Appeared in Computer Graphics and Image Processing, 1977

UP-MS-CIS-77-47

DISK GENERATORS FOR A RASTER DISPLAY DEVICE

Movement Project Report No. 2

Norman I. Badler

December 1976

Work reported herein was supported by National Science Foundation Grants Nos. ENG75-10535 and MCS-76-19464

ABSTRACT

A simple modification of Horn's circle drawing procedure yields a disk generator for a class of graphic devices capable of drawing rectangular areas. Another variation produces a disk a scan-line at a time allowing it to be drawn at the refresh rate of the display. The calculations involve only additions and binary shifts.

The problem of generating circles efficiently on a graphic display device has been discussed by Horn [1]. His circle generator can be modified to enable a raster display to produce disks, that is, "filled-in" circles. Two algorithms are presented: the first for use where the graphic commands must be transmitted over a slow communications channel and the second for generating disks a scan-line at a time at the refresh rate of the raster display. It will be shown that the first algorithm is optimal in the sense of requiring the fewest graphic processor commands, provided that the processor can draw rectangles as a graphic primitive. Assuming enough local intelligence in the display processor to execute programs, the second algorithm is the most efficient in time and requires a constant amount of storage independent of the size of the disk.

1. DISK GENERATOR 1

Using Horn's notation and $s = (x+\frac{1}{2})^2 + y^2 - r^2$, the following procedure generates a disk with center x_o, y_o and radius r.

```
disk1 (x_o, y_o, r):  x←r; y←o; s←-r
                      do until y>x
                         s←s+2y+1
                         if s>o
                            then plotrects (x,y); s←s-2x+2; x←x-1
                         y←y+1
                         end
end
plotrects (x,y):      rect (x_o-x, y_o+y, x_o+x, y_o-y)
                      rect (x_o-y, y_o+x, x_o+y, y_o-x)
end
```

Most of the changes are transparent. Procedure rect is the graphic command which draws a filled-in rectangle from upper left corner (first two parameters) to lower right corner (third and fourth parameters). Since each rectangle draws points in each quadrant, the outer iterations in Horn's algorithm are unnecessary. Instead of plotting each point on the circle, the loop waits until the x coordinate must be changed. Before the change is made to x and y, the rectangle is drawn. If no rectangle is to be drawn, then y is incremented and the loop continues. Stated another way, disk1 plots a step function approximation to the disk boundary at each unit change in x (Figure 1). Once the pixel values on the video display are set, the "overprinting" of the rectangles is of no consequence.

2. DISK GENERATOR 2

By extending the step function concept in disk generator 1 an algorithm can be derived for producing disks one scan-line at a time so that refresh synchronization will not be lost between graphic commands.

The left edge of the disk is traced from top to bottom through octants 3,4,5, and 6, each octant computation resembling the basic algorithm but having a different expression for s or a different increment for x.

```
disk$^2$ : procedure ($x_o$,$y_o$,r):
    x←o;y←-r;s←-r+1            /*s= $x^2+(y+\frac{1}{2})^2-r^2$ */
    do while y<x
       if s>o then do plotline (x,y)
                      s←s+2y+2 ; y←y+1; end
               else do  x←x-1; s←s-2x+1 ; end
       end
    do while y<o               /*s=$(x+\frac{1}{2})^2+y^2-r^2$ */
       if s>o then do plotline (x,y)
                      s←s+2y+1; y←y+1; end
               else do  x←x-1; s←s-2x; end
       end
    s←-r                       /*s=$(x+\frac{1}{2})^2+y^2-r^2$ */
    do while y<-x
       if s≤o then do plotline (x,y)
                      s←s+2y+1; y←y+1; end
               else do  s←s+2x+2; x←x+1; end
       end
    do while y≤r               /*s=$x^2+(y-\frac{1}{2})^2-r^2$ */
       if s≤o then do plotline (x,y)
                      s←s+2y; y←y+1; end
               else do  s←s+2x+1; x←x+1; end
       end
    end disk2
plotline (x,y):  line ($x_o$+x, $y_o$+y, $x_o$-x, $y_o$+y)
    end plotline
```

For octant 3 the initialization of s is derived from $s=x^2+(y+\frac{1}{2})^2-r^2=-r+\frac{1}{4}$ at $(x,y)=(o,-r)$. The fraction is ignored since only integer values are used. The value of s is adjusted prior to decrementing x so that scan-lines will be drawn when the current x value is the last that would fall inside the disk boundary defined by $s\leq o$. Therefore an adjustment is performed initally: $s=s-2x+1=-r+1$ at $x=o$.

