#include <stdio.h>

Expt No.: 06

Expt Name : Scanline Polygon Fill

Student Name : Akashdeep Singh

Student Roll No.:16102A0035

#include <conio.h>

#include <math.h>

#include <graphics.h>

typedef struct Matrix

{

int rows, columns;

float \*\*matrix;

}Matrix;

void ScanlineFill();

void AcceptPolygon(Matrix \*polygon);

void InitMatrix(Matrix \*m,int r,int c);

void DrawPolygon(Matrix \*polygon);

void InitEdges(Matrix \*polygon);

void MinMax(Matrix \*polygon, int col, int lb, int ub, int \*min, int \*max);

void DPQS(Matrix \*m,int col, int low, int high);

void Partition(Matrix \*m, int col, int low, int high, int \*lp, int \*rp);

void Swap(float \*a,float \*b);

void InitGraph();

main()

{

ScanlineFill();

while(!kbhit()){delay(100);}

}

void ScanlineFill()

{

Matrix polygon;

int boxLeft, boxRight, boxHigh, boxLow;

int i,j,k;

int edgeXat\_i,lim\_y\_up,lim\_y\_down;

Matrix xlist;

AcceptPolygon(&polygon);

InitGraph();

DrawPolygon(&polygon);

InitEdges(&polygon);

MinMax(&polygon,0,0,polygon.rows-1,&boxLeft,&boxRight);

MinMax(&polygon,1,0,polygon.rows-1,&boxLow,&boxHigh);

InitMatrix(&xlist,polygon.rows-1,1);

for(i=boxLow+1,k=0;i<boxHigh;i++,k=0)

{

for(j=0;j<polygon.rows-1;j++)

{

if((int)polygon.matrix[j][0] == (int)polygon.matrix[j+1][0])

edgeXat\_i = polygon.matrix[j][0];

else if((int)polygon.matrix[j][1] == (int)polygon.matrix[j+1][1])

edgeXat\_i = polygon.matrix[j][0]<polygon.matrix[j+1][0]?polygon.matrix[j][0]:polygon.matrix[j+1][0];

else

edgeXat\_i = (i - polygon.matrix[j][3])/polygon.matrix[j][2];

if(edgeXat\_i>=boxLeft && edgeXat\_i<=boxRight )

{

lim\_y\_up = polygon.matrix[j][1]<polygon.matrix[j+1][1]?polygon.matrix[j][1]:polygon.matrix[j+1][1];

lim\_y\_down = polygon.matrix[j][1]>polygon.matrix[j+1][1]?polygon.matrix[j][1]:polygon.matrix[j+1][1];

if(lim\_y\_up<=i && i<=lim\_y\_down)

xlist.matrix[k++][0] = edgeXat\_i;

}

DPQS(&xlist,0,0,k-1);

}

for(j=0;j<k;j+=2)

line(xlist.matrix[j][0]+1,i,xlist.matrix[j+1][0],i);

}

}

void AcceptPolygon(Matrix \*polygon)

{

int i,nv;

printf("Give number of vertices: ");

scanf("%d",&nv);

InitMatrix(polygon,nv+1,4);

for(i=0;i<nv;i++)

{

printf("Give coordinates of vertex(%d): ",i+1);

scanf("%f%f",&polygon->matrix[i][0],&polygon->matrix[i][1]);

}

polygon->matrix[i][0] = polygon->matrix[0][0];

polygon->matrix[i][1] = polygon->matrix[0][1];

}

void InitMatrix(Matrix \*m,int r,int c)

{

int i;

m->rows = r;

m->columns = c;

m->matrix = (float\*\*)malloc(r\*sizeof(float\*));

for(i=0;i<r;i++)

m->matrix[i] = (float\*)malloc(c\*sizeof(float));

}

void DrawPolygon(Matrix \*polygon)

{

int i;

int x1,x2,y1,y2;

for(i=0;i<polygon->rows-1;i++)

{

x1 = polygon->matrix[i][0];

y1 = polygon->matrix[i][1];

x2 = polygon->matrix[i+1][0];

y2 = polygon->matrix[i+1][1];

line(x1,y1,x2,y2);

}

}

void InitEdges(Matrix \*polygon)

{

int i;

float x1,y1,x2,y2;

float M,C;

for(i=0;i<polygon->rows-1;i++)

{

x1 = polygon->matrix[i][0];

y1 = polygon->matrix[i][1];

x2 = polygon->matrix[i+1][0];

y2 = polygon->matrix[i+1][1];

// if its a vertical edge

if((int)x2 == (int)x1)

M=C=0;

//if its a horizontal edge

else if((int)y1 == (int)y2)

{

M=0;

C=y2;

}

//any other edge

else

{

M = (y2-y1)/(x2-x1);

C = y2 - M \* x2;

}

polygon->matrix[i][3] = C;

polygon->matrix[i][2] = M;

}

}

void MinMax(Matrix \*polygon, int col, int lb, int ub, int \*fmin, int \*fmax)

{

int i;

int min,max;

min = polygon->matrix[lb][col];

max = polygon->matrix[lb][col];

for(i=lb;i<=ub;i++)

{

if(polygon->matrix[i][col]<min)

min = polygon->matrix[i][col];

if(polygon->matrix[i][col]>max)

max = polygon->matrix[i][col];

}

\*fmin = min;

\*fmax = max;

}

void DPQS(Matrix \*m,int col, int low, int high)

{

int lp, rp;

if(low<high)

{

Partition(m,col, low, high, &lp, &rp);

DPQS(m,col,low,lp-1);

DPQS(m,col,lp+1,rp-1);

DPQS(m,col,rp+1,high);

}

}

void Partition(Matrix \*m,int col, int low, int high, int \*lp, int \*rp)

{

int down1,down2,up,p,q;

if(m->matrix[low][col]>m->matrix[high][col])

Swap(m->matrix[high],m->matrix[low]);

down1=down2=low+1;

up=high-1;

q=m->matrix[low][col];

q=m->matrix[high][col];

while(down2<=up)

{

if(m->matrix[down2][col]<p)

{

Swap(m->matrix[down2],m->matrix[down1]);

down1++;

}

else if(m->matrix[down2][col]>=q)

{

while(m->matrix[up][col]>q && down2<up)

up--;

Swap(m->matrix[down2],m->matrix[up]);

up--;

if(m->matrix[down2][col]<p)

{

Swap(m->matrix[down2],m->matrix[down1]);

down1++;

}

}

down2++;

}

down1--;

up++;

Swap(m->matrix[down1],m->matrix[low]);

Swap(m->matrix[high],m->matrix[up]);

\*lp = down1;

\*rp = up;

}

void Swap(float \*a,float \*b)

{

float \*temp;

temp = a;

a=b;

b=temp;

}

void InitGraph()

{

int gd,gm;

gd=gm=DETECT;

initgraph(&gd,&gm,"C:\\TC\\BIN");

}





