An algorithm is given for computer control of a digital plotter.

The algorithm may be programmed without multiplication or division instructions and is efficient with respect to speed of execution and memory utilization.

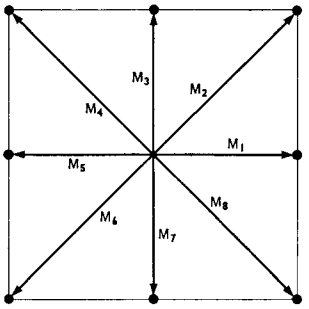
Algorithm for computer control of a digital plotter

by J. E. Bresenham

This paper describes an algorithm for computer control of a type of digital plotter that is now in common use with digital computers.[1]

The plotter under consideration is capable of executing, in response to an appropriate pulse, any one of the eight linear movements shown in Figure 1. Thus, the plotter can move linearly from a point on a mesh to any adjacent point on the mesh. A typical mesh size is 1/100th of an inch.

Figure 1 Plotter movements

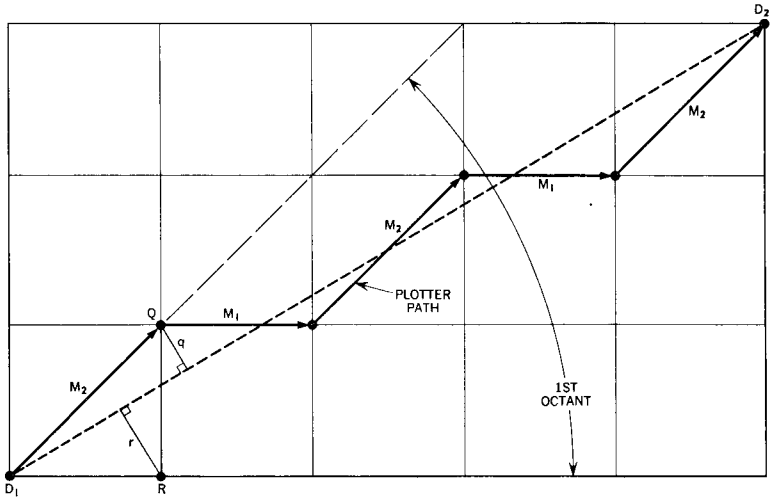


The data to be plotted are expressed in an (x, y) rectangular coordinate system which has been scaled with respect to the mesh; i.e., the data points lie on mesh points and consequently have integral coordinates.

It is assumed that the data include a sufficient number of appropriately selected points to produce a satisfactory representation of the curve by connecting the points with line segments, as illustrated in Figure 2. In Figure 3, the line segment connecting

Figure 2 Curve defined by linear segments joining data points

Figure 3 Sequence of plotter movements



the two adjacent data points $D_1(x_1, y_1)$ and $D_2(x_2, y_2)$ is shown on the mesh, drawn on an enlarged scale. Also shown is the path actually taken by the plotter in accordance with the algorithm. In each instance, the mesh point nearest the desired line segment is selected. For example, since Q is closer to the line segment than R, Q is chosen as the second mesh point in the path taken by the plotter in approximating the desired segment joining D_1 and D_2.

algorithm for first octant

In the first case to be considered, it is assumed that D_2 lies in the first octant, relative to a rectangular coordinate system obtained by translation of the origin to D_1. It is apparent that the plotter movement can be accomplished by a sequence of moves involving only M_1 and M_2, as illustrated in Figure 3.

In Figure 4, an (a, b) coordinate system obtained by translation of the origin to D_1 is shown. Consequently, the new coordinates of D_2 are $(\Delta a, \Delta b) = (x_2 - x_1, y_2 - y_1)$.

When the plotter has progressed to the point P_{i-1}, as indicated in Figure 4, the next movement is either M_1 (to the point R_i) if $r_i < q_i$, or M_2 (to the point Q_i) if $r_i \geq q_i$.