Between octants 3 and 4, the value of s is not changed. Since $y=x$, $s=x^2+(y+\frac{1}{2})^2-r^2=y^2+(x+\frac{1}{2})^2-r^2$.

Moving into octant 5, the value of s is reset because s is now adjusted after x is incremented. Scan-lines will be drawn when the current x value is the first that lies inside the disk boundary. The re-initialization is simply $s=(x+\frac{1}{2})^2+y^2-r^2=-r+\frac{1}{4}$ at $(x,y)=(-r,o)$. Again the fraction can be ignored.

Between octants 5 and 6 no change in s is required. Since $y=-x$, $s=(x+\frac{1}{2})^2+y^2-r^2=((-y)+\frac{1}{2})^2+(-x)^2-r^2=(y-\frac{1}{2})^2+x^2-r^2$.

An example of the output of disk2 appears in Figure 2. If desired, clipping to the screen boundaries is easily accomplished by tests inside plotline.

3. CONCLUSIONS

The choice of which algorithm to use depends on the graphic configuration and the local intelligence in the display processor. If graphic commands must be sent over a slow communications channel to the raster device then disk1 generates fewer commands and would be more efficient. That is, to draw a disk of radius r requires at worst $2\lceil r-r/\sqrt{2}\rceil=\lceil r(2-\sqrt{2})\rceil$ rectangles or $4\lceil r(2-\sqrt{2})\rceil$ integers. (Here $\lceil r\rceil$ signifies the smallest integer not smaller than r). Algorithm disk2, however, must generate 2r+1 lines or

8r+4 integers. Comparing these costs:

$$4\lceil r(2-\sqrt{2})\rceil < 8r+4$$

or approximately $\lceil 0.6r\rceil < 2r+1$

for all positive r, a saving of more than two-thirds.

Assuming that Horn's algorithm is the most efficient for generating the circle boundary points, it is easy to show that the disk can be drawn with no fewer graphic command parameters than those generated by disk1, excluding the cases where the radius is only one or two. Since each rectangle accounts for "corners" in the edge of the disk, the use of fewer rectangles would be impossible.

Disk1 is also more efficient than a disk generator described by Fulton [2] for a plasma-panel display. Instead of drawing filled-in rectangles, Fulton draws four horizontal lines for every disk boundary point computed in the first octant by a method nearly identical to Horn's. In all, $\lceil r/\sqrt{2}\rceil$ values of y are calculated so that $4\lceil r/\sqrt{2}\rceil$ lines are drawn. Since $2r+1<4\lceil r/\sqrt{2}\rceil$ for $r\geq 2$, extra lines are drawn near the top and bottom of the disk where several y values may be identical. This technique is therefore more inefficient than either disk1 or disk2. Moreover, refresh synchronization cannot be maintained since the lines are not generated in the raster scan order.

If the disks can be generated within the graphics processor, then disk2 is clearly superior. Since each scan-line is computed in sequence during a single refresh cycle of the raster display, the limiting factor is not the absolute number of graphic commands but the refresh rate of the display. The worst case is one disk per frame, or 60 disks per second in a video display. The only requirements of the processor are addition and multiplication (by shifting) by 2, and storage locations for x_0, y_0, r, x, y, s and the actual code.

Acknowledgements. The author wishes to gratefully acknowledge the support of NSF Grants ENG75-10535 and MCS-76-19464.

REFERENCES

1. B.K.P. Horn, Circle generators for display devices, Computer Graphics and Image Processing 5, 1976, 280-288.

2. Fulton, David L., A plasma-panel interactive graphics system, Proceedings of the Society for Information Display 15(2), 1974, pp.74-80.

