网格分辨率,且仅需要为每个单元增加一个 bit 位记录单元中心点的状态,因此新方法的复杂度仍为 $O(N)$.此外,新算法与 CBCA 具有相同的点包容性判定时间复杂度.这是因为若被测点位于空单元内,由于它与该单元中心点具有相同的属性,因此可在 $O(1)$时间内得到判定结果.若被测点位于非空单元内,则需要与该单元内的所有多边形的边进行求交测试.根据文献[9]中的分析,平均每个非空单元中有 $N^{1/2}$ 条边,因此点包容性判定操作可在 $O(N^{1/2})$时间内完成.对于预处理操作,新算法的期望时间复杂度为 $O(N)$,分析如下.

新算法的预处理过程分为 3 步:

(1) 创建均匀网格.如前所述,由于我们使用与文献[9]相同的公式确定网格单元数,因此根据文献[9]中的分析,这步的时间复杂度为 $O(N)$;

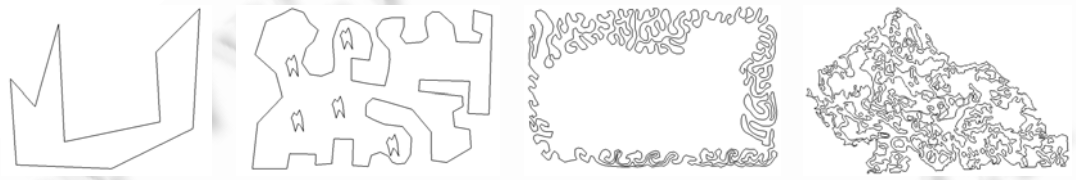
(2) 将所有多边形的边加入到对应网格单元中.根据文献[9],这步的复杂度也为 $O(N)$;

(3) 确定所有网格单元中心点的位置属性.新算法确定每个中心点的位置属性,至多需要检测两个单元内的边;又根据文献[9]中的分析,平均每个单元包含一条多边形的边.因此,这步的时间复杂度也为 $O(N)$.

综上,新算法的总期望预处理时间复杂度为 $O(N)$.

3 实验与分析

为了验证新算法的性能,我们在一台配置有 Intel(R) Core(TM)2 2.83GHz CPU,4GB RAM 的台式机上与其他算法进行了对比实验.用于对比的算法分别来自于 CBCA[9]、QCPM[10]和凸剖分法[11].其中,CBCA 和 QCPM 的数据直接引自文献[10],其实验中使用的计算机配置为 2.93GHz CPU,512MB RAM,优于我们实验的机器.凸剖分法为三叉树实现.被测多边形均来自文献[9],如图 5 所示.被测点个数均为 1 000 000,均匀分布在比网格包围盒稍大的范围内.



(a) pol10 (b) pol100 (c) pol1249 (d) pol28000

Fig.5 Polygons used in our experiments

图 5 实验使用的被测多边形

表 1 列出了实验结果,包括各种算法的预处理时间、1 000 000 个点的判定时间、预处理和判定的总时间(单位均为秒)以及新方法相对于这些对比方法的计算加速率.在此,各种网格方法都是基于同样的网格划分分辨率进行的.

由表 1 可知,与凸剖分法相比较,除多边形 pol10 外,网格中心点算法的效率均优于凸剖分算法.这是因为在多边形边数较少、凹点较少的情况下,凸剖分算法需要处理的子部分较少,因此会稍快.而当多边形的复杂程度升高时,网格中心点算法则优于凸剖分法,且多边形的边数越多,凹点数越多,网格中心点算法的优势越明显.与其他网格类算法相比,中心点算法的优势明显,在预处理时可节省数倍的时间,而在判定计算时可提高一个数量级.从表 1 中可知,新方法的总时间开销很少,这很有利于动态情况下的点在多边形内的判定计算,比如确定洪水泛滥时某些位置点是否被洪水淹没,其中,表示洪水淹没区域的多边形是动态改变的.

此外,由于新算法除存储基本网格结构外,仅需要存储中心点属性,且这在实际中可以仅用一个比特位实现,因此额外附加的空间很少.实验显示,用于存储中心点的空间平均仅占总空间开销的 6.5%.

Table 1 Results of comparison experiments between our new method and other point-in-polygon algorithms

表 1 网格中心点算法与其他算法的对比实验数据

多边形	边数	分辨率	算法	预处理时间(加速率 r_p)#	判定时间(加速率 r_d)^	总时间(加速率 r_t)$
pol10	10	—	凸剖分	0.000 023 6 (1.15)	0.043 39 (−0.05)	0.043 4136 (−0.05)
		6×8	CBCA	0.000 055 (4.00)	2.555 309 (55.13)	2.555 364 (55.12)
			QCPM	0.000 044 (3.00)	0.972 392 (20.36)	0.972 436 (20.36)
			新算法	0.000 011	0.045 521	0.045 532
pol100	149	—	凸剖分	0.000 39 (2.71)	0.087 58 (1.10)	0.087 97 (1.10)
		18×38	CBCA	0.000 645 (5.14)	1.839 879 (43.06)	1.840 524 (42.97)
			QCPM	0.000 414 (2.94)	0.801 133 (18.19)	0.801 547 (18.15)
			新算法	0.000 105	0.041 757	0.041 863
pol1249	1249	—	凸剖分	0.003 02 (4.04)	0.16 (2.79)	0.163 02 (2.81)
		46×100	CBCA	0.004 844 (7.09)	1.707 092 (39.46)	1.711 936 (39.01)
			QCPM	0.002 827 (3.72)	0.740 552 (16.55)	0.743 379 (16.37)
			新算法	0.000 599	0.042 192	0.042 791
pol28000	28 000	—	凸剖分	0.13 (25.97)	0.18 (0.94)	0.31 (2.17)
		100×100	CBCA	0.042 282 (7.77)	7.483 888 (79.47)	7.526 17 (75.94)
			QCPM	0.024 614 (4.11)	1.205 676 (11.96)	1.230 29 (11.58)
			新算法	0.004 820	0.093 002	0.097 823