It follows from similar triangles that $r_i' - q_i'$ has the same sign as $r_i - q_i$. Since the segment D_1D_2 lies in the first octant, $\Delta a > 0$. Thus, $\nabla_i = (r_i' - q_i')\Delta a$ also has the same sign as $r_i - q_i$ and may be used for computational convenience in selecting the appropriate movement, either M_1 or M_2. Later in the paper, ∇_i is shown to satisfy the recursive relation:

$$\left.\begin{aligned} \nabla_1 &= 2\Delta b - \Delta a \\ \nabla_{i+1} &= \begin{cases} \nabla_i + 2\Delta b - 2\Delta a & \text{if } \nabla_i \geq 0 \\ \nabla_i + 2\Delta b & \text{if } \nabla_i < 0 \end{cases} \end{aligned}\right\}, \tag{1}$$

where

$$\Delta a = x_2 - x_1, \qquad \Delta b = y_2 - y_1. \tag{2}$$

The values of ∇_i computed by means of (1) and (2) are used to determine the movement of the plotter:

$$\text{if } \left\{\begin{array}{ll} \nabla_i < 0, & \text{execute } m_1 \\ \nabla_i \geq 0, & \text{execute } m_2 \end{array}\right\}, \qquad i = 1, \cdots, \Delta a, \tag{3}$$

where

$$m_1 = M_1 \quad \text{and} \quad m_2 = M_2. \tag{4}$$

Expressions (1), (2), (3), and (4) constitute the algorithm for the present case. For other octants, the right members of each equality in (2) and (4) must be modified.

proof of the recursive relation

Before indicating this modification, the recursive relation (1) is shown to hold. The notation employed in Figure 4 is as follows: $(a_{i-1}, \hat{b}_{i-1})$ is used to denote the coordinates of P_{i-1}. Consequently, the coordinates of R_i and Q_i are, respectively, $(a_i, \lfloor b_i \rfloor)$ and $(a_i, \lceil b_i \rceil)$, where "$\lfloor\ \rfloor$" and "$\lceil\ \rceil$" are used to denote the *floor* and *ceiling* operators.[2] Denoting the ordinate of S_i by b_i, the coordinates of S_i are (a_i, b_i).

This notation is used to rewrite the expression for ∇_i:

$$\nabla_i = (r_i' - q_i')\Delta a = [(b_i - \lfloor b_i \rfloor) - (\lceil b_i \rceil - b_i)]\Delta a.$$

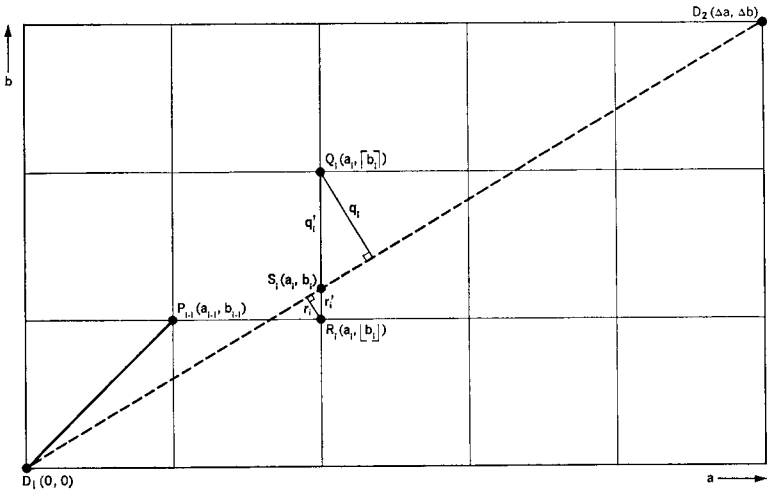
By noting that $b_i = (\Delta b/\Delta a)a_i$,

$$\nabla_i = 2a_i\Delta b - (\lfloor b_i \rfloor + \lceil b_i \rceil)\Delta a.$$

Since the line segment D_1D_2 lies in the first octant, $a_i = a_{i-1} + 1$. By definition, $\lceil b_i \rceil = \hat{b}_{i-1} + 1$ and $\lfloor b_i \rfloor = \hat{b}_{i-1}$. These relations are used to rewrite the latter expression for ∇_i in a form free of a_i and b_i:

