Using Noise to Automatically Generate Generic Terrain

Mike Bailey
mjb@cs.oregonstate.edu

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

The General Idea

Use noise to determine terrain heights. Utilize as many different parameters as we can to give a variety of terrain.

Reading a texture from within the vertex shader

Cross product to get a normal vector

It's always a heated discussion about how much quality lighting to put on terrain. We usually don't multiply by the normal matrix because you generally don't turn a landform around in your hands.

Cross product to get a normal vector

It's always a heated discussion about how much quality lighting to put on terrain. We usually don't multiply by the normal matrix because you generally don't turn a landform around in your hands.
void main( )
{
  vec3 Normal = vec3( 0., 0., 1. );
  vec3 color = BLUE;
  if( vMC.z > 0. )
  {
    float t = smoothstep( uLevel1-uTol, uLevel1+uTol, vMC.z);
    color = mix( GREEN, GRAY, t );
    Normal = normalize( vNs );
  }
  if( vMC.z > uLevel1+uTol )
  {
    float t = smoothstep( uLevel2-uTol, uLevel2+uTol, vMC.z);
    color = mix( GRAY, WHITE, t );
    Normal = normalize( vNs );
  }
  vec3 Light = normalize( vLs);
  vec3 Eye = normalize( vEs );
  vec3 ambient = uKa * color;
  float d = dot(Normal,Light);
  vec3 diffuse = uKd * d * color;
  float s = 0.;
  if( d > 0. ) // only do specular if the light can see the point
  {
    vec3 ref = normalize( 2. * Normal * dot(Normal,Light) - Light );
    s = pow( max( dot(Eye,ref),0. ), uShininess );
  }
  vec3 specular = uKs * s * uSpecularColor.rgb;
  gl_FragColor = vec4( ambient.rgb + diffuse.rgb + specular.rgb, 1. );
}

What does it mean to do specular lighting on terrain? No, I don’t know either, but here it is if you want it.