计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(8):170−175 [doi: 10.15888/j.cnki.csa.006437]
©中国科学院软件研究所版权所有.
E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

任意多边形窗口的圆裁剪算法①

杨　琴[1], 李　宁[2], 王亮亮[1]

[1](新疆师范高等专科学校, 乌鲁木齐 830043)
[2](北京邮电大学世纪学院 移动媒体与文化计算北京市重点实验室, 北京 102101)

摘　要: 针对任意多边形窗口内圆的裁剪问题, 本文提出一种更加全面、有效的裁剪算法. 该方法提出借助 *x*-扫描线算法来判断圆和多边形窗口的位置关系, 排除圆完全在窗口内或者窗口外的情况; 针对多边形窗口和圆相交的情况, 按照逆时针方向依次求出多边形各边与圆的交点; 最终, 通过判断两点间的关系, 决定两点之间画线还是画弧, 完成圆的裁剪. 实验结果表明, 该方法能够有效全面的完成多边形窗口的圆裁剪.

关键词: 多边形窗口; 圆; 裁剪; 位置关系

引用格式: 杨琴,李宁,王亮亮.任意多边形窗口的圆裁剪算法.计算机系统应用,2018,27(8):170–175. http://www.c-s-a.org.cn/1003-3254/6437.html

Algorithm for Circle Clipping Based on Arbitrary Polygon Window

YANG Qin[1], LI Ning[2], WANG Liang-Liang[1]

[1](Xinjiang Teacher's College, Urumqi 830043, China)
[2](Mobile Media and Culture Computing Key Laboratory of Beijing, Century College, Beijing University of Posts and Telecommunications, Beijing 102101, China)

Abstract: For the problem of the circle clipping against arbitrary polygon window, the more comprehensive and effective clipping algorithm is proposed in this study. First, according to *x*-scan line algorithm, the spatial relationship between the circle and the polygon window is determined. Next, for the case of the polygon window and the circle intersection, the intersect points of the circle and each side of the polygon window are calculated in the counterclockwise direction and sorted correctly. At last, according to the relationship between two points, determining to draw a line or a circle arc. The whole circle clipping is obtained. The result expresses that the algorithm can be comprehensive and effective to complete circle clipping.

Key words: polygon window; circle; clipping; spatial relationship

1　引言

在日常生活中, 随着计算机动画、人机交互以及虚拟现实等技术的不断融入, 人们对计算机图形学技术提出了更高要求. 计算机图形学研究的内容十分广泛, 如基本图形生成、裁剪、三维几何造型、真实感图形的显示、交互技术等等. 裁剪作为计算机图形学的基础操作, 也是图像处理、模式识别等问题的基础.

在实际应用中, 裁剪窗口可能是各种形状, 比较常用的裁剪窗口为矩形或者多边形. 目前, 对裁剪算法的研究工作主要包括: 矩形窗口的线裁剪[1,2]、圆裁剪[3]和多边形裁剪[4], 多边形窗口的线裁剪[5,6]、圆的裁剪[7,8]和多边形裁剪[9]等等. 这些研究为后续的算法研究奠定的一定的基础. 但是, 由于多边形窗口的凹凸性和无规则性, 与矩形窗口内图形裁剪相比, 多边形窗口内图形裁剪更为复杂. 在实际应用中经常会遇到关于任意多边形窗口的圆裁剪问题, 如两个或多个实体间的碰

① 基金项目: 国家科技支撑课题 (2014BAH13F02)
Foundation item: National Science and Technology Support Program (2014BAH13F02)
收稿时间: 2017-11-14; 修改时间: 2017-12-06; 采用时间: 2017-12-15; csa 在线出版时间: 2018-07-28

© 中国科学院软件研究所 http://www.c-s-a.org.cn

撞、检测等等. 因此, 研究多边形窗口内圆的裁剪问题具有重要的意义.

为了提升已有的矢量圆裁剪算法效率和减少算法对内存占用率等问题, 一种具有线性复杂度的任意多边形窗口的矢量圆裁剪算法被提出[7], 该算法结合投影和射线法的线性几何思想, 借助向量叉积和向量共线基本定理, 降低了裁剪过程的运算量, 提高了裁剪效率, 具有较高的鲁棒性和通用性.

