(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 163 872**
A2

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **85104590.6**

(22) Date of filing: **17.04.85**

(51) Int. Cl.⁴: **G 06 F 15/72**, G 09 G 1/06

(30) Priority: **05.05.84 GB 8411594**

(43) Date of publication of application: **11.12.85**
Bulletin 85/50

(84) Designated Contracting States: **DE FR GB**

(71) Applicant: **International Business Machines Corporation, Old Orchard Road, Armonk, N.Y. 10504 (US)**

(72) Inventor: **Normington, Glyn, 10 Clifton Hill, Winchester Hampshire (GB)**

(74) Representative: **Appleton, John Edward, IBM United Kingdom Patent Operations Hursley Park, Winchester, Hants, SO21 2JN (GB)**

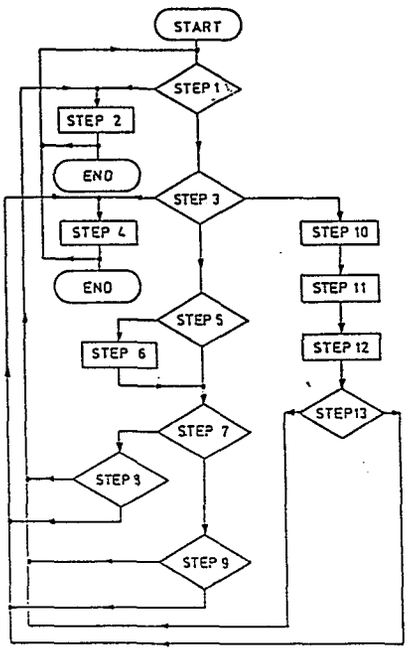
(54) Method of correlating on straight lines in an interactive display system.

(57) A method of correlating on straight lines with a correlation window in an interactive display including the steps of:

a) defining a correlation window on a display area and dividing the rest of the display area into eight surrounding regions.

b) determining which lines stored as coordinates in a display list buffer can be trivially accepted or rejected by computing the configuration of regions in which the end points lie.

c) determining for each line not trivially accepted or rejected, on which side of the line one or two selected corner points of the correlation window lie and consequently whether or not it intersects the correlation window.



0163872

METHOD OF CORRELATING ON STRAIGHT LINES IN AN INTERACTIVE DISPLAY SYSTEM

The present invention relates to improvements in methods of correlating on straight lines with a correlation window in an interactive display.

A common form of interaction with a graphic picture is for an operator to want to identify, to the application with which the operator is interacting, a particular line within the picture. This may be for one of a variety of reasons, for example to move the line in its relation to the other objects, modify or delete the object. A possible method of accomplishing this would be to identify the object by means of a command, e.g., DELETE LINE 23. It is much more natural however for the operator to use a suitable device to actually point to the line. Typical devices are joystick, thumbwheel, trackball, or optical mouse coupled to a cursor in the display screen.

The immediate result of such a pointing is an (x, y) position on the screen. A correlation operation has then to be carried out to determine which of the lines in the picture actually passes through (or sufficiently close to) that point.

In order to do this a picture definition (display list) has to be scanned, just as if it were being drawn, but instead of actually drawing the lines defined in the list, they are each inspected to see whether they intersect a small rectangle (for example) drawn around the point of interest.

It is important that this process occurs quickly. The operator expects a rapid system response to indicate whether or not the pointing was successful. Since the operator has generally only pointed at one line out of the many in the

picture, a critical performance factor is how quickly each of the remaining objects can be rejected.

The Cohen-Sutherland clipping algorithm is described on page 146 'Fundamentals of Interactive Computer Graphics' - J D Foley, A Van Dam published by Addison Wesley 1982.

The Cohen-Sutherland clipping algorithm is designed to identify efficiently those lines that can be trivially accepted or rejected by using region checks. Intersection calculations are required only for those lines for which neither occurs. The algorithm is especially efficient in the two extreme cases of a large window including most of the primitives, or a relatively small window and a large, dense picture in which most primitives lie outside the window; most lines can then be trivially accepted or trivially rejected, respectively.