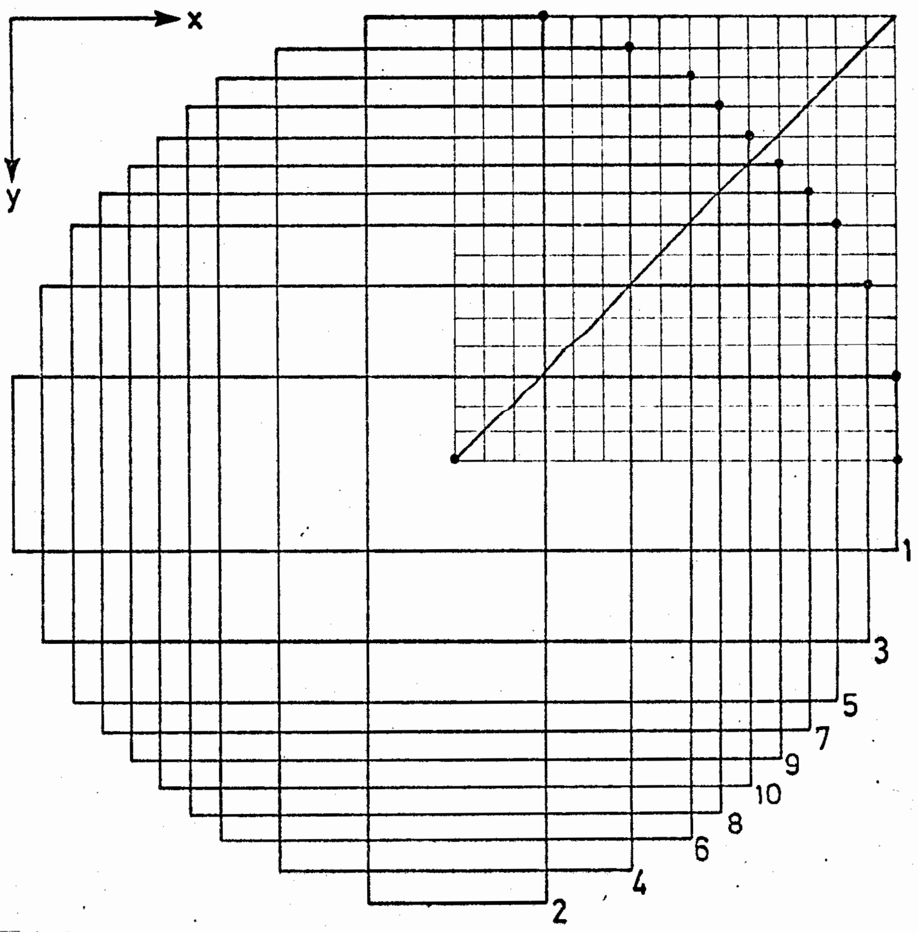

Figure 1. Disk generated by rectangles (in order indicated).

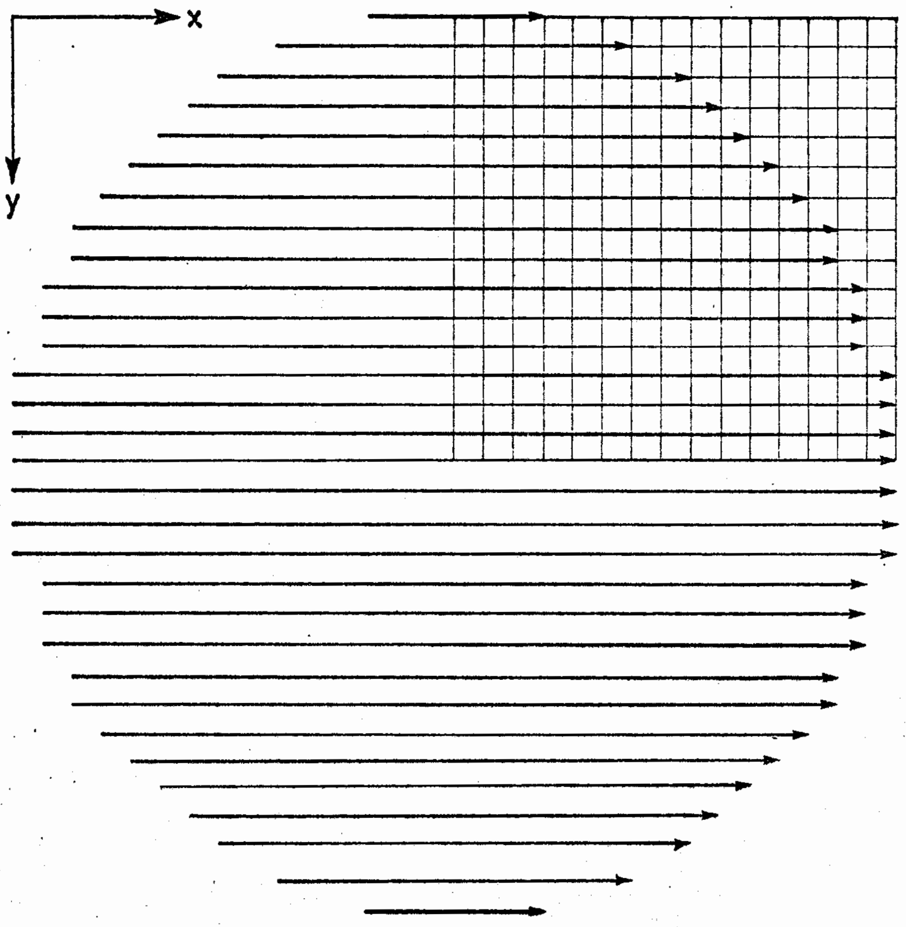

Figure 2. Disk generated by scan lines.