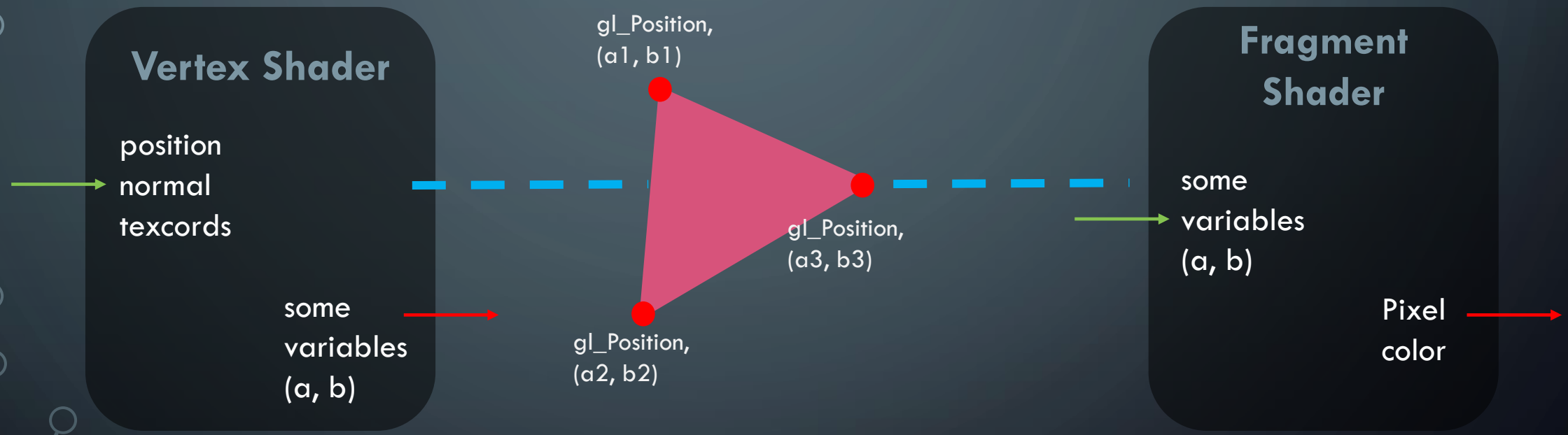
SHADERS

TUTORIAL

Types of Shaders:

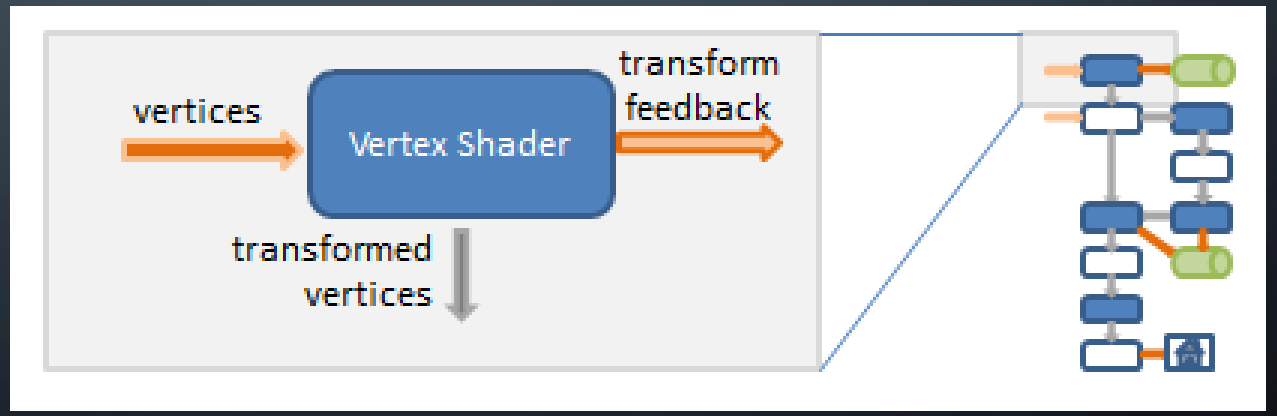
- Vertex Shader (only mandatory shader in pipeline)
- Fragment Shader (widely used)
- Geometry Shader (optional)
- Tessellation Shader (optional)