文献[8]中提出了一种任意多边形窗口的圆裁剪算法, 该算法按照逆时针方向依次求出圆与多边形各条边之间的交点并排序, 然后借助“中点检测法”判定相邻两点是线还是弧的连接, 最终完成圆的裁剪. 该算法在效率和稳定性上都取得了比较理想的结果. 但该算法只考虑了圆和多边形窗口相交及圆在多边形窗口内的情况, 并未考虑圆与多边形窗口相离以及多边性窗口在圆内的情况. 针对该问题, 如果更加全面的讨论多边形窗口和圆的位置关系的情况下完成圆的裁剪是十分有必要的.

本文提出了一种更加有效全面的多边形窗口的圆裁剪算法. 在考虑任意多边形窗口与圆的不同位置关系的情况下, 能够全面的分析多边形窗口各顶点、各边与圆之间的关系, 并对其进行了不同的裁剪. 最后仿真例子验证该方法的有效性和全面性.

2 直线段和圆的位置关系

基于任意多边形窗口, 圆的裁剪关键要解决以下两个问题:

(1) 圆与多边形窗口每条直线段是否有相交; 如果相交, 如何求交点;

(2) 圆和多边形的位置关系如何确定.

已知被裁剪圆的方程为:

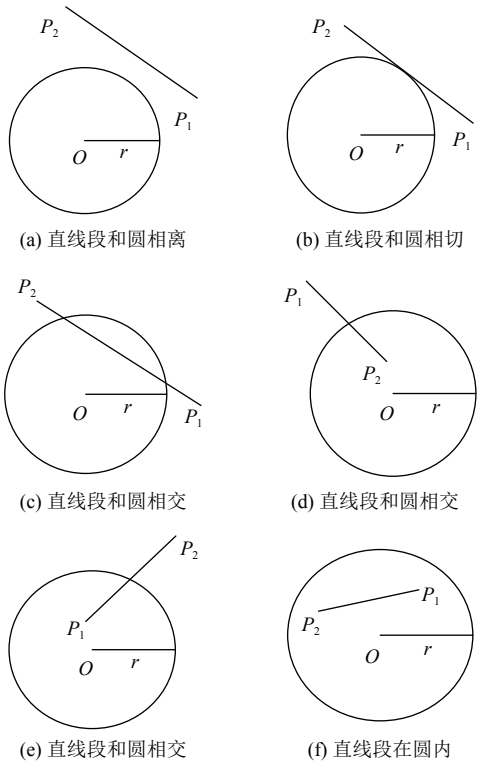
$$(x-x_0)^2+(y-y_0)^2=r^2 \tag{1}$$

其中, 圆心坐标 O 为 (x_0, y_0), 半径为 r.

给定直线段 P_1P_2, 端点坐标分别为 $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, 则按照逆时针方向总可得到其参数方程表示. 不失一般性, 假设有向直线段 P_1P_2, 为逆时针方向, 则其参数方程表示为:

$$\begin{cases} x=x_1+(x_2-x_1)t \\ y=y_1+(y_2-y_1)t \end{cases} \quad t\in[0,1] \tag{2}$$

按逆时针方向, 考虑直线段和圆的位置关系, 有 5 种可能的情况 (如图 1).



(a) 直线段和圆相离 (b) 直线段和圆相切

(c) 直线段和圆相交 (d) 直线段和圆相交

(e) 直线段和圆相交 (f) 直线段在圆内

图 1 直线段和圆的位置关系

确定直线段和圆的位置关系:

将 (2) 代入 (1), 化简整理可得:

$$at^2+bt+c=0 \tag{3}$$

其中,

$$a=(x_2-x_1)^2+(y_2-y_1)^2 \tag{4}$$

$$b=2[(x_2-x_1)(x_1-x_0)+(y_2-y_1)(y_1-y_0)] \tag{5}$$

$$c=(x_1-x_0)^2+(y_1-y_0)^2-r^2 \tag{6}$$

令$\Delta=b^2-4ac$,

若$\Delta<0$, 说明方程 (3) 没有解;

若$\Delta=0$, 说明方程 (3) 有且仅有一个解: $t_1=t_2=-\frac{b}{2a}$;

若$\Delta>0$, 说明方程 (3) 有两个解, 分别为: $t_1=\frac{-b+\sqrt{\Delta}}{2a}$, $t_2=\frac{-b-\sqrt{\Delta}}{2a}$.