$$\nabla_i = 2a_{i-1}\Delta b - 2\hat{b}_{i-1}\Delta a + 2\Delta b - \Delta a.$$

Figure 4 Notation for the algorithm



Applying the initial condition for the coordinates of P_0, $a_0 = 0$ and $\hat{b}_0 = 0$,

$$\nabla_1 = 2\Delta b - \Delta a.$$

If $\nabla_i \geq 0$,

$$\hat{b}_i = \hat{b}_{i-1} + 1,$$

so that

$$\nabla_{i+1} = 2(a_{i-1} + 1)\Delta b - 2(\hat{b}_{i-1} + 1)\Delta a + 2\Delta b - \Delta a$$
$$= \nabla_i + 2\Delta b - 2\Delta a.$$

If $\nabla_i < 0$,

$$\hat{b}_i = \hat{b}_{i-1},$$

so that

$$\nabla_{i+1} = 2(a_{i-1} + 1)\Delta b - 2\hat{b}_i\Delta a + 2\Delta b - \Delta a$$
$$= \nabla_i + 2\Delta b.$$

Thus (1) has been shown to hold.

other octants

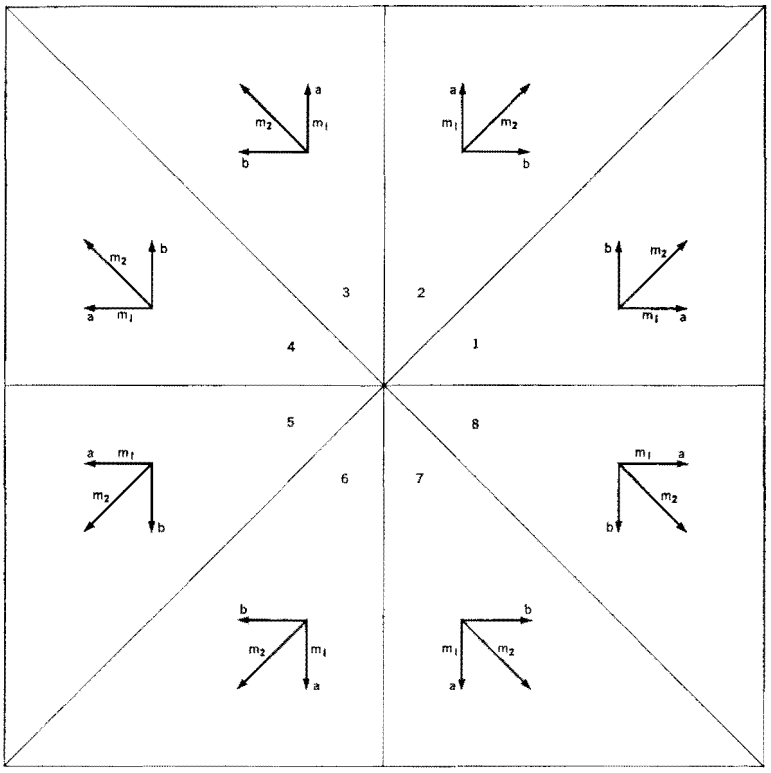
The second data point has been assumed to be in the first octant with respect to the first data point. If the second data point lies in another octant, an (a, b) rectangular coordinate system is again chosen with origin at the first data point, but with the axes oriented individually for each octant, as shown in Figure 5. Directions associated with the plotter movements m_1 and m_2 are also indicated. This information is summarized in the left columns of Table 1 together with the assignments made to m_1 and m_2. Thus, the variants of Equations (1) and (4) have been specified for each of the eight octants, and the reader may verify that, in conjunction with (2) and (3) as previously stated, they comprise a correct formulation for the general case.

To complete the algorithm, a computational procedure is needed to determine the applicable octant for an arbitrary pair of data points so that the appropriate forms of (2) and (4) can be determined. The form of (2) depends on the sign of $|\Delta x| - |\Delta y|$.

