| | |
|---|---|
| VIT Vidyalankar Institute of Technology<br>ACCREDITED A+ BY NAAC | **Department of Computer Engineering** |

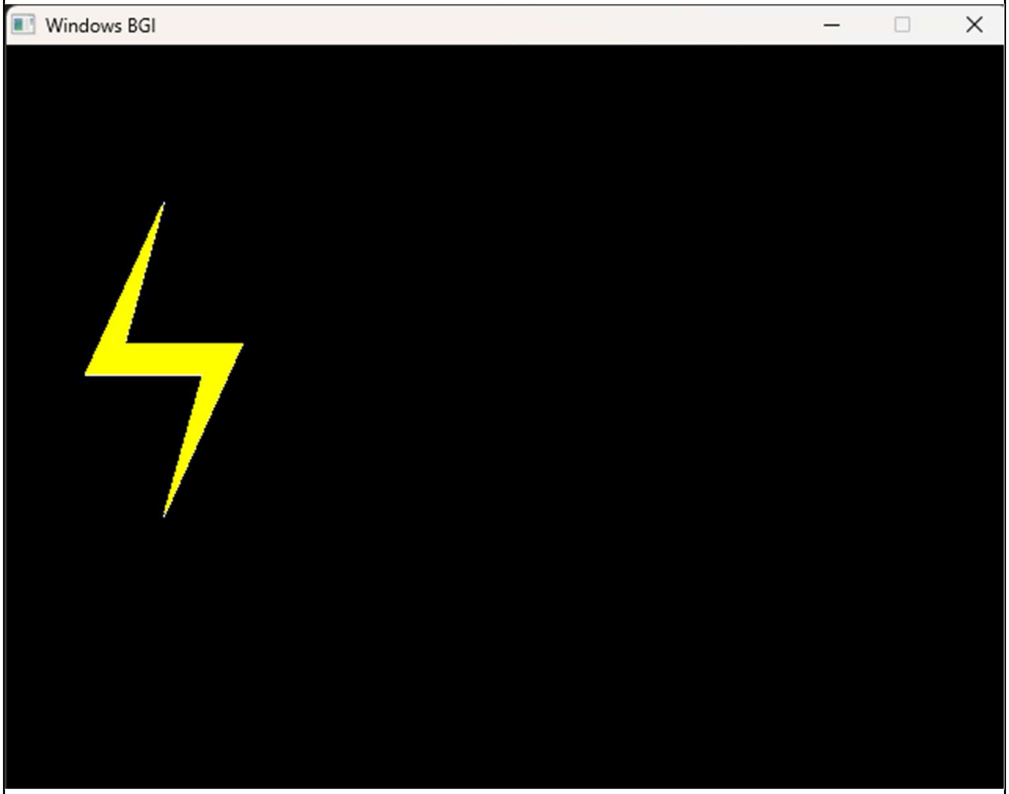
| | |
|---|---|
| Semester | S.E. Semester IV – CMPN |
| Division & Batch | Div C Batch 1 |
| Subject | Computer Graphics |
| Laboratory Teacher: | Divya Nimbalkar |
| Laboratory | M310A |

| | | |
|---|---|---|
| Student Name | Atharva Sheramkar | |
| Roll Number | 24102B0069 | |
| Grade and Subject Teacher's Signature | | |

| | | |
|---|---|---|
| Experiment Number | 5 | |
| Experiment Title | **Implementation of Scan Line Algorithm for filling objects** | |
| Problem Statement | **Implementation of Scan Line Algorithm for filling objects** | |
| Resource s / Apparatus Required | Hardware: Desktop | Software: Dev C++ |

Code

```
#include <graphics.h>
#include <stdio.h>
#include <conio.h>

void scanfill(int x[], int y[], int n)
{
    int i, j, k, temp, y_scan;
    int inter_x[20];
    int ymax = 0, ymin = 480;

    for(i = 0; i < n; i++) {
        if(y[i] > ymax) ymax = y[i];
        if(y[i] < ymin) ymin = y[i];
    }

    for(y_scan = ymax; y_scan >= ymin; y_scan--)
    {
        k = 0;
        for(i = 0; i < n; i++)
        {
            int next = (i + 1) % n;
            int x1 = x[i], y1 = y[i];
            int x2 = x[next], y2 = y[next];

            if(y1 != y2)
            {
                if ((y_scan >= y1 && y_scan < y2) || (y_scan >= y2 && y_scan < y1))
                {
                    inter_x[k] = x1 + (float)(y_scan - y1) * (x2 - x1) / (y2 - y1);
                    k++;
                }
            }
        }

        for(i = 0; i < k - 1; i++) {
            for(j = 0; j < k - 1 - i; j++) {
                if(inter_x[j] > inter_x[j + 1]) {
                    temp = inter_x[j];
                    inter_x[j] = inter_x[j + 1];
                    inter_x[j + 1] = temp;
                }
            }
        }

        setcolor(YELLOW);
        for(i = 0; i < k; i += 2) {
            if(i + 1 < k) {
                line(inter_x[i], y_scan, inter_x[i + 1], y_scan);
            }
        }
        delay(5);
    }
}

int main() {
    int gd = DETECT, gm;

int x_coord[] = {100,75,150,100,125,50,100};
    int y_coord[] = {100,190,190,300,210,210,100};
    int n = 7;

    initgraph(&gd, &gm, (char*)"");

    setcolor(WHITE);
    for(int i = 0; i < n; i++) {
        line(x_coord[i], y_coord[i], x_coord[(i + 1) % n], y_coord[(i + 1) % n]);
    }

    scanfill(x_coord, y_coord, n);

    getch();
    closegraph();
    return 0;
}
```



| | |
|---|---|
| Conclusion | In this experiment, the **Scan-Line Fill Algorithm** was successfully implemented to populate the interior of a polygon by intersecting horizontal scan lines with polygon edges. By maintaining an **Active Edge Table (AET)** and sorting intersections by their x-coordinates, the algorithm efficiently determines "inside-outside" spans for pixel shading. This approach ensures mathematical precision in filling complex or concave shapes while minimizing redundant calculations through the use of **edge coherence**, ultimately providing a robust solution for rendering solid geometric primitives. |