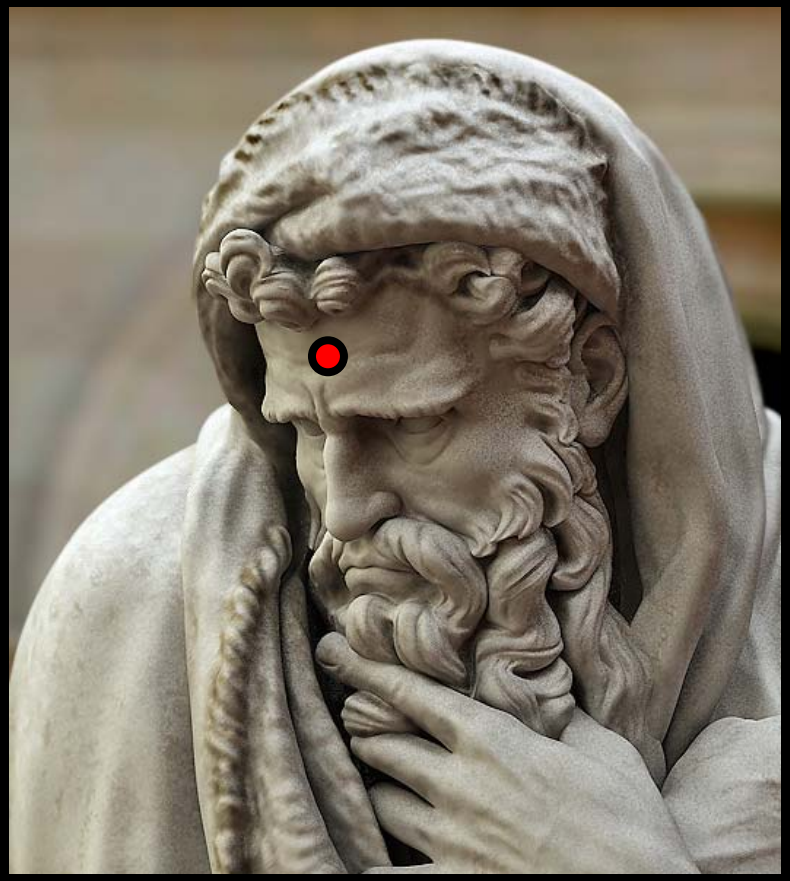
Programming graphics hardware

Gustav Taxén, Ph. D.
Associate Professor
School of Computing Science and Communication
Royal Institute of Technology, Stockholm
gustavt@csc.kth.se

Shading