The algorithm starts by assigning to each endpoint of a line a four-bit outcode based on nine regions with bit 1 the leftmost bit. The nine regions are the correlation window surrounded by eight regions, four having a side adjacent the correlation window sides and four diagonal corner regions. Each bit in the out-code is set to 1 (TRUE) if a given relation between the endpoint and window is true:

Bit 1 - point is above window,
Bit 2 - point is below window,
Bit 3 - point is to right of window,
Bit 4 - point is left of window;

otherwise the bit is 0 (FALSE).

One way to calculate the outcode in systems allowing bit manipulation derives from the observation that bit 1 is the

sign bit of (ymax - y); bit 2 is the sign bit of (y - ymin); bit 3, (xmax - x); bit 4, (x - xmin). A point is inside the window (outcode 0000) if all these differences are nonnegative. A line can be trivially accepted if both ends are in the window (corresponding to outcodes of 0000). A line is trivially rejected if both endpoints are in a region above, below, to the left, or to the right of the window. This will be the case if the corresponding bits in the outcodes for both endpoints are 1, and this can easily be tested by taking the logical AND of the outcodes and testing for "NOT 0000".

If the result of the logical and is 0000, the line can be neither trivially rejected nor accepted - it may intersect the window.

The algorithm has the drawback in that in clipping lines which extend through the correlation window at least two intersections must be calculated. This can be very time consuming in data processing terms.

U.S.A. Patent 4,412,296 - Smiths Industries Inc. describes a circuit arrangement for determining line conflict in a display.

The circuit is capable of generating curvilinear as well as linear border segments as produced by a graphics generator. The circuit is capable of generating a multitude of clipping boundaries at different horizontal positions across the CRT screen.

Positional information is fed from the graphics generator to the circuit so that this input information may address a look-up table to determine whether a conflict exists between certain portions of the symbol to be displayed and an area in which a high priority symbol is to be displayed. If a

conflict is determined to exist, the circuit clips the portion of the symbol that would otherwise obscure the high priority information.

It is an object of the present invention to provide an improved method of correlating on straight lines in an interactive display device whereby the time taken for the processing of the method is shorter than in previous methods resulting in a shorter wait time for the user of the system.

According to the invention there is provided a method of correlating on straight lines with a correlation window in an interactive display system comprising the steps of:

a) defining a correlation window on a display area and dividing the rest of the display area into eight surrounding regions.

b) determining which lines stored as coordinates in a display list buffer can be trivially accepted or rejected by computing the configuration of regions in which the end points lie.

c) determining for each line not trivially accepted or rejected, on which side of the line one or two selected corner points of the correlation window lie and consequently whether or not it intersects the correlation window.

According to a second aspect of the invention there is provided an interactive display system comprising a display screen and a control unit including a processor device, a read only memory for storing a control program and a random access store having at least register regions and a display test buffer store,

first means under the control of the control program for defining a correlation window on a display area and dividing the rest of the display area into eight surrounding regions,

second means under the control of the control program for determining which lines stored as coordinates in the display list buffer can be trivially accepted or rejected by computing the configuration of regions in which the end points lie,

third means under the control of the control program for determining for each line not trivially accepted or rejected, on which side of the line one or two selected corner points of the correlation window lie and consequently whether or not it intersects the correlation window.

In order that the invention may be fully understood a preferred embodiment will now be described with reference to the accompanying drawings in which:

FIG. 1 diagrammatically shows a division of a display area into nine regions including a correlation window.

FIG. 2 is a flow chart illustrating the steps of the preferred method.

FIG. 3 is a block schematic of an interactive display system.

The Cohen-Sutherland algorithm referred to above involves performing intersection calculations between the input line and the correlation window. These calculations are performed either by using divide instructions or by using an iterative

mid-point subdivision algorithm.

If the input line is found to intersect the correlation window, then a correlate hit has occurred, otherwise no correlate hit has occurred.

The method embodying the invention has the following advantages:

(i) No divisions are performed.

(ii) The method is not iterative and operates in a single 'pass'.

(iii) The method utilises the 'side function' of a point with respect to a line. An intersection between a line segment and a semi-infinite ray is detected by computing the side function of the end point of the ray with respect to the line segment.

(iv) The outcodes of the line segment end points are extended to 5-bit outcodes and are used in a novel way to distinguish between various sub-cases which arise in the new algorithm. In particular:-

1. The 'corner to corner' sub-case is distinguished from the 'middle to corner' sub-case by testing the logical OR of the outcodes for equality with 11111000.