综上, 如果$\Delta\geqslant 0$, 说明方程 (3) 有解, 则需要判断$t_i(i=1,2)$的取值范围来进一步确定直线段和圆的交点个数. 如果$t_i\in[0,1]$, 则其对应的点 (代入方程 (3) 可得对应的点的坐标) 即为直线段和圆的交点, 否则不是直

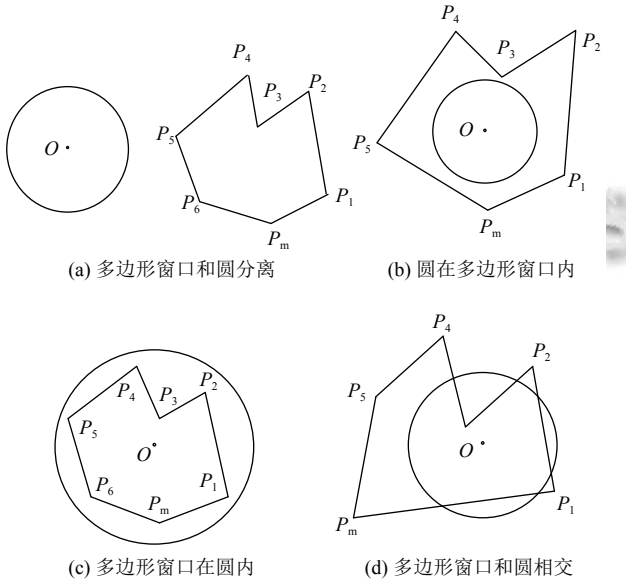
© 中国科学院软件研究所 http://www.c-s-a.org.cn

线段和圆的交点.

3 基于多边形窗口的圆的裁剪

给定任意多边形窗口$P_1P_2\cdots P_m$, $P_i(x_i,y_i)(i=1,2,\cdots,m)$ 为多边形窗口的顶点.

考虑任意多边形窗口和被裁剪圆的位置关系, 有四种可能的情况 (如图 2):



(a) 多边形窗口和圆分离 (b) 圆在多边形窗口内

(c) 多边形窗口在圆内 (d) 多边形窗口和圆相交

图 2 多边形窗口和圆的位置关系

3.1 x–扫描线算法判断多边形窗口和圆的位置关系

借助x–扫描线算法填充多边形的基本思想, 这里研究如何利用x–扫描线算法判断多边形窗口与圆的位置关系, 其基本思想为:

从多边形各个顶点$P_i(x_i,y_i)(i=1,2,\cdots,m)$出发, 依次做平行于$x$轴的扫描线（方向向右）, 并计算每条扫描线与圆的交点个数情况n_i, $n_i\in\{0,1,2\}$.

考虑以下情况:

如果$\forall\, n_i=0$或$\forall\, n_i=2$, 则说明圆在多边形窗口外, 如图 2(a);

如果n_i的取值即有 0 也有 2, 则说明圆在多边形窗口内, 如图 2(b);

如果$\forall\, n_i=1$, 则说明多边形窗口在圆内, 如图 2(c);

如果n_i的取值中 0, 1, 2 都有, 则说明圆在多边形窗口内, 如图 2(d).

备注: 借助于x–扫描线算法, 不仅能够确定多边形窗口和圆的位置关系, 而且能够准确的区分图 2 所示的不同情况, 对多边形窗口和圆的位置关系给出更加准确的判定. x–扫描线算法避免了由计算确定位置关系的大量计算, 而且解决了由计算不能区分图 2 中 (a), (b), (c) 中多边形和圆位置关系判定的问题.

3.2 基于多边形窗口的圆的裁剪.

根据 3.1 节基于x–扫描线算法对多边形窗口和圆的位置关系的判断, 接下来讨论确定圆关于多边形窗口的裁剪的问题. 针对以上四种不同情况进行不同的讨论:

(1) $\forall\, n_i=0$或$\forall\, n_i=2$

在这种情况, 被裁剪圆要么落在多边形窗口外 (如图 2(a)), 则整个圆被裁剪;

(2) n_i的取值即有 0 也有 2

在这种情况, 被裁剪圆落在多边形窗口内 (如图 2(b)), 则绘制整个圆;

(3) $\forall\, n_i=1$

在这种情况下, 多边形窗口在被裁剪圆内 (如图 2(c)), 则绘制整个多边形;

(4) n_i的取值中 0, 1, 2 都有

在这种情况下, 则说明多边形窗口与圆相交 (如图 2(d)).

