




1




Instancing



Oregon State University
Mike Bailey
mjb@cs.oregonstate.edu



This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License](https://creativecommons.org/licenses/by-nc-nd/4.0/)



Oregon State University
Computer Graphics

Instancing.pptx

mjb - December 27, 2022

Instancing – What and why?


2

- Instancing is the ability to draw the same object multiple times
- It uses all the same vertices and the same graphics pipeline data structure each time
- It avoids the overhead of the program asking to have the object drawn again, letting the GPU/driver handle all of that

```
vkCmdDraw( CommandBuffers[nextImageIndex], vertexCount, instanceCount, firstVertex, firstInstance );
```

BTW, when not using instancing, be sure the **instanceCount** is 1, not 0 !

But, this will only get us multiple instances of identical objects drawn on top of each other. How can we make each instance look differently?



Oregon State University
Computer Graphics

mjb - December 27, 2022

Making each Instance look differently -- Approach #1

3

Use the built-in shader variable `gl_InstanceIndex` to define a unique display property, such as position or color.

`gl_InstanceIndex` starts at 0

In the vertex shader:

```

layout( std140, set = 0, binding = 0 ) uniform sporadicBuf
{
    int     uMode;
    int     uUseLighting;
    int     uNumInstances;
} Sporadic;
...
void
main( )
{
    ...

    float DELTA      = 3.0;
    float s = sqrt( float( Sporadic.uNumInstances ) );
    float c = ceil( float(s) );
    int cols = int( c );
    int fullRows = gl_InstanceIndex / cols;
    int remainder = gl_InstanceIndex % cols;

    float xdelta = DELTA * float( remainder );
    float ydelta = DELTA * float( fullRows );
    vec3 vColor = vec3( 1., float( (1.+ gl_InstanceIndex) ) / float( Sporadic.uNumInstances ), 0. );

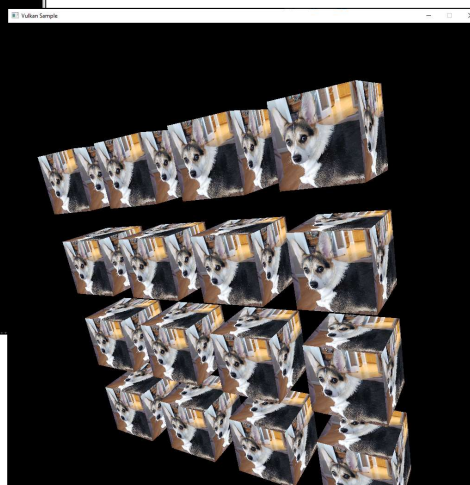
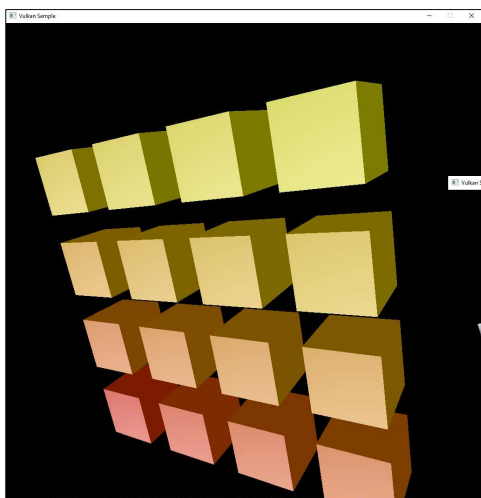
    vec4 vertex = vec4( aVertex.xyz + vec3( xdelta, ydelta, 0. ), 1. );
    gl_Position = PVM * vertex;
}

```



mjb - December 27, 2022

`uNumInstances = 16`



mjb - December 27, 2022

Making each Instance look differently -- Approach #2

5

Put the unique characteristics in a uniform buffer array and reference them

Still uses **gl_InstanceIndex**

In the vertex shader:

```
layout( std140, set = 4, binding = 0 ) uniform colorBuf
{
    vec3 uColors[1024];
} Colors;

out vec3 vColor;

...

int index = gl_InstanceIndex % 1024; // gives 0 - 1023
vColor = Colors.uColors[ index ];

...

vec4 vertex = vec4( aVertex.xyz + vec3( xdelta, ydelta, 0. ), 1. );
gl_Position = PVM * vertex;
```