Basic shader pipeline

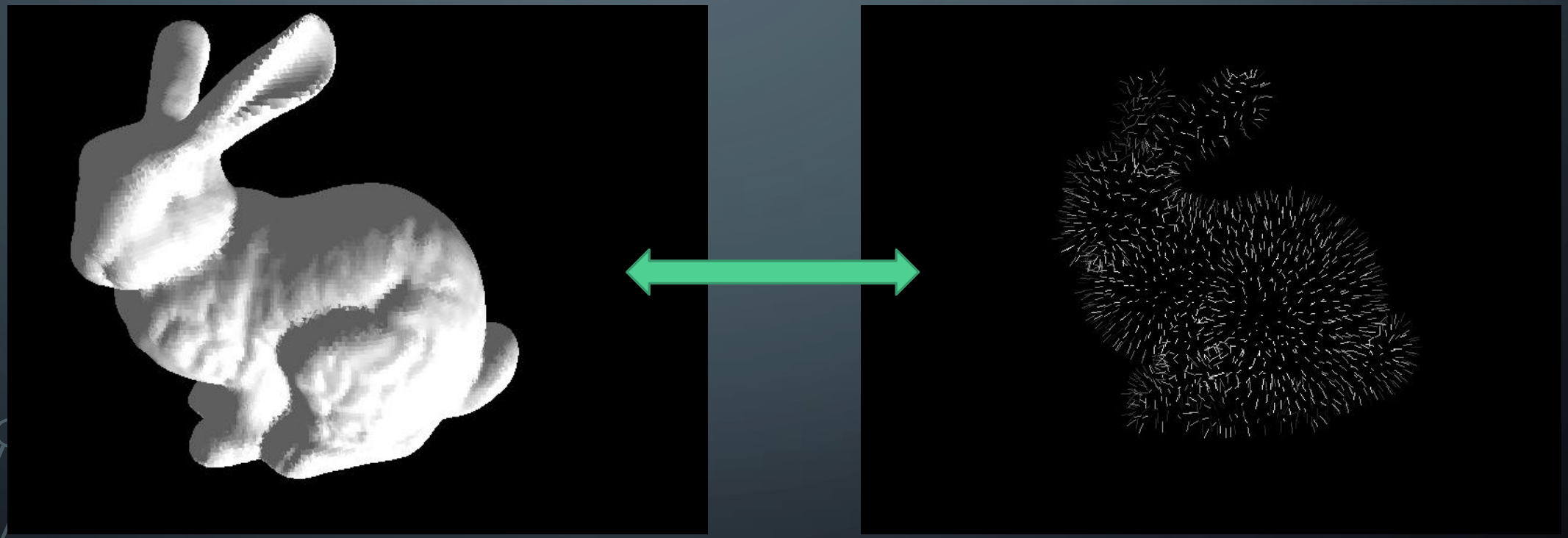


Need for other shaders:

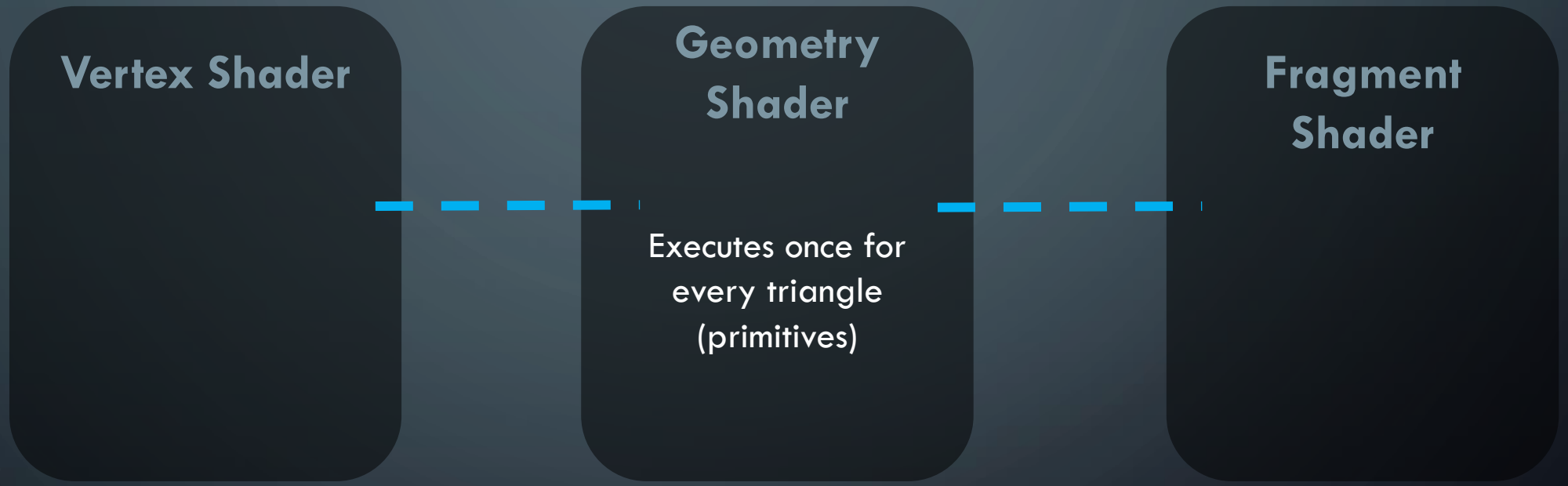
- VS cannot create additional geometry
- VS cannot access data of other vertices
- Additional capabilities for real-time interface