下面针对情况 (4), 讨论圆弧是否被裁剪:

为了方便, 首先引入输入顶点数组P(多边形顶点) 和输出顶点数组Q(圆与多边形窗口交点), 输入顶点数组P中以逆时针方向列出, 即首先定义多边形窗口顶点的首点, 然后以逆时针方向将多边形顶点依次存入数组P.

按照 2.1 节给出的直线段和圆的交点的求法, 依次沿逆时针方向分别求多边形窗口各个边与圆的交点, 并将其按照顺序存入数组Q中, 其排序具体方法为:

取出多边形窗口的顶点数组P中的首边P_1P_2, 按照逆时针方向依次求出多边形窗口各个边P_iP_{i+1}与圆的交点参数序列, 记为: $t_1, t_2,\cdots, t_k(1\leqslant k\leqslant 2m)$, 并将其对应的点$Q_1, Q_2,\cdots, Q_k$依次存入数组$Q$中, 记$Q_{k+1}=Q_1$.

同时, 判断多边形窗口的顶点P_i是否在圆内, 如果在圆内, 将其按照逆时针方向存入数组Q中. 如果$P_{i-1}P_i$与圆有交点Q_j, 且P_i点在圆内, 则将点P_i存入数组Q中, 并将其存放在Q_j后面, 记为Q_{j+1}.

下面判断两个相邻交点之间是圆弧连接还是线段连接:

① 如果Q_i, Q_{i+1}为多边形窗口中同一条直线段与圆的交点, 则弧Q_iQ_j被裁剪, 绘制直线段Q_iQ_j;

② 如果Q_i, Q_{i+1}为多边形窗口中不同直线段与圆

© 中国科学院软件研究所 http://www.c-s-a.org.cn

的交点, 则弧Q_iQ_{i+1}被绘制;

③ 如果Q_i, Q_{i+1}中, 一个点为多边形窗口中一条边与圆的交点, 另一点为圆内的多边形顶点, 则绘制直线段Q_iQ_{i+1}.

4 算法实现与实例仿真

4.1 多边形窗口和圆的位置关系

按照上述原理, 给出本文方法的基本框架, 如图 3 所示.

4.2 实例仿真

根据本文提出的任意多边形窗口的圆裁剪算法的有效性和全面性, 讨论了图 2 给出的四种多边形窗口和圆的位置关系的情况下圆的裁剪.

图 4 验证了当多边形窗口和圆分离时, 根据本文提出算法, 得到的裁剪结果为: 圆被裁减掉.

图 5 验证了当圆在多边形窗口内时, 根据本文提出算法, 得到的裁剪结果为: 绘制整个圆.

图 6 验证了当多边形窗口在圆内时, 根据本文提出算法, 得到的裁剪结果为: 绘制整个多边形.

图 7 验证了当多边形窗口及圆相交时, 根据本文提出算法, 得到的裁剪结果为: 绘制圆在多边形窗口内部分及新生成的圆弧.

