Displacement Textures

Oregon State University

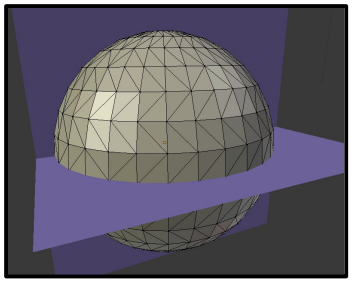
Mike Bailey

mjb@cs.oregonstate.edu

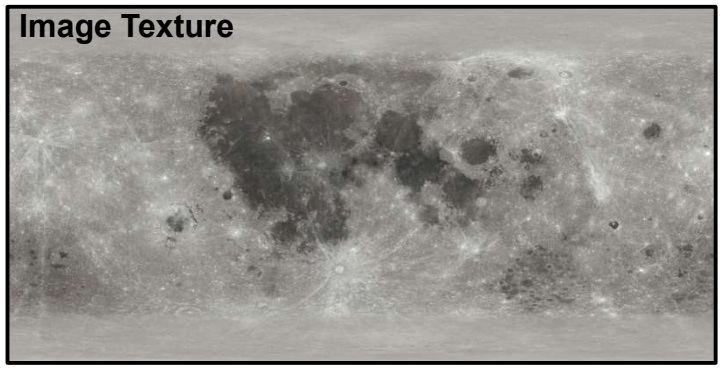
Oregon State University
Computer Graphics

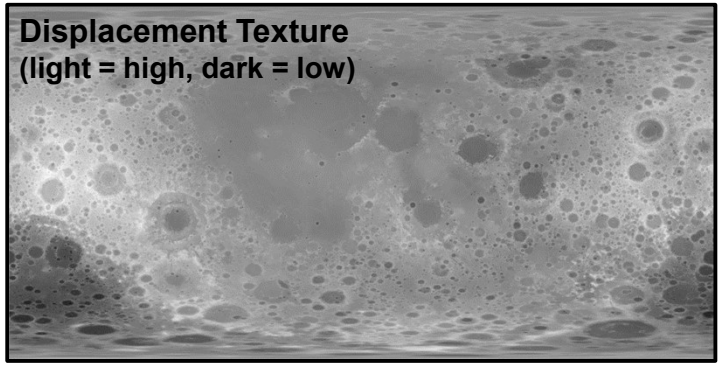
Displacement Textures are a Special Way to Model 3D Geometry