Geometry Shader



Geometry Shader



gl_Position,
(a1, b1)

Geometry Shader

a[], b[]
(variable
for each
primitive)

Executes once for every triangle
(primitives)

in (points,
lines,
triangles etc.)

out (points,
lines,
triangles etc.)

Cant output
mixture

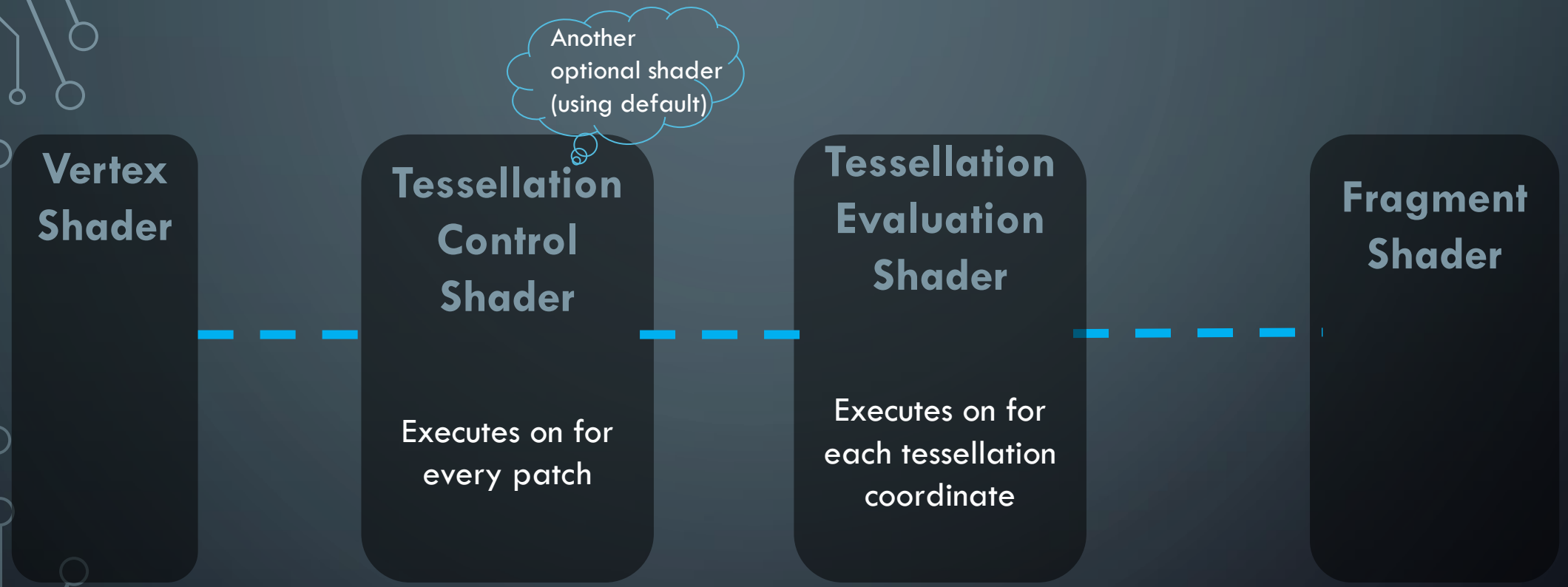
BASIC GEOMETRY SHADER

```
#version 330 core
layout (points) in;
layout (line_strip, max_vertices = 2) out;

void main() {
    gl_Position = gl_in[0].gl_Position + vec4(-0.1, 0.0, 0.0, 0.0);
    EmitVertex();

    gl_Position = gl_in[0].gl_Position + vec4( 0.1, 0.0, 0.0, 0.0);
    EmitVertex();

    EndPrimitive();
}
```