Table 1 Determination of form of Equations 2 and 4

Δx	Δy	$\lvert\Delta x\rvert - \lvert\Delta y\rvert$	OCT	Δa	Δb	X	Y	Z	m_1	F	m_2	G
≥ 0	≥ 0	≥ 0	1	$\lvert\Delta x\rvert$	$\lvert\Delta y\rvert$	1	1	1	M_1	(1, 0, 0, 0)	M_2	(1, 0, 0, 0)
≥ 0	≥ 0	< 0	2	$\lvert\Delta y\rvert$	$\lvert\Delta x\rvert$	1	1	0	M_3	(0, 1, 0, 0)	M_2	(1, 0, 0, 0)
≥ 0	< 0	≥ 0	8	$\lvert\Delta x\rvert$	$\lvert\Delta y\rvert$	1	0	1	M_1	(1, 0, 0, 0)	M_8	(0, 0, 0, 1)
≥ 0	< 0	< 0	7	$\lvert\Delta y\rvert$	$\lvert\Delta x\rvert$	1	0	0	M_7	(0, 0, 0, 1)	M_8	(0, 0, 0, 1)
< 0	≥ 0	≥ 0	4	$\lvert\Delta x\rvert$	$\lvert\Delta y\rvert$	0	1	1	M_5	(0, 0, 1, 0)	M_4	(0, 1, 0, 0)
< 0	≥ 0	< 0	3	$\lvert\Delta y\rvert$	$\lvert\Delta x\rvert$	0	1	0	M_3	(0, 1, 0, 0)	M_4	(0, 1, 0, 0)
< 0	< 0	≥ 0	5	$\lvert\Delta x\rvert$	$\lvert\Delta y\rvert$	0	0	1	M_5	(0, 0, 1, 0)	M_6	(0, 0, 1, 0)
< 0	< 0	< 0	6	$\lvert\Delta y\rvert$	$\lvert\Delta x\rvert$	0	0	0	M_7	(0, 0, 0, 1)	M_6	(0, 0, 1, 0)

Figure 5 Axes orientation



To find the form of (4), Boolean variables X, Y, and Z, corresponding to Δx, Δy, and $|\Delta x| - |\Delta y|$, are introduced. As shown in Table 1, these variables assume the value 0 or 1, depending on whether or not the correspondent is negative. To determine the assignment of m_1, the function

$$F(X, Y, Z) = (XZ, Y\bar{Z}, \bar{X}Z, \bar{Y}\bar{Z}),$$

found by inspection of Table 1, is introduced. Correspondence between values assumed by F and the assignment of m_1 is indicated by columns headed F and m_1. Similarly,

$$G(X, Y) = (XY, \bar{X}Y, \bar{X}\bar{Y}, X\bar{Y})$$

is used in conjunction with the G- and m_2-columns of Table 1 to make the appropriate assignment to m_2.

concluding remarks

The algorithm can be programmed without the use of multiplication or division. It was found that 333 core locations were sufficient for an IBM 1401 program (used to control an IBM 1627). The average computation time between successive incrementations was approximately 1.5 milliseconds.

A functionally similar algorithm reported in the literature[3] is described as requiring 513 core positions and 2.4 milliseconds between successive incrementations.[4]

ACKNOWLEDGMENT

The suggestions of the author's colleagues, D. Clark and A. Hoffman, were of considerable assistance.

CITED REFERENCES AND FOOTNOTES

1. This paper is based on "An incremental algorithm for digital plotting," presented by the author at the ACM National Conference at Denver, Colorado, on August 30, 1963.
2. The *floor* ($\lfloor\ \rfloor$) and *ceiling* ($\lceil\ \rceil$) operators are defined as follows: $\lfloor x \rfloor$ denotes the greatest integer not exceeding x, and $\lceil x \rceil$ denotes the smallest integer not exceeded by x. This notation was introduced by Iverson; see, for example: K. E. Iverson, "Programming notation in systems design," *IBM Systems Journal* **2**, 117 (1963).
3. F. G. Stockton, Algorithm 162, XMOVE PLOTTING, *Communications of the ACM* **6**, Number 4, April 1963. Certification appears in **6**, Number 5, May 1963.
4. F. G. Stockton, *Plotting of Computer Output*, Bulletin No. 139, California Computer Products, May 1963.