Vertex-described Object



+



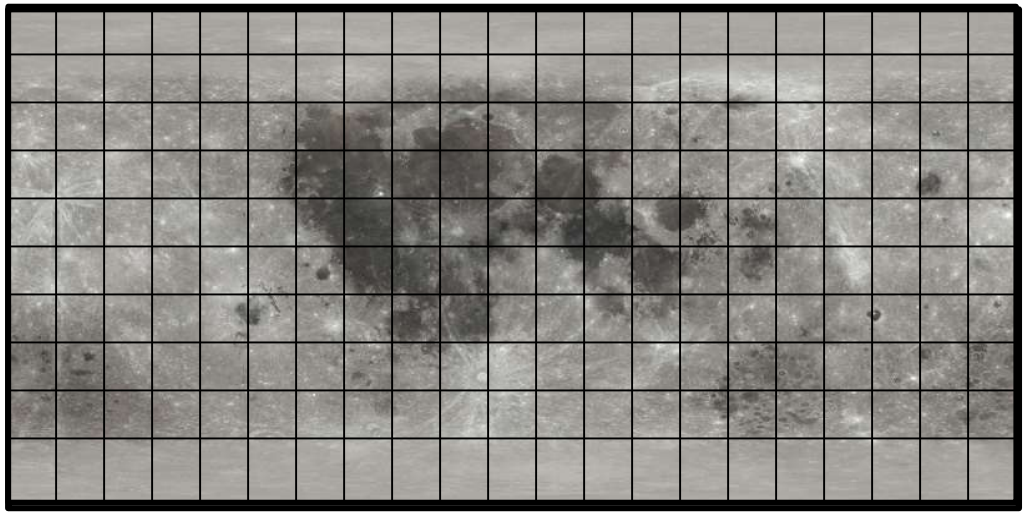


=

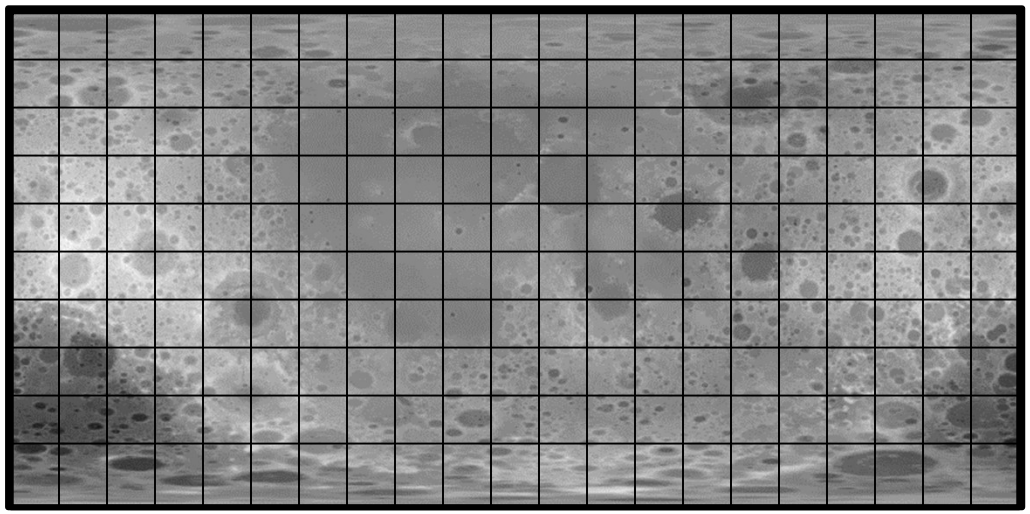
Oregon State University
Computer Graphics

Think of the Textures as *2D Data Tables*: One to Hold RGB Values and One to Hold Height Values

Image Texture

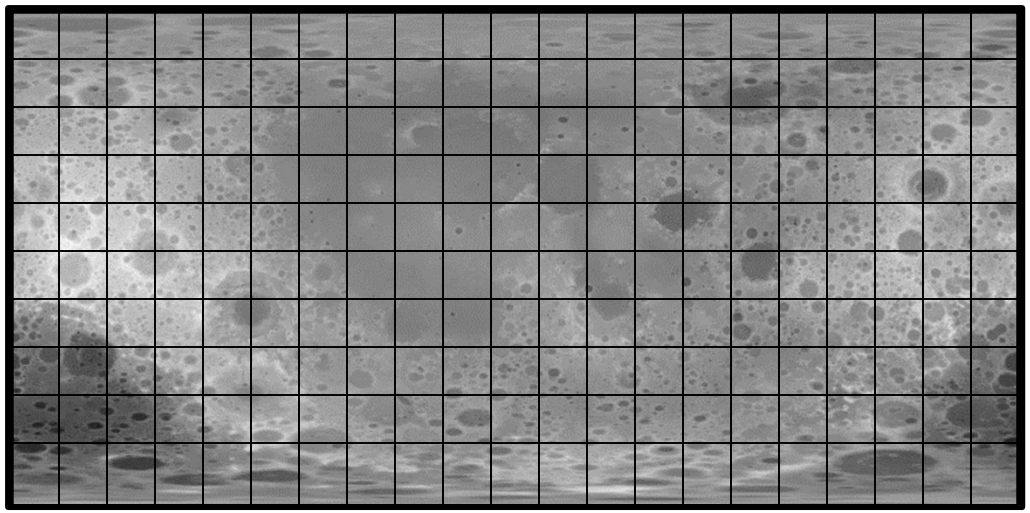


Displacement Texture
(light = high, dark = low)



Why Use a Texture to Hold Height Values?

Displacement Texture
(light = high, dark = low)



Textures:

1. Can be very large.
2. Have been architected for fast lookup.
3. Are capable of bilinear interpolation if you lookup an (s,t) that doesn't fall exactly on a height value. Therefore, you have lots of flexibility with how dense your vertex mesh is.