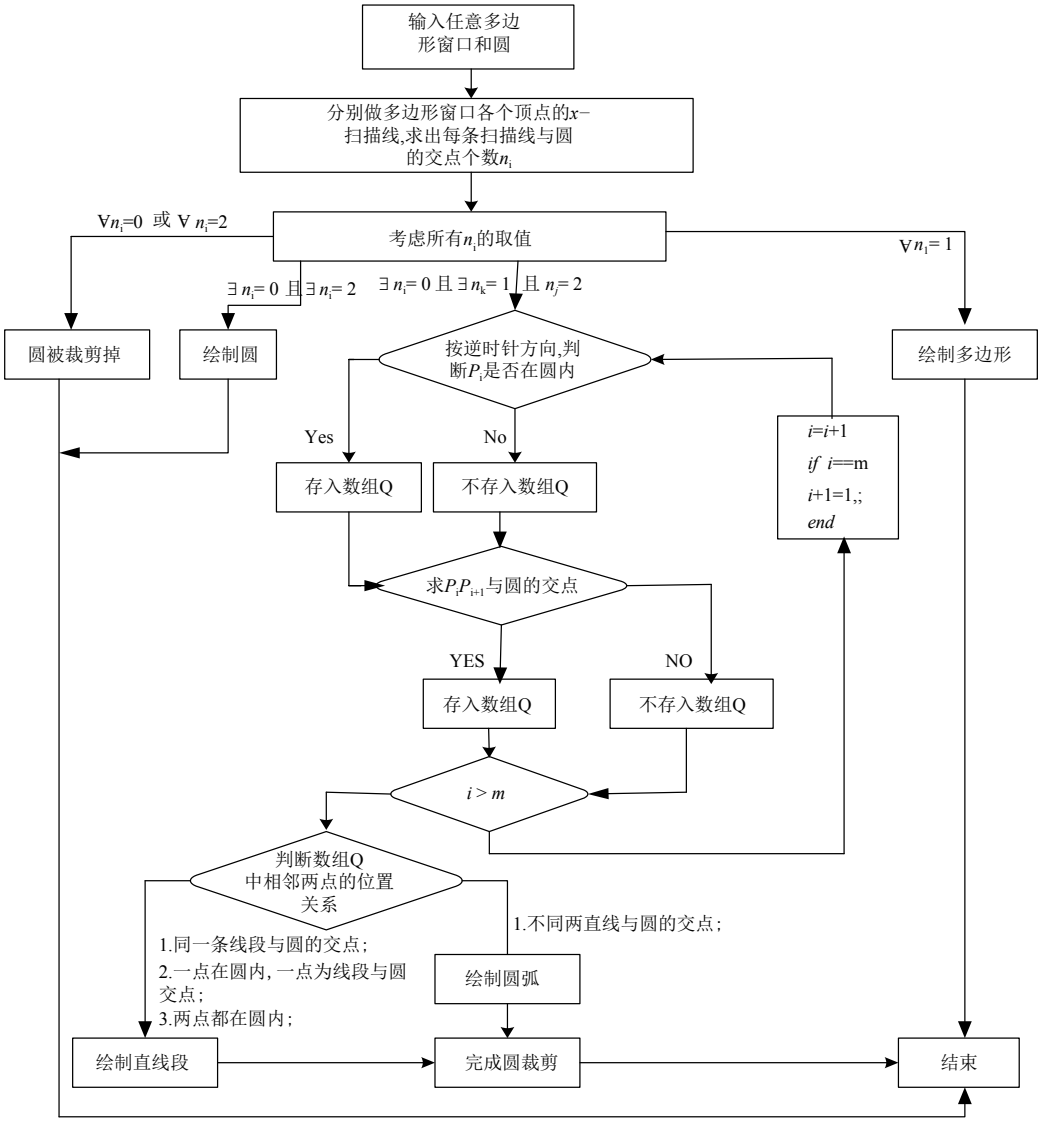
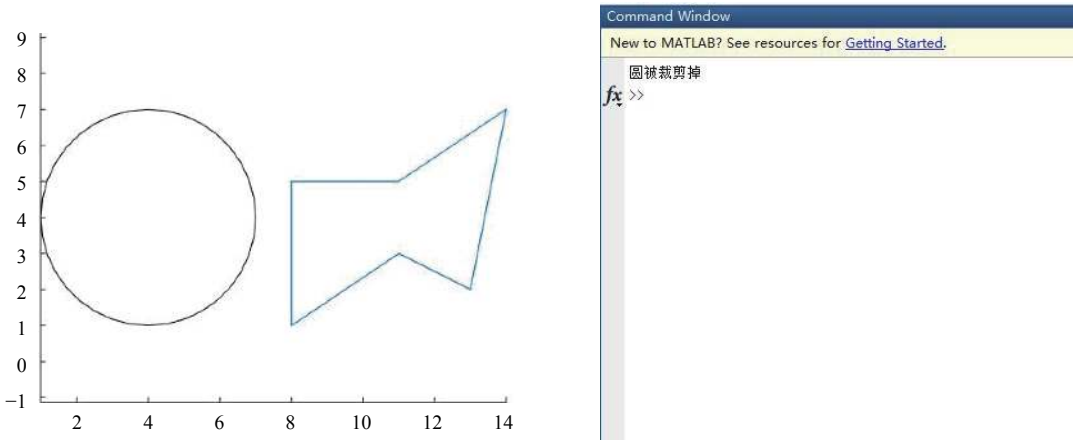


