EDV & Multimedia

Game Development

Shading

Prof. Dr. Jochen Koubek
Ein (Pixel-) Shader ist ein kleines Programm, das auf der Grafikkarte ausgeführt wird und für jeden Bildschirmpunkt (Pixel) die auszugebende Farbe bestimmt.
Workflow

Da Shading die meiste Rechenzeit beansprucht, wird es erst am Ende der Produktion eingesetzt.

http://www.youtube.com/watch?v=Wj0XXYJ3gCY
Unity built-in Shader


Shader Models

All rendering in Unity is done with Shaders - small scripts that let you configure the how the graphics hardware is set up for rendering.

**Diffuse** computes a simple (Lambertian) lighting model. The lighting on the surface decreases as the angle between it and the light decreases. The lighting depends only on the this angle, and does not change as the camera moves or rotates around.

**Specular** computes the same simple (Lambertian) lighting as Diffuse, plus a viewer dependent specular highlight. This is called the Blinn-Phong lighting model. It has a specular highlight that is dependent on surface angle, light angle, and viewing angle. The highlight is actually just a realtime-suitable way to simulate blurred reflection of the light source. The level of blur for the highlight is controlled with the Shininess slider in the Inspector.

**Normal mapping** simulates small surface details using a texture, instead of spending more polygons to actually carve out details. It does not actually change the shape of the object, but uses a special texture called a Normal Map to achieve this effect. In the normal map, each pixel's color value represents the angle of the surface normal. Then by using this value instead of the one from geometry, lighting is computed. The normal map effectively overrides the mesh's geometry when calculating lighting of the object.

The **Parallax mapping** technique is pretty simple, so it can have artifacts and unusual effects. Specifically, very steep height transitions in the Height Map should be avoided. Adjusting the Height value in the Inspector can also cause the object to become distorted in an odd, unrealistic way. For this reason, it is recommended that you use gradual Height Map transitions or keep the Height slider toward the shallow end.
Material = Shader + Farben + Texturen
Beispiel: Reflective Parallax Specular

http://unity3d.com/support/documentation/Components/shader-ReflectiveParallaxSpecular.html
Textur

http://www.cgtextures.com/

Base-Textur (Albedo)

Normal-Bump Map
UV-Textur

Unity, 3rd Person Controller
UV-Texturing

Modell Entfalten und bemalen
Uncanny Valley

http://www.cubo.cc/
http://www.youtube.com/watch?v=PF1NQFmOoAE

Beowulf
Shading
Rendern
Surface Shader

The input structure Input generally has any texture coordinates needed by the shader. Texture coordinates must be named "uv" followed by texture name (or start it with "uv2" to use second texture coordinate set).

Standard output structure of surface shaders is this:

```c
struct SurfaceOutput {
    half3 Albedo;
    half3 Normal;
    half3 Emission;
    half Specular;
    half Gloss;
    half Alpha;
};
```

http://unity3d.com/support/documentation/Components/SL-SurfaceShaders.html
Surface Shader

http://unity3d.com/support/documentation/Components/SL-SurfaceShaderExamples.html
Surface Shader Lighting

http://unity3d.com/support/documentation/Components/SL-SurfaceShaderLightingExamples.html
Beispiel: Double Side Shader

Die Unity-Shader rendern von einer Fläche (Terrain, Plane, Cloth) nur eine Seite und schneiden die andere ab (Occlusion Culling). Wir benötigen daher einen Shader ohne Occlusion Culling

1. Download Build-In-Shader
   http://unity3d.com/support/resources/assets/built-in-shaders

2. Neues Material erzeugen

3. Shader kopieren: Specular, Normal-Glossy
   Rename to: Specular 2, Culling deaktivieren:

   ```
   SubShader {
       Tags {
           "RenderType"="Opaque"
       }
       LOD 200
       CULL OFF
   }
   ```
Occlusion Culling

http://unity3d.com/support/documentation/Manual/Occlusion%20Culling.html
Filled Polygons
Flat Shading / Lambertian Shading

Atari: I, Robot, 1983

Normal Simple Shader
Gouraud Shading
Vertex Lighting

LucasArts: TIE Fighter, 1994
Normal Vertex Lit
Texture Mapping

Parallax: Descent, 1995

Material
Pre-Computed Lighting
Lightmap

Id: Quake II, 1997

Lightmap
http://docs.unity3d.com/Documentation/Manual/Lightmapping.html
MIP Mapping

http://unity3d.com/support/documentation/Manual/Textures.html#mipmaps

http://aras-p.info/blog/2011/05/03/a-way-to-visualize-mip-levels/
Environment Mapping

Cube Map

http://unity3d.com/support/documentation/Components/class-Cubemap.html
Parametric Surface

id: Quake III, 1999

http://unity3d.com/support/documentation/Components/class-ProceduralMaterial.html
Shadow Mapping

Realtime Shadows

Ubisoft: Splinter Cell, 2002

http://unity3d.com/support/documentation/Manual/DirectionalShadowDetails.html
http://unity3d.com/support/documentation/Manual/Shadows.html
Post Processing

Killzone 2

Anti Aliasing

Tomb Raider ohne und mit 3DFX

http://unity3d.com/support/documentation/Components/script-AntialiasingAsPostEffect.html
http://unity3d.com/support/documentation/Components/class-QualitySettings.html
HDR Rendering

http://unity3d.com/support/documentation/Manual/HDR.html

Crytek: Far Cry, 2001
Ambient Occlusion

Vavle: Half Life 2, 2004

http://unity3d.com/support/documentation/Components/script-SSAOEffect.html
Relief Mapping

id: Doom 3

Level of Detail

Level of Details: Far Cry, 2004

http://unity3d.com/support/documentation/Manual/LevelOfDetail.html
Global Illumination

http://docs.unity3d.com/Documentation/Manual/LightmappingInDepth.html

Non Photorealistic Rendering


Okami, 2006

Assets > Import Package > toon shading