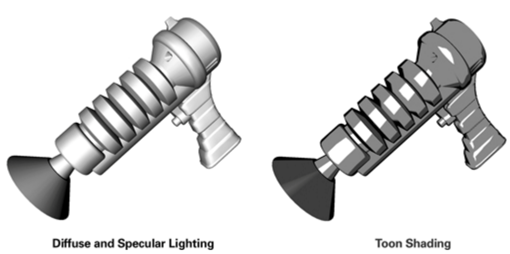
Cg Hacking

Toon Shading

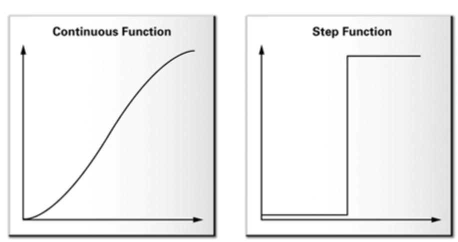


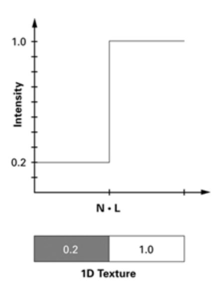
Toon Shading

Your toon shader has three main components:

1. The diffuse shading needs to be represented by just two values: one for bright regions, and another for dark regions.
2. Specular highlights need to be identified and represented as a single color where their intensity is sufficiently high.
3. Objects need to be outlined to complete the cartoon look.

Toon Shading 1. Diffuse Shading

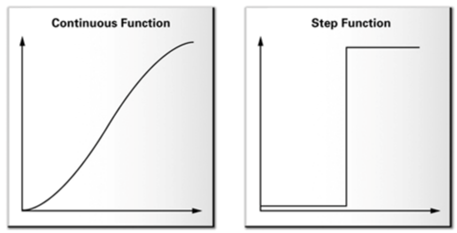


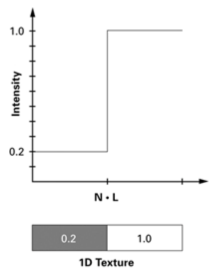


Vertex Prog:
diffuseLight = **max**(**dot**(N, L), 0);

Frag Prog:
diffuseLighting = **tex1D**(diffuseRamp, diffuseLight);

Toon Shading 2. Specular Highlights





Vert Prog
// Calculate specular lighting
float3 V = **normalize**(eyePosition - position.xyz);
float3 H = **normalize**(L + V);
specularLight = **pow**(**max**(**dot**(N, H), 0), shininess);

Frag Prof:
specularLighting = **tex1D**(diffuseRamp, specularLight);

Toon Shading 3. Silhouette Outlining

// Calculate edge color
float edge = **max**(**dot**(N, V), 0);
edge = **tex1D**(edgeRamp, edge);

Only works for curved surfaces.

Demo

Shadow Mapping

Shadow mapping is a two-pass technique:

1. **The scene is rendered from the light's point of view. The depth at each pixel of the resulting image is recorded in a "depth texture." (the *shadow map*.)**
2. **Next, the scene is rendered from the eye position, but with the shadow map projected down from the light onto the scene using standard projective texturing. At each pixel, the depth sample (from the projected shadow map texture) is compared with the fragment's distance from the light. If the latter is greater, the pixel is not the closest surface to the light source. This means that the fragment is shadowed, and that it should not receive light during shading.**
3. **The shadow map is indexed using (s/q, t/q). Because the light source is the center of projection for the shadow map, r/q holds the distance from the light. Therefore, by comparing the shadow map texel depth at (s/q, t/q) with r/q, you can determine if the current pixel is lit or in shadow.**

Shadow Mapping

tex2Dproj: returns a four-component vector
(*c, c, c,* 1), where
***c* is 0 if the pixel is in shadow**
1 if the pixel is lit.

You can then treat this vector as a color.

If bilinear texture filtering is enabled, *c* will range from 0 to 1 instead of being restricted to just the two values.

Shadow Mapping

Setup the texture matrix:

$$\begin{bmatrix} s \\ t \\ r \\ q \end{bmatrix} = \begin{bmatrix} \frac{1}{2} & & & \frac{1}{2} \\ & \frac{1}{2} & & \frac{1}{2} \\ & & \frac{1}{2} & \frac{1}{2} \\ & & & 1 \end{bmatrix} \begin{bmatrix} \textit{Light} \\ \textit{Frustum} \\ \textit{(projection)} \\ \textit{Matrix} \end{bmatrix} \begin{bmatrix} \textit{Light} \\ \textit{View} \\ \textit{(look at)} \\ \textit{Matrix} \end{bmatrix} \begin{bmatrix} \\ \textit{Modeling} \\ \textit{Matrix} \\ \\ \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ w_0 \end{bmatrix}$$

Vert Prog (use it)
float4 texCoordProj = **mul**(textureMatrix, position);

Shadow Mapping

Frag Prog:

tex2Dproj:
divides the *s* and *t* texture coordinates by the *q* texture coordinate

Use it:
float4 textureColor = **tex2Dproj**(projTexture, texCoordProj);

Demo