What is Bump-Mapping?

Bump-mapping is the process of creating the illusion of 3D depth by using a manipulated surface normal in the lighting, rather than actually creating the extra surface detail.

Displacement-mapped

Bump-mapped

Definition of Height Fields – Think of the Pin Box!

The Most Straightforward Type of Bump-Mapping is Height Fields
void main() {
    // Vertex positions
    vMCposition = gl_Vertex .xyz;
    vECposition = (gl_ModelViewMatrix * gl_Vertex).xyz;

    // Compute light intensity
    float LightIntensity = dot(normalize(vec3(uLightX, uLightY, uLightZ) - vMCposition), normal);

    // Determine color
    if (LightIntensity < 0.1) {
        LightIntensity = 0.1;
    }
    if (uUseColor) {
        float here = texture2D(uHgtUnit, vST).r;
        vec3 color = BLUE;
        if (here > 0.) {
            float t = smoothstep(uLevel1-uTol, uLevel1+uTol, here);
            color = mix(GREEN, BROWN, t);
        }
        if (here > uLevel1+uTol) {
            float t = smoothstep(uLevel2-uTol, uLevel2+uTol, here);
            color = mix(BROWN, WHITE, t);
        }
        gl_FragColor = vec4(LightIntensity*color, 1.);
    } else {
        gl_FragColor = vec4(LightIntensity*uColor.rgb, 1.);
    }
}
Terrain Height Bump-mapping: Coloring by Height

No Exaggeration

Exaggerated

Terrain Height Bump-mapping: Even Zooming-in Looks Good

Crater Lake

Visualization by Nick Gebbie
The Second Most Straightforward Type of Bump-Mapping is Height Field Equations

Rock A Dropped  Rock B Dropped  Both Rocks Dropped

This is the coordinate system we will be using. The plane is X-Y with Z pointing up.

Bump-mapping to Create Polar Ripples

In 2D, a slope $m = \frac{dy}{dx}$. It can be expressed as the vector $[1, m]$.

The normal to the shape is the vector perpendicular to the vector slope:

$\begin{bmatrix} -m \\ 1 \end{bmatrix}$

Note that $[1, m] \cdot \begin{bmatrix} -m \\ 1 \end{bmatrix} = 0$, as it must be.

So, if $z = -Amp \cdot \cos(\frac{2\pi x}{Pd} - 2\pi Time)$, then the scalar slope $\frac{dz}{dx}$ is:

$\frac{dz}{dx} = Amp \cdot \frac{2\pi}{Pd} \cdot \sin(\frac{2\pi x}{Pd} - 2\pi Time)$,

and the vector slope is:

Tangent Vector (i.e., slope) = $\begin{bmatrix} 1. \\ 0. \\ Amp \cdot \frac{2\pi}{Pd} \cdot \sin(\frac{2\pi x}{Pd} - 2\pi Time) \end{bmatrix}$

Bump-mapping to Create Polar Ripples

Following the pattern from before, the normal vector is:

Normal Vector = $\begin{bmatrix} -Amp \cdot \frac{2\pi}{Pd} \cdot \sin(\frac{2\pi x}{Pd} - 2\pi Time) \\ 0. \\ 1. \end{bmatrix}$

This is true along just the X axis. The trick now is to rotate the normal vector into where we really are. Because we are just talking about a rotation, the transformation is the same as if we were rotating a vertex.

N' = Nx cos\Theta - Ny sin\Theta = Nx cos\Theta

Ny' = Nx sin\Theta + Ny cos\Theta = Nx sin\Theta

Nz' = Nz = 1.

In the final code, you would substitute R for x in the slope and normal equations.

(Also note that you could include some exponential decay to make this behave more like real ripples.)

Combining Bump and Cube Mapping

[Diagram of combined effects]