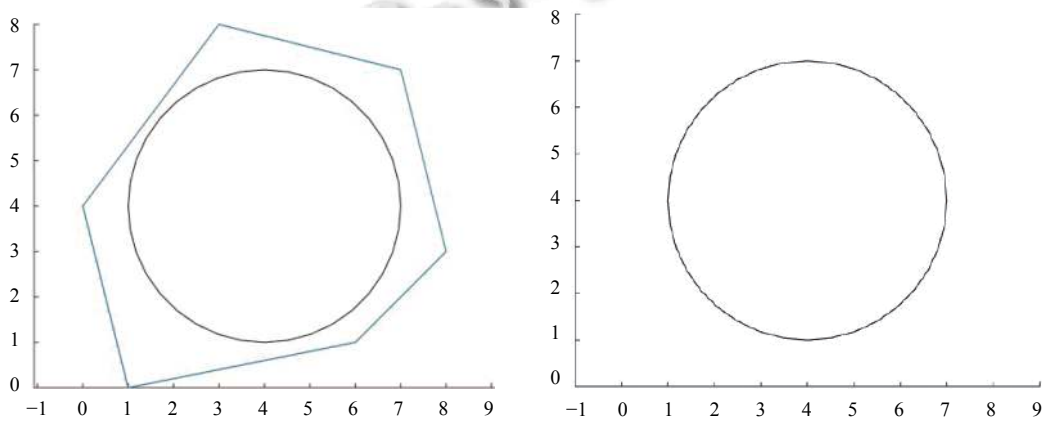
图 3 算法框架

© 中国科学院软件研究所 http://www.c-s-a.org.cn



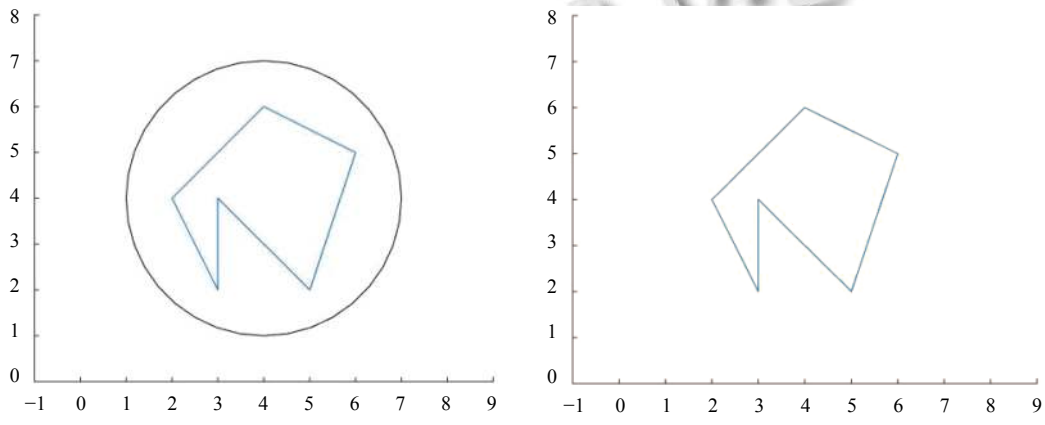
(a) 多边形窗口与圆分离 (b) 裁剪结果

图 4 多边形窗口与圆分离及裁剪结果



(a) 圆在多边形窗口内 (b) 裁剪结果

图 5 圆在多边形窗口内及裁剪结果



(a) 多边形窗口在圆内 (b) 裁剪结果

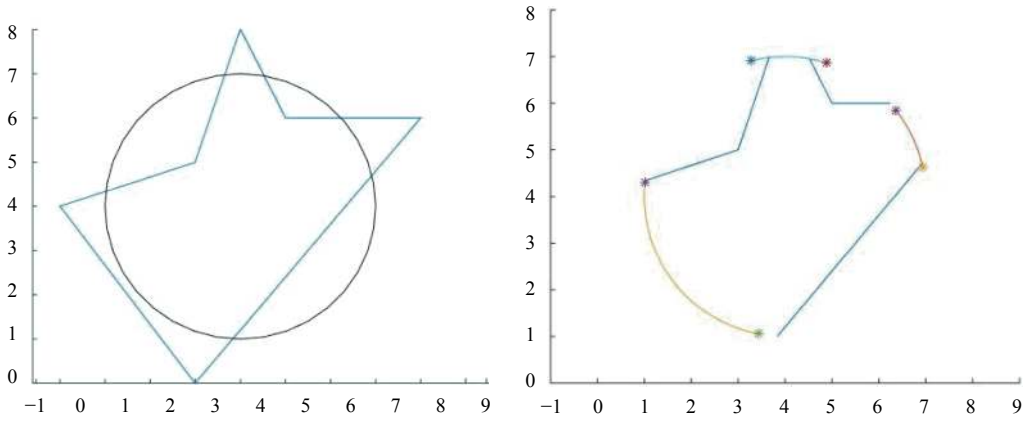
图 6 多边形窗口在圆内及裁剪结果

对于任意多边形窗口和圆的位置关系, 图 4-7 中的结果验证了本文提出方法的全面性和有效性.

