Programming Exercise 3 : scan conversion of polygon

Introduction

In this exercise, you will implement the sweep line algorithm for scan conversion of polygons. Before starting on this programming exercise, we strongly recommend watching the lectures.

To get started with the exercise, you will need to download the starter codes. There is a skeleton file "***ScanPolygon.cpp***" which provided all nessary code for you to start, you just need to finish the function *scanPolygon()* to scan conversion of a polygon .

We implemented a simple code called scanRectange() for *scan conversion of rectange* to warm you up for the concept of scan conversion of polygon. **Your task** is to implement weep line algorithm for scan conversion of polygons in the function *scanPolygon()*.

To run these programs ,the glut enviroment should be setup.You can get the glut from glut-3.7.6-bin.zip (117 KB)(http://user.xmission.com/~nate/glut/glut-3.7.6-bin.zip) ,unzip it and copy three files to different locations on your window as following:

- glut.h – This is the file you'll have to include in your source code. The common place to put this file is in the gl folder which should be inside the include folder of your system. for example ,your vc2010 include folder "*c:\program files\Microsoft Visual Studio 10.0\VC\include\gl*" or "*c:\program files(x86)\Microsoft Visual Studio 10.0\VC\include\gl*" in(my conputer
- glut32.lib (Windows version) – This file must be linked to your application so make sure to put it your lib folder. example ,your vc2010 lib folder "*c:\program files\Microsoft Visual Studio 10.0\VC\lib*" or "*c:\program files(x86)\Microsoft Visual Studio 10.0\VC\lib*" in(my conputer
- glut32.dll (Windows) – You could place the dll file in your exe's folder.for example,the sustem folder: "*c:\windows\system32*" or "*c:\windows\sysWOW64*" in my conputer

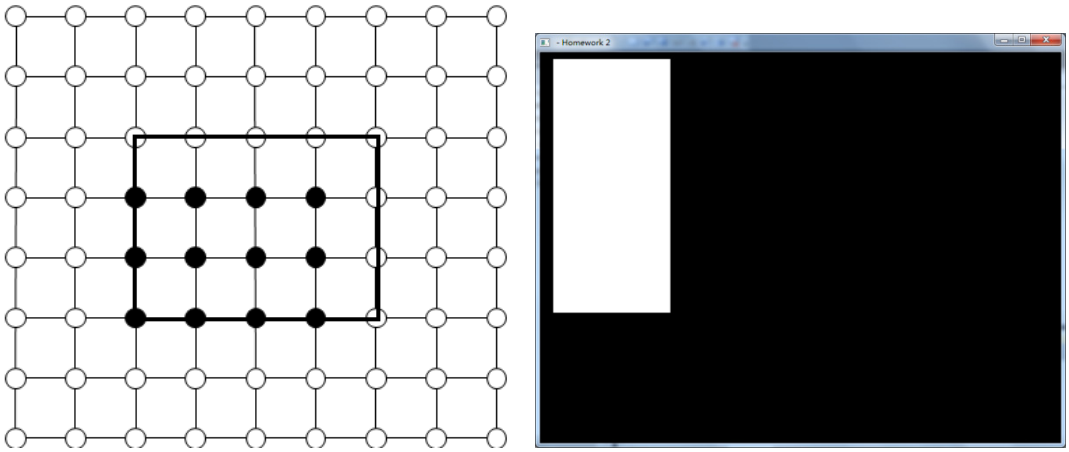
Scan conversion of Rectanges

The scan conversion of rectanges is simple (see figure 1.a)): we just need to set pixels bounded by left ,right,bottom and top edge of the rectange with given color.The code is as following:

```
for(int y = y0;y<y1;y++)
    for(int x = x0;x<x1;x++)
        setpixel(x,y,R,G,B);
```

When you run the starter code " ***ScanPolygon***.*cpp* " ,the scanRectange

is invoked in the dispaly function display() to fill a rectange from (20,10) to (200,400) and the result will be similar to figure 1.b).



a) fill pixels in a rectange b) a rectange from (20,10) to (200,400)

figure 1 scan conversion of rectanges

Please note that the origin of the screen coordinate frame is located on the top left.

Scan conversion of Polygons

Your task in this programming exercise is to finish the sweep line scan conversion of a polygon in the function s*canPolygon()* .

For each sweep line intersecting with the polygon,the sweep line scan conversion algorithm find all intersectiong point of the edges of the polyong and the sweep line and sort these interstion points to get the segment of the sweep line lying in the polygon and fill these segments with given color.For example,infigure 2,the y=2 sweep line intersecting with the polygon has two segments lying in the polygon.