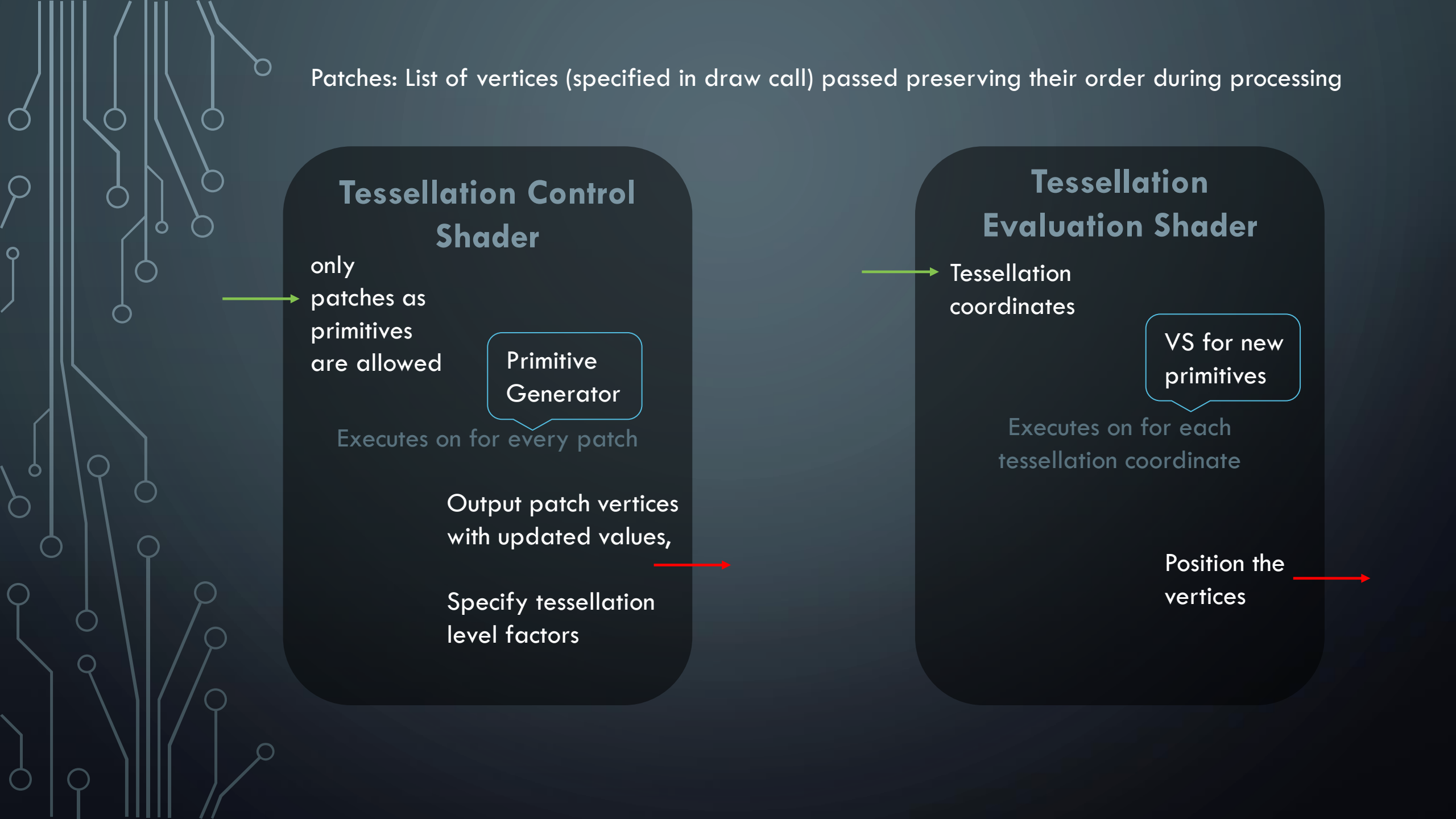
- ‘in’ identifier depends on primitive declared in program
- ‘out’ can be set as any primitive
- EmitVertex() add the gl_Position vector to output primitive.
- EndPrimitive() combine all the vertices to render output.

Tessellation Shader

Tessellation Shader



Patches: List of vertices (specified in draw call) passed preserving their order during processing



THANK YOU!