Graphics Programming

Xi Chen

Modified based on slides from Ed Angel
• Computer all the points first and put them into an array
  - add(p1, p2)
  - scale(s, u)
  - mix(u, v, s)
  - Math.random()

• Display all the points by calling gl.drawArrays(gl.POINTS, start, length) function.
Primitives

- `gl.POINTS`
- `gl.LINES`
- `gl.LINE_STRIP`
- `gl.LINE_LOOP`

**FIGURE 2.7** Point and line-segment types.
Primitives

• gl.TRIANGLES
• gl.TRIANGLE_STRIP
• gl.TRIANGLE_FAN
Primitives

**FIGURE 2.13** Points and triangles types.

**FIGURE 2.14** Triangle strip and triangle fan.
Examples

• \(\text{Desktop}\backslash \text{CS315}\backslash \text{Example}\backslash \text{week2}\backslash \text{primitive.html} \)

• Change the size of points
• \(\text{gl\_PointSize} = 5.0; \)
Draw circle (filled with red)
Draw circle (filled with red)

numOfPoints = 6
numOfPoints = 12

Draw circle (filled with red)
Draw circle (filled with red)

numOfPoints = 24
Draw circle (filled with red)

numOfPoints = 50
Draw circle (filled with red)

numOfPoints = 500
Draw circle (filled with red)

numOfPoints = 500
Generate points on the circle

- Points on the circle
  \[ p = (\text{radius} \times \cos(\theta), \text{radius} \times \sin(\theta)) \]
  \[ \theta = \frac{2 \times \pi}{\text{numOfPoints}} \]
  - Math.sin(radians)
  - Math.cos(radians)
  - Math.PI
Generate points on the circle

```javascript
// generate points
function GeneratePoints(){
    var radius = 1; // Radius of the circle

    // Push all points into points[]
    var angle = Math.PI*2/NumberOfPoints_Circle;
    for(var i =0; i<NumberOfPoints_Circle; ++i){
        var x = radius*Math.cos(angle*i);
        var y = radius*Math.sin(angle*i);
        points.push(vec2(x, y));
    }
}
```
The Siperginski gasket (Polygons)

Triangles after 3 subdivisions
The Siperpinski gasket (Polygons)

count = 3
The Siperpinski gasket (Polygons)

Triangles after 1 subdivision
count = 3-1 = 2
The Sierpinski gasket (Polygons)

Triangles after 3 subdivisions
count = 1 - 1 = 0
The Siperpinski gasket (Polygons)

```javascript
function divideTriangle(a, b, c, count)
{
    // Check for end of recursion
    if(count === 0){
        // Push a, b, c to points
        points.push(a, b, c);
    }else{
        var ab = mix(a, b, 0.5);
        var ac = mix(a, c, 0.5);
        var bc = mix(b, c, 0.5);

        count--;

        divideTriangle(a, ab, ac, count);
        divideTriangle(b, ab, bc, count);
        divideTriangle(c, ac, bc, count);
    }
}
```