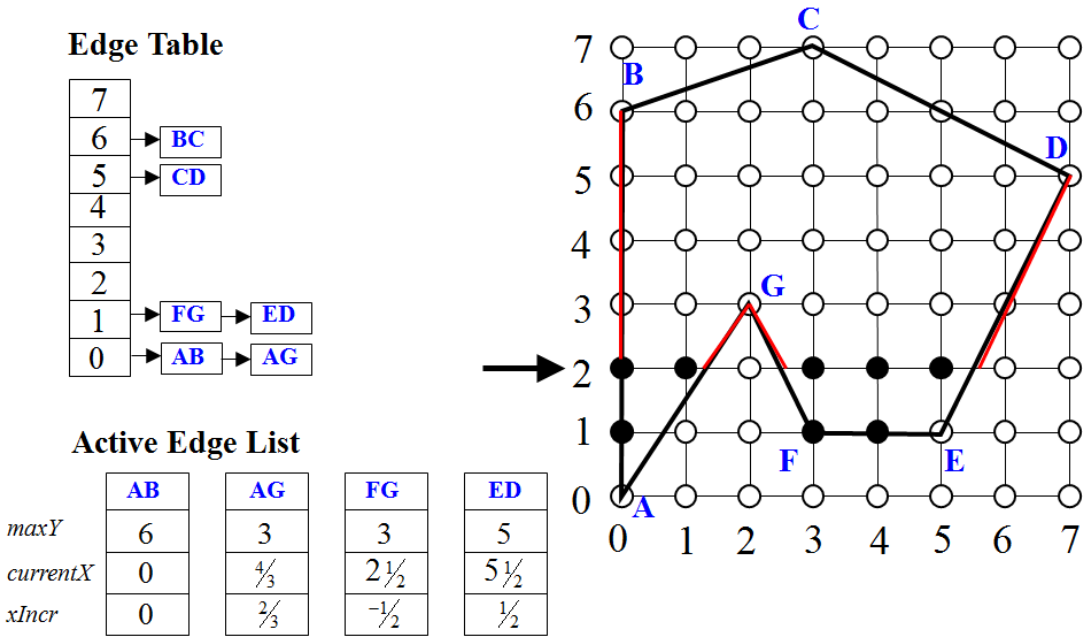


figure2 Sweep line algorithm for scan conversion of polygons

A data structure called "Edge" is used to represent an edge of a polygon which stores its y-coordinate of its up vertex,the x-coordinate of the intersect point with current sweep line and the reciprocal of ite slope:

```
typedef struct {
    int maxY;
    float currentX,xInc;
} Edge;
```

The sweep line use a data structure called "Edge Table" to store all non-horizontal edges of the polygon and edges with their lower vertices lying on the same sweep line are put into a single edge list in the "Edge Table". You can use the vector to represent the "Edge Table" (see figure 2):

```
typedef vector<Edge>    EdgeList;
vector< EdgeList >    edgeTable; //store all non- horizontal edges
```

A variable called "Active Edge List" to store all edge intersecting with the current sweep line.

```
EdgeList activeEdgeList;
```

The sweep line scan conversion algorithm could be describe as following:

```
line y = 0
While (y < height )
    Add edges to Active Edge List from Sorted Edge Table starting at line y
    Remove edges that end at line
    Fill pixels
    Increment x-values on edges in Active Edge List
    Increment line y
```

We have implemented some code for you in the ScanPolygon( ) function and left some code for you to implement. For example,you should implement " *Increment x-values on edges in Active Edge List* " to update the x-coordinates of intersection points from sweep line y to sweep line y+1 as following (see figure 3):

```
x = x+ 1/m;    //where m is the slope of the edge line.
```

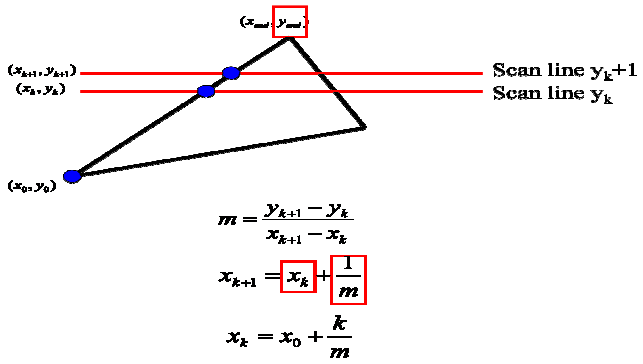


figure 3 incremental evaluate the x-coordinate of the intersection points

We use a helper function *intputPolygon ( )* to help you input the coordinates of vertices of a polygon in the *main( )* function. After you finished the ScanPolygon( ),you should uncommet the " *intputPolygon(polygon);"* to set the vertices of a polygon. For example ,if you input a polygon like the figure 4.left), you will see the conversed

polygon on the scree like the figure 4.right).

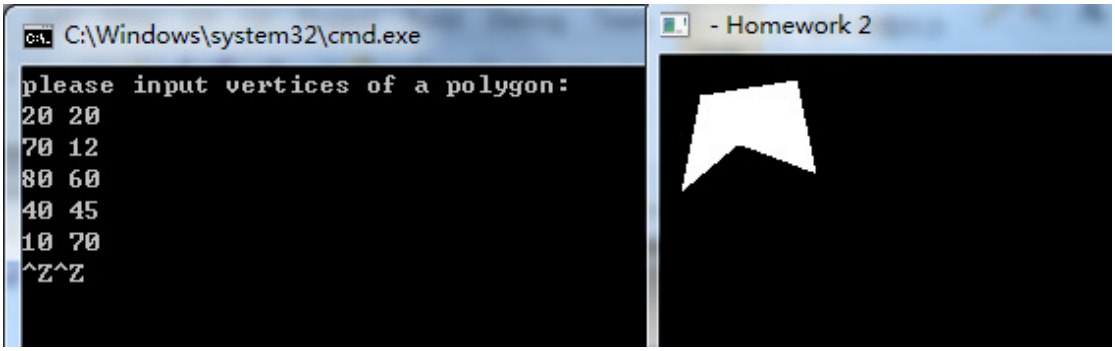


figure 4 input polygon and scan conversion of it.

Please note that the origin of the screen coordinate frame is located on the top left.

All parts of this programming exercise are due 2012/10/24 PM at 23:59