相比于文献[8]的算法, 本文提出的算法对于圆在多边形窗口外的情况进行了完善, 增加了多边形窗口

© 中国科学院软件研究所 http://www.c-s-a.org.cn

和圆相离及多边形窗口在圆内的两种不同情况, 其裁剪结果验证了本文提出方法的全面性.



(a) 多边形窗口与圆相交 (b) 裁剪结果

图 7 多边形窗口与圆相交及裁剪结果

对于相邻两点之间是圆弧连接还是直线段连接上, 文献[8]中提出的“中点检测法”需要计算两点中点与当前圆弧段之间的位置关系, 从而确定其连接方式; 本文在决定两点之间到底是圆弧还是直线段连接的判断只需判断相邻两点及前后点之间的位置关系断定, 节省了一定的计算量. 相比, 本文提出方法不仅能够全面有效地完成任意多边形窗口内圆的裁剪问题, 而且能够节省一定的工作量.

5 结论

本文讨论了任意多边形窗口内圆的一种更加全面、有效地裁剪算法. 其主要思想为: 首先, 通过x–扫描线的思想, 确定由圆心出发的射线与多边形窗口的交点个数, 如果交点个数为奇数, 说明圆在多边形窗口内, 则绘制整个圆; 如果交点个数为偶数, 说明圆在多边形窗口外, 则整个圆被裁剪掉. 其次, 对于圆和多边形窗口相交的情况, 按照逆时针方向, 判断多边形顶点是否在圆内, 在圆内存入数组 Q, 否则不存入; 然后求出多边形窗口每条直线段与圆的交点, 并存入数组 Q. 最后, 通过判断两点间的关系, 决定两点之间画线还是画弧, 完成圆的裁剪. 实验结果表明, 该方法不仅能够全面、有效地完成任意多边形窗口内圆的裁剪, 而且大大降低了裁剪过程中的计算量.

参考文献

1 Huang WJ, Wang Y. A novel algorithm for line clipping. Proceedings of 2009 International Conference on Computational Intelligence and Software Engineering. Wuhan, China. 2009. 1–5.

2 洪燕, 洪智化, 刘欣. 一种更有效的矩形窗口线裁剪算法. 景德镇高专学报, 2014, 29(3): 17–18. [doi: 10.3969/j.issn.1008-8458.2014.03.008]

3 王蕊, 阎晓敏, 唐棣. 基于矩形窗口分区编码的圆形裁剪新算法. 辽宁大学学报 (自然科学版), 2011, 38(2): 177–180. [doi: 10.3969/j.issn.1000-5846.2011.02.019]

4 张钧, 王鹏. 一种新的矢量数据多边形的快速裁剪算法. 中国图象图形学报, 2008, 13(12): 2409–2413. [doi: 10.11834/jig.20081225]

5 黄文钧. 基于矩阵乘法的多边形窗口线裁剪算法. 计算机科学, 2013, 40(10): 309–317. [doi: 10.3969/j.issn.1002-137X.2013.10.066]

6 陆国栋, 邢世海, 彭群生. 基于顶点编码的多边形窗口线裁剪高效算法. 计算机学报, 2002, 25(9): 987–993. [doi: 10.3321/j.issn:0254-4164.2002.09.014]

7 苗永春, 唐权华. 任意多边形窗口的矢量圆裁剪算法. 计算机辅助设计与图形学学报, 2016, 28(9): 1451–1458. [doi: 10.3969/j.issn.1003-9775.2016.09.007]

8 杭后俊, 孙丽萍. 任意多边形窗口的圆裁剪算法. 计算机技术与发展, 2009, 19(5): 235–237, 241. [doi: 10.3969/j.issn.1673-629X.2009.05.066]

9 Simonson L J. Industrial strength polygon clipping: a novel algorithm with applications in VLSI CAD. Computer-Aided Design, 2010, 42(12): 1189–1196. [doi: 10.1016/j.cad.2010.06.008]

© 中国科学院软件研究所 http://www.c-s-a.org.cn