2. In the 'middle to corner' sub-case, the direction of the line segment (ordered from start point to end point) is

determined to be clockwise or anticlockwise around the correlation window by shifting one outcode left and computing the logical AND of the result with the other outcode.

Referring now to Fig. 1, a screen 10 is shown divided into nine areas. A pick window 1 and areas 2-9 surrounding the pick or correlation window. Each area is identified with a 5-bit outcode. The bits within each outcode are indicated as follows:

BITS 1 and 5	represent below pick rectangle
BIT 2	represents right of the pick rectangle
BIT 3	represents above the pick rectangle
BIT 4	represents left of the pick rectangle.

Thus Area 1 = 00000 (Pick area)
Area 2 = 00110 Top left.
Area 3 = 00100 Centre top.
Area 4 = 01100 Top right.
Area 5 = 00010 Centre left.
Area 6 = 01000 Centre right.
Area 7 = 10011 Bottom left.
Area 8 = 10001 Centre bottom.
Area 9 = 11001 Bottom right.

Using this notation the effect of shifting an outcode left by one bit is to produce, in the first four bits of the code, the outcode of the region which corresponds to the result of rotating the original outcode region 90 degrees anti-clockwise.

While in Figure 1 the areas are shown to be more or less the same size in reality the pick window can be of the order

of ten or fifteen pels square and the other areas taking up a correspondingly larger area so that the centre areas have one short edge (adjacent the pick window) and comparatively larger side adjacent the other areas.

The preferred method embodying the invention will now be described with reference to the flow chart of Fig. 2 and the interactive display illustrated schematically in Fig. 3. The component parts of the system of Fig. 3 are a display screen 30 connected to a control unit 31. The control unit 31 includes a processor 32 connected through a bus 33 to a read only store 34 a random access store 35, and a display buffer 36. Input/output connections are made to a keyboard 37 and a cursor control device 38 which may be a light pen, an optical mouse, a tablet or similar device.

The processor 32 is controlled by a control program stored in the ROS 34. The RAM 35 contains areas set aside to perform the function of temporary storage registers. In the preferred method ten registers A-J are utilised.

The control unit 31 also has a connection 39 to a host data processing unit, which send a display list, through a suitable link which defines the contents of a picture to be displayed on the screen 30.

The display list is stored in the display list buffer 36 in the form of coordinate values for the end points of lines which form the display.

A user of the system will wish to indicate a line or area which requires some action to be taken, e.g., deleted from total picture, highlighted, moved etc. The user moves the cursor on the screen through the device 38.

In a complex picture the cursor will not necessarily come to rest on a particular line and the system must determine which lines are to be selected by the cursor position. A pick window 1 (Fig. 1) is defined around the cursor position. The method described below is then used to correlate the lines of the display with the pick window, under control of a control program stored in the ROS 34.

Register areas A-J are defined as follows:

Register A pick window left edge x_L
" B pick window right edge x_R
" C pick window bottom edge y_B
" D pick window top edge y_T
" E line end coordinates x, y,: x_2 y_2
" F Outcode 1
" G Outcode 2
" H Hit Flag set initially to '0'
" I OR
" J AND.
" K Work register.

The setting of Register F and G is as follows.

The outcodes oc of a point (x,y) with respect to the pick window are defined to be an 8-bit byte with bits set as follows:

oc (1), oc (5) set to 1 if $y < y_B$ otherwise 0
oc (2) set to 1 if $x > x_R$ otherwise 0
oc (3) set to 1 if $y > y_T$ otherwise 0
oc (4) set to 1 if $x < x_L$ otherwise 0

Outcode bit 1 is duplicated in bit 5 so that in middle to corner cases it can easily be determined whether the line segment points clockwise or anti-clockwise around the pick

window. The outcodes are ordered in an anti-clockwise sequence around the pick window.

SHIFT LEFT is defined as:

SHIFT_LEFT(oc) = oc' where
oc'(1) = oc(2)
oc'(2) = oc(3)
oc'(3) = oc(4)
oc'(4) = oc(5)
oc'(5) = oc(6)
oc'(6) = oc(7)
oc'(7) = oc(8)
oc'(8) = 0

Register I is set with the result of register F OR register G.

Register J is set with the result of register F AND register G.

The following steps, the flow of which is illustrated in Fig. 2 are then processed.

Step 1 The following is determined.