$r_p=(T_{p\text{-}old}-T_{p\text{-}new})/T_{p\text{-}new}$, $T_{p\text{-}old}$ 是对比方法预处理的时间,$T_{p\text{-}new}$ 是新方法的预处理时间.

^ $r_d=(T_{d\text{-}old}-T_{d\text{-}new})/T_{d\text{-}new}$, $T_{d\text{-}old}$ 是对比方法判定计算的时间,$T_{d\text{-}new}$ 是新方法判定计算的时间.

$ $r_t=(T_{t\text{-}old}-T_{t\text{-}new})/T_{t\text{-}new}$, $T_{t\text{-}old}$ 是对比方法的总时间开销,$T_{t\text{-}new}$ 是新方法的总时间开销.

4 结 语

本文提出一种判定点是否位于多边形内的新方法.它基于均匀网格结构,并通过邻近待测点的已知位置属性(多边形外/多边形内)的点来快速判定待测点的属性.它首先在预处理时计算并存储每个网格单元中心点的位置属性,然后在判定时,连接被测点与其所在单元的中心点,计算该线段与此单元中的多边形的边的交点个数,根据交点数的奇偶性判定被测点是否位于多边形内.在预处理与判定计算时,新方法能够很好地利用邻近点的已知位置属性,因此能有效地进行局部化计算,大幅降低了计算开销.实验结果表明,相比于类似的基于均匀网格划分的方法,新方法能将预处理速度提高数倍,而将判定时间提高一个数量级;而且,新方法的空间需求也不大,只是为记录每个中心点的属性增加一个比特.特别是新方法的预处理与判定计算的总时间开销很小,非常有利于处理动态的情况.

References:

[1] de Berg M, Cheong O, van Kreveld M, Overmars M. Computational Geometry: Algorithms and Applications. 3rd ed., Berlin, Heidelberg: Springer-Verlag, 2008. [doi: 10.1007/978-3-540-77974-2]

[2] Foley JD, van Dam A, Feiner SK, Hughes JF. Computer Graphics—Principles and Practice. 2nd ed., Reading: Addison-Wesley, 1990.

[3] Haines E. Point in polygon strategies. In: Heckbert P, ed. Proc. of the Graphics Gems IV. New York: Academic Press, 1994. 24−46.

[4] Preparata FP, Shamos MI. Computational Geometry: An Introduction. 2nd ed., New York: Springer-Verlag, 1985.

[5] Hormann K, Agathos A. The point in polygon problem for arbitrary polygons. Computational Geometry: Theory and Applications, 2001,20(3):131−144. [doi: 10.1016/S0925-7721(01)00012-8]

[6] Feito FR, Torres JC, Urena A. Orientation, simplicity, and inclusion test for planar polygons. Computers & Graphics, 1995,19(4): 595−600. [doi: 10.1016/0097-8493(95)00037-D]

[7] Jiménez JJ, Feito FR, Segura RJ. Robust and optimized algorithms for the point-in-polygon inclusion test without pre-processing. Computer Graphics Forum, 2009,28(8):2264−2274. [doi: 10.1111/j.1467-8659.2009.01481.x]

[8] Zalik B, Clapworthy GJ. A universal trapezoidation algorithm for planar polygons. Computers & Graphics, 1999,23(3):353−363. [doi: 10.1016/S0097-8493(99)00044-8]

[9] Zalik B, Kolingerova I. A cell-based point-in-polygon algorithm suitable for large sets of points. Computers & Geosciences, 2001, 27(10):1135−1145. [doi: 10.1016/S0098-3004(01)00037-1]

[10] Yang S, Yong JH, Sun J, Gu H, Paul JC. A point-in-polygon method based on a quasi-closest point. Computers & Geosciences, 2010,36(2):205−213. [doi: 10.1016/j.cageo.2009.06.008]

[11] Li J, Wang WC, Wu EH. Point-in-Polygon tests by convex decomposition. Computers & Graphics, 2007,31(4):636−648. [doi: 10.1016/j.cag.2007.03.002]

[12] Wang WC, Li J, Wu EH. 2D point-in-polygon test by classifying edges into layers. Computers & Graphics, 2005,29(3):427−439. [doi: 10.1016/j.cag.2005.03.001]

[13] Jiménez JJ, Feito FR, Segura RJ. A new hierarchical triangle-based point-in-polygon data structure. Computers & Geosciences, 2009,35(9):1843−1853. [doi: 10.1016/j.cageo.2008.09.013]

[14] Cleary JC, Wyvill G. Analysis of an algorithm for fast ray tracing using uniform space subdivision. The Visual Computer, 1988, 4(2):65−83. [doi: 10.1007/BF01905559]

李静(1972－),女,北京市人,博士,助理研究员,CCF 会员,主要研究领域为计算机图形学,计算几何.

王文成(1967－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为可视化,虚拟现实,计算机图形学.