|  |  |
| --- | --- |
| **Name:** | Yash Ravindra Kerkar |
| **Roll No:** |  |
| **Class/Sem:** | SE/III |
| **Experiment No.:** | 1 |
| **Title:** | Digital Differential Analyzer Line Drawing Algorithm |
| **Date of Performance:** |  |
| **Date of Submission:** |  |
| **Marks:** |  |
| **Sign of Faculty:** |  |

# Experiment No. 1

|  |  |
| --- | --- |
| **Aim** | Write a program to implement Digital Differential Analyzer Line Drawing Algorithm in C. |
| **Objective** | To implement DDA line drawing algorithm for drawing a line segment between two points A (x1, y1) and B (x2, y2) |
| **Theory** | DDA stands for Digital Differential Analyzer. It is an incrementalmethod of scan conversion of line. In this method calculation is performed at each step but by using results of previous steps.  **DDA Working Mechanism**  Case 1: Slope < 1 (m<1)(theta<45)x coordinates increase fast  dx is set to unit interval dx=1; dy is computed; m = dy/dx therefore; dy = m Calculation for next pixel for line processed from left to right  xk+1 = x k + dx = xk + 1 yk+1 = yk + dy = yk + m  Calculation for next pixel for line processed from right to left xk+1 = x k + dx = xk - 1  yk+1 = yk + dy = yk - m  Case 2: Slope > 1 (m>1) theta>45 y coordinates increase fast  dy is set to unit interval dy=1; dx is computed; m = dy/dx therefore; dx = 1/m Calculation for next pixel for line processed from left to right  xk+1 = x k + dx = xk + 1/m yk+1 = yk + dy = yk + 1  Calculation for next pixel for line processed from right to left xk+1 = x k + dx = xk - 1/m  yk+1 = yk + dy = yk - 1  Case 3: Slope =1 (m=1) theta=45  dx and dy is set to unit interval dx=1 and dy=1  Calculation for next pixel for line processed from left to right xk+1 = x k + dx = xk + 1  yk+1 = yk + dy = yk + 1  Calculation for next pixel for line processed from right to left xk+1 = x k + dx = xk - 1  yk+1 = yk + dy = yk – 1 |

|  |  |
| --- | --- |
| **Algorithm** | **DDA Algorithm:**  **Step1:** Start Algorithm  **Step2:** Declare x1,y1,x2,y2,dx,dy,x,y as integer variables.  **Step3:** Enter value of x1,y1,x2,y2. **Step4:** Calculate dx = x2-x1 **Step5:** Calculate dy = y2-y1 **Step6:** If ABS (dx) > ABS (dy)  Then step = abs (dx) Else  **Step7:** xinc=dx/step  yinc=dy/step assign x = x1 assign y = y1  **Step8:** Set pixel (x, y)  **Step9:** x = x + xinc  y = y + yinc  Set pixels (Round (x), Round (y)) **Step10:** Repeat step 9 until x = x2 **Step11:** End Algorithm  **Code:**  #include <graphics.h>  #include <stdio.h>  #include <math.h>  #include <dos.h>  int main()  {  float x, y, x1, y1, x2, y2, dx, dy, step;  int i, gd = DETECT, gm;  initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");  printf("\nEnter the x-coordinate of the first point:");  scanf("%f", &x1);  printf("\nEnter the y-coordinate of the first point:");  scanf("%f", &y1);  printf("\nEnter the x-coordinate of the second point:");  scanf("%f", &x2);  printf("\nEnter the y-coordinate of the second point:");  scanf("%f", &y2);  dx = abs(x2 - x1);  dy = abs(y2 - y1);  if (dx > dy)  {  step = dx;  }  else  {  step = dy;  }  dx = dx / step;  dy = dy / step;  x = x1;  y = y1;  i = 1;  while (i <= step)  {  putpixel(x, y, 14);  x = x + dx;  y = y + dy;  i = i + 1;  delay(100);  }  getch();  closegraph();  return 0;  } |
| **Output** |  |
| **Conclusion:** |  |

In this practical, we learned how to draw a line on the screen using the Bresenham Line Drawing algorithm. The program takes two sets of coordinates (x1, y1) and (x2, y2) as input and calculates the intermediate points to draw a line between them.