Displacement Textures are a Special Way to Model 3D Geometry

moondisp.glib

```
##OpenGL GLIB
Perspective 70
LookAt 0 0 3   0 0 0   0 1 0

Texture2D 6 lroc_color_poles_4k.bmp
Texture2D 7 ldem_16_uint.bmp

Vertex      moondisp.vert
Fragment moondisp.frag
Program  MoonDisp                                   \
                        uDoElevations <false>       \
                        uDoLighting <false>         \
                        uLightX <-10. 0. 10.>        \
                        uLightY <-10. 0. 10.>        \
                        uLightZ <  1. 10. 10.>       \
                        uKa <0. 0.2 1.0>             \
                        uKd <0. 0.9  1.0>            \
                        uSeaLevel  <0. 0.25 1.0>     \
                        uHeightScale <0. 0.25 0.50>  \
                        uNormalScale <0. 0.04 0.10>  \
                        uColorUnit 6                  \
                        uDispUnit  7

Color 1. .3 0.
Sphere 1. 600 600
```

Displacement Textures are a Special Way to Model 3D Geometry

moondisp.vert

```
#version 330 compatibility

uniform float                   uLightX, uLightY, uLightZ;
uniform float                   uHeightScale;
uniform float                   uSeaLevel;
uniform sampler2D               uDispUnit;
uniform bool                    uDoElevations;

out vec2        vST;
out vec3        vN;     // normal vector
out vec3        vL;     // vector from point to light
vec3 LIGHTPOS = normalize( vec3( uLightX, uLightY, uLightZ ) );

void
main( )
{
        vec2 st = gl_MultiTexCoord0.st;
        vST = st;

        vec3 norm = normalize( gl_NormalMatrix * gl_Normal );          // normal vector
        vN = norm;
        vec4 ECposition = gl_ModelViewMatrix * gl_Vertex;       // eye coordinate position
        vL = LIGHTPOS - ECposition.xyz;                                 // vector from the point to the light position

        vec3 vert = gl_Vertex.xyz;
        if( uDoElevations )
        {
                float disp = texture( uDispUnit, st ).r;                // in half-meters, relative to a radius of 1727400 meters
                disp -= uSeaLevel;
                disp *= uHeightScale;
                vert += normalize(gl_Normal) * disp;
        }

        gl_Position = gl_ModelViewProjectionMatrix * vec4( vert, 1. );
}
```

Oregon State University
Computer Graphics

Displacement Textures are a Special Way to Model 3D Geometry

moondisp.frag, I

```
#version 330 compatibility

uniform bool            uDoLighting;
uniform float           uKa, uKd;
uniform float           uHeightScale;
uniform float           uNormalScale;
uniform sampler2D       uColorUnit;
uniform sampler2D       uDispUnit;
in vec2             vST;
in vec3             vN;
in vec3             vL;
const float DELTA = 0.01;

void main( )
{
        vec3 newColor = texture( uColorUnit, vST ).rgb;
        gl_FragColor = vec4( newColor, 1. );
        if( uDoLighting )
        {
                . . .
```

Oregon State University
Computer Graphics

Displacement Textures are a Special Way to Model 3D Geometry

moondisp.frag, II

```
        if( uDoLighting )
        {
                vec2 stp0 = vec2( DELTA, 0. );
                vec2 st0p = vec2( 0. , DELTA );
                float west  = texture2D( uDispUnit, vST-stp0 ).r;
                float east  = texture2D( uDispUnit, vST+stp0 ).r;
                float south = texture2D( uDispUnit, vST-st0p ).r;
                float north = texture2D( uDispUnit, vST+st0p ).r;
                vec3 stangent = vec3( 2.*DELTA, 0., uNormalScale * ( east - west ) );
                vec3 ttangent = vec3( 0., 2.*DELTA, uNormalScale * ( north - south ) );
                vec3 Normal = normalize( cross( stangent, ttangent ) );
                vec3 Light  = normalize(vL);

                vec3 ambient = uKa * newColor;
                float d = 0.;
                if( dot(Normal,Light) > 0. ) // only do diffuse if the light can see the point
                {
                        d = dot(Normal,Light);
                }
                vec3 diffuse = uKd * d * newColor;
                gl_FragColor = vec4( ambient+diffuse, 1. );
        }
}
```

Oregon State University
Computer Graphics

Displacement Textures are a Special Way to Model 3D Geometry

Just the image texture

Lighting only, no displacements

Displacements only, no lighting

Displacements + Lighting

Note: as you can imagine, static images do not do this justice. Being able to dynamically rotate the Moon and change the height exaggeration and light position makes a big difference!