



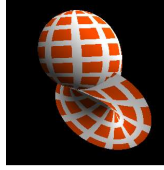
Geometric Morphing with the Vertex Shader



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/).




mjb - November 22, 2022

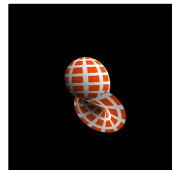
Morphing a Sphere into a Circle



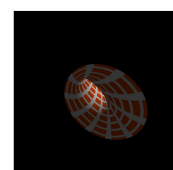
Blend = 0.00



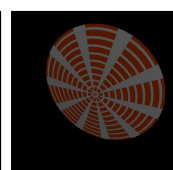
Blend = 0.25




Blend = 0.50



Blend = 0.75



Blend = 1.00



Computer Graphics

mjb - November 22, 2022

```

out vec2  vST;
out float vLightIntensity;
out vec3  vColor;
const float TWOPI = 2.*3.14159265;


// original model coords (sphere):
vec4 vertex0 = gl_Vertex;
vec3 norm0 = gl_Normal;

// circle coords:
vST = gl_MultiTexCoord0.st;
float radius = 1. - vST.t;
float theta = TWOPI * vST.s;
vec4 circle = vec4( radius*cos(theta), radius*sin(theta), 0., 1. );
vec3 circlenorm = vec3( 0., 0., 1. );
vST += vec2( OffsetS, OffsetT );

// blend:
vec4 theVertex = mix( vertex0, circle, Blend );
vec3 theNormal = normalize( mix( norm0, circlenorm, Blend ) );

// do the lighting:
vec3 tnorm = normalize( vec3( uNormalMatrix * theNormal ) );
vec3 LightPos = vec3( 5., 10., 10. );
vec3 ECposition = vec3( uModelViewMatrix * theVertex );
vLightIntensity = abs( dot( normalize( LightPos - ECposition ), tnorm ) );


vColor = gl_Color.rgb;
gl_Position = gl_ModelViewProjectionMatrix * theVertex;
    
```



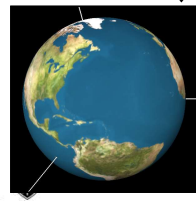
Computer Graphics

mjb - November 22, 2022


A possible vis application ??
What an interesting 2D view of Earth!




Original texture map



Mapped onto a Sphere



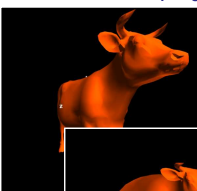
Morphed into a Circle



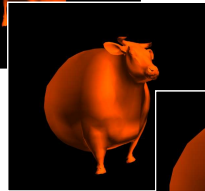
Computer Graphics

mjb - November 22, 2022


Morphing a Cow into a Sphere



```
vec4 vertex = gl_Vertex;
vertex.xyz *= 4. / length(vertex.xyz);
```



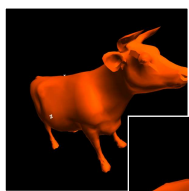
Note: the "face" in the sphere cow is there because the normals were not morphed into sphere normals – they were left as cow normals



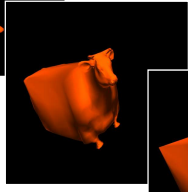
Computer Graphics

mjb - November 22, 2022


Morphing a Cow into a Cube



```
const float SIDE = 2.;
vec4 vertex = gl_Vertex;
vertex.xyz *= 4. / length(vertex.xyz);
vertex.xyz = clamp( vertex.xyz, -SIDE, SIDE );
```



Note: the "face" in the cube cow is there because the normals were not morphed into cube normals – they were left as cow normals



Computer Graphics

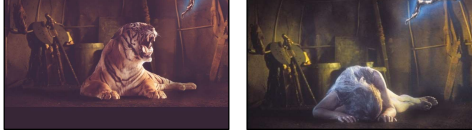
mjb - November 22, 2022

For you movie fans, what about "Real Morphing"?

7

"Real Morphing" involves interpolating key points from one object into key points in another. This flies in the face of graphics hardware's philosophy of dealing with one triangle and then getting rid of any record of it. Movies do this in software. We got away with it in our class because we knew the equation of a disk, a sphere, and a cube and so could interpolate in a vertex shader.

The first movie-morphing I remember seeing is from the fantasy movie *Willow*:



<https://www.youtube.com/watch?v=lKzbsDG58pc>

The "making of" video for this is here:

<https://www.youtube.com/watch?v=kxVwNIZDQJ0>



But, my nomination for #1 morphing ever is in Michael Jackson's *Black or White* video:

<https://www.youtube.com/watch?v=F2AitPI5U0>

The morphing starts at around 05:30.



mp - November 22, 2022