If any of the following conditions is true

Contents of Register F = 00000000
" " G = 00000000
" " I = 01010000
" " I = 10101000

then the line is accepted as passing through the pick window and Step 2 is entered. Otherwise Step 3 is

entered.

Step 2 Register H is set to '1' indicating a 'hit' for the line being processed and a corresponding flag is set in the display list buffer.

The coordinates for the next line to be processed are obtained and Step 1 is re-entered. Processing stops when the display list buffer is exhausted.

Step 3 The following is determined.

If the contents of Register J = 00000000 and the contents of Register I = 11111000 then Step 5 is entered.
If the contents of Register J = 00000000 and the contents of Register I do not equal 11111000 then Step 10 is entered.

If the contents of Register J do not equal 00000000 then Step 4 is entered.

Step 4 The line is rejected Register H remains at '0' and a corresponding flag is set in the display list buffer. The coordinates for the next line to be processed are obtained and Step 1 is re-entered. Processing stops when the display list buffer is exhausted.

Step 5 x_1 and x_2 are compared if x_1 is greater than x_2 then Step 6 is entered.
If not then Step 7 is entered.

Step 6 This step swaps the ends of the line to ensure that x_1 is less than x_2. A simple routine for performing this function uses two work registers in the RAM, T and

U, as follows

$T = x_1$
$x_1 = x_2$
$x_2 = T$
$T = y_1$
$y_1 = y_2$
$y_2 = T$
U = Register F
Register F = Register G
Register G = Register U

Continue at Step 7.

Step 7, 8 and 9 (corner to corner routine) determine whether a line having end points in diagonally opposite corner regions 2 and 9 or 4 and 7 passes between the corner points of the pick window 1 which intersect the other pair of corner regions.

Step 7 y_1 and y_2 are compared if y_1 is less than y_2 then Step 8 is entered if not Step 9 is entered.

Step 8 If the expression:
(NOT(left side(x_1, y_1, x_2, y_2, x_R, y_B)))
AND (NOT(right side(x_1, y_1, x_2, y_2, x_L, y_T)))
is TRUE then Step 2 is entered otherwise Step 4 is entered.

Step 9 If the expression:
(NOT(left side(x_1, y_1, x_2, y_2, x_L, y_B)))
AND(NOT(right side(x_1, y_1, x_2, y_2, x_R, y_T)))
is TRUE then Step 2 is entered otherwise Step 4 is entered.

Steps 10, 11, 12 and 13 (middle to corner) determine for a line having one end point in a centre area 3, 5 6 or 8 and its other end point in an area in a different row and column of Figure 1 to the area containing the first end point. Whether the line passes between the corner points of the pick window on the extended diagonal which passes between the areas containing the line end points.

Step 10 This step ensures that the line from (x_1, y_1) to (x_2, y_2) points clockwise around the pick window. This is done by computing a shift left of register F and ANDing the result with register G. If the result of the AND is equal to 00000000, then the ends are swapped as in Step 6 above and the AND of the Shift Left of register F with register G recomputed. The AND is stored in register K.

Step 11 This step sets a register y with a value of y_B or y_T depending upon the first and fourth bits of Register F.

If either bit is '1' then $y = y_B$
If neither bit is '1' then $y = y_T$

Step 12 This step sets a register x with a value x_R or x_L depending upon the first and second bits of Register F.

If either bit is '1' then $x = x_R$
If neither bit is '1' then $x = x_L$

Step 13 If the expression:
(NOT(right side(x_1, y_1, x_2, y_2, x, y)))
is TRUE then Step 2 is entered otherwise Step 4 is

entered.

The function left side (px, py, qx, qy, rx, ry) is performed by the following routine:

IF(xp-xr)*(yq-yp)>(yp-yr)*(xq-xp) THEN left side = TRUE
OTHERWISE left side = FALSE.

The function right side (px, py, qx, qy, rx, ry) is performed by the following routine:

IF (xp-xr)*(yq-yp)<(yp-yr)*(xq-xp) THEN right side = TRUE
OTHERWISE right side = FALSE.

The method described above can be implemented in either hardware or software. In an interactive display system such as illustrated in FIG. 3, the processor 32 performs the method under the control of a central program formed in the read only store 34. In other embodiments the control program may reside on a diskette and read into a portion of the random access store 35 when required.

In a display control unit such as illustrated at 31 a general control program residing in ROS 34 controls the position of the cursor on the screen 30 in response to input from devices 38 or 37 and defines the correlation or pick window in response to commands received from the Keyboard 37. The processor 32 under control of the control program performs the process for correlating the lines with the pick window. Thus in the following claims both the first, second and third means operating under control of the control program may be the processor 32, parts thereof or the processor in combination with the registers of the random access store 35.

CLAIMS

1. A method of correlating on straight lines with a correlation window in an interactive display system comprising the steps of:

a) defining a correlation window on a display area and dividing the rest of the display area into eight surrounding regions.

b) determining which lines stored as coordinates in a display list buffer can be trivially accepted or rejected by computing the configuration of regions in which the end points lie.

c) determining for each line not trivially accepted or rejected, on which side of the line one or two selected corner points of the correlation window lie and consequently whether or not it intersects the correlation window.

2. A method as claimed in claim 1 in which step (c) includes the step (d) of determining whether the line has end points in diagonally opposite corner regions in which case the selected corner points are either or both of the corner points on the extended diagonal of the correlation window.

3. A method as claimed in claim 1 in which the step (c) includes the step (e) of determining whether the line has at least one end point in a region adjacent to a side of the correlation window in which case the selected corner point is the intersection of the region and the area formed by combining the region in which the other end point lies and any adjacent middle region or regions.

4. A method as claimed in claim 3 in which when the other end point lies in a middle region it is combined with itself and when the other end point lies in a corner region it is combined with adjacent middle region having an intersection with the region in which the first end point lies.

5. An interactive display system comprising a display screen and a control unit including a processor device, a read only memory for storing a control program and a random access store having at least register regions and a display test buffer store,

first means under the control of the control program for defining a correlation window on a display area the rest of the display area into eight surrounding regions,

second means under the control of the control program for determining which lines stored as coordinates in the display list buffer can be trivially accepted or rejected by comparing the regions in which the end points lie,

third means under the control of the control program for determining for each line not trivially accepted or rejected, on which side of the line one or two selected corner points of the correlation window lie and consequently whether or not it intersects the correlation window.

2 00110	3 00100	4 01100
5 00010	1 00000	6 01000
7 10011	8 10001	9 11001

10

FIG. 1

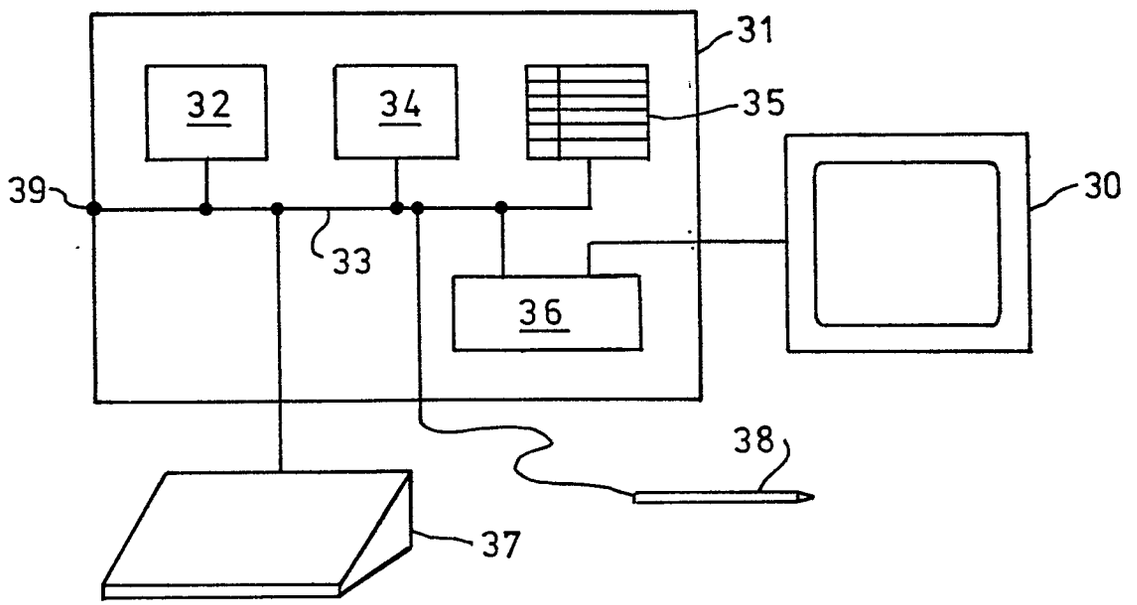


FIG. 3

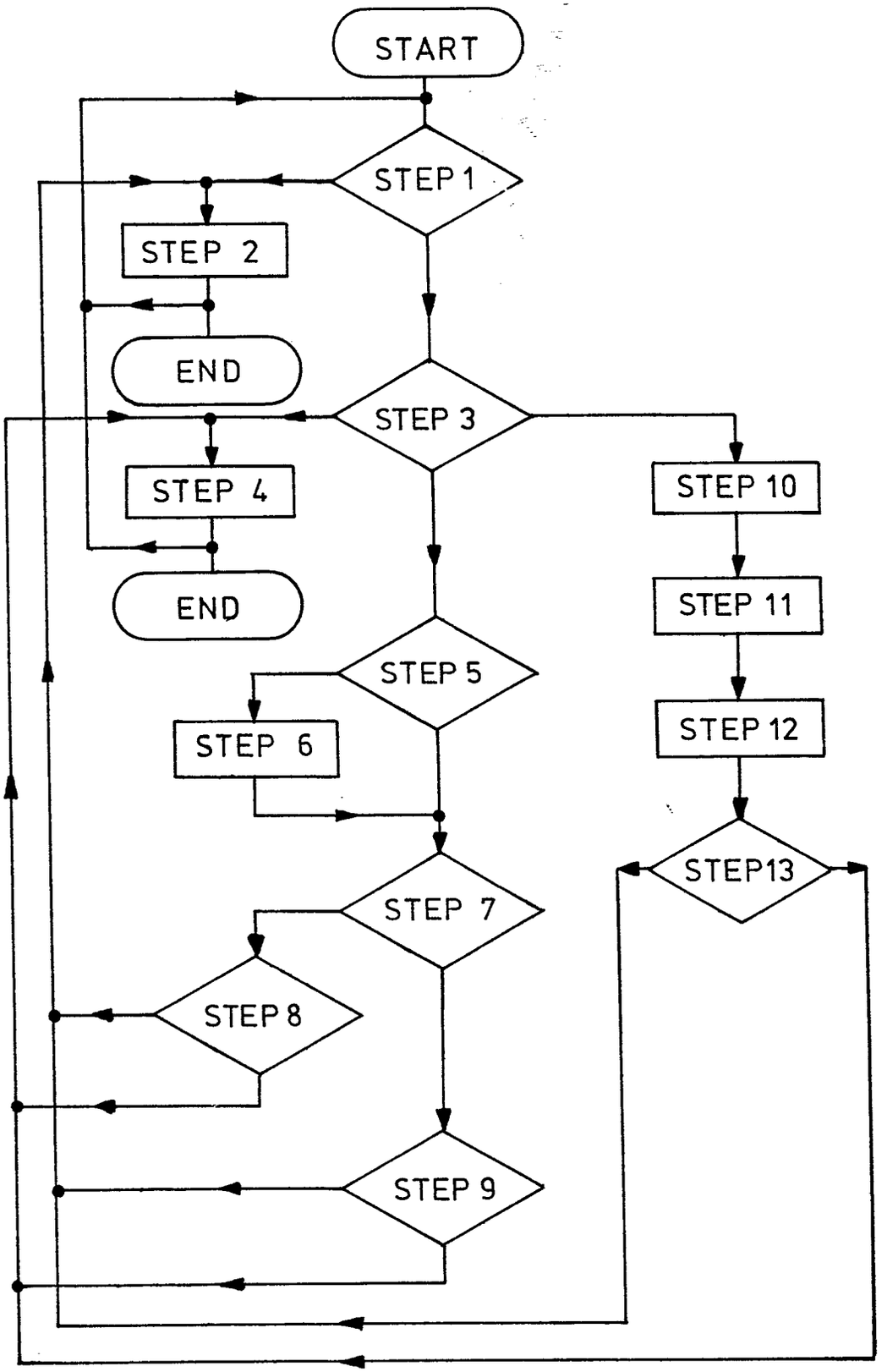

START
STEP 1
STEP 2
END
STEP 3
STEP 4
END
STEP 10
STEP 11
STEP 5
STEP 6
STEP 12
STEP13
STEP 7
STEP 8
STEP 9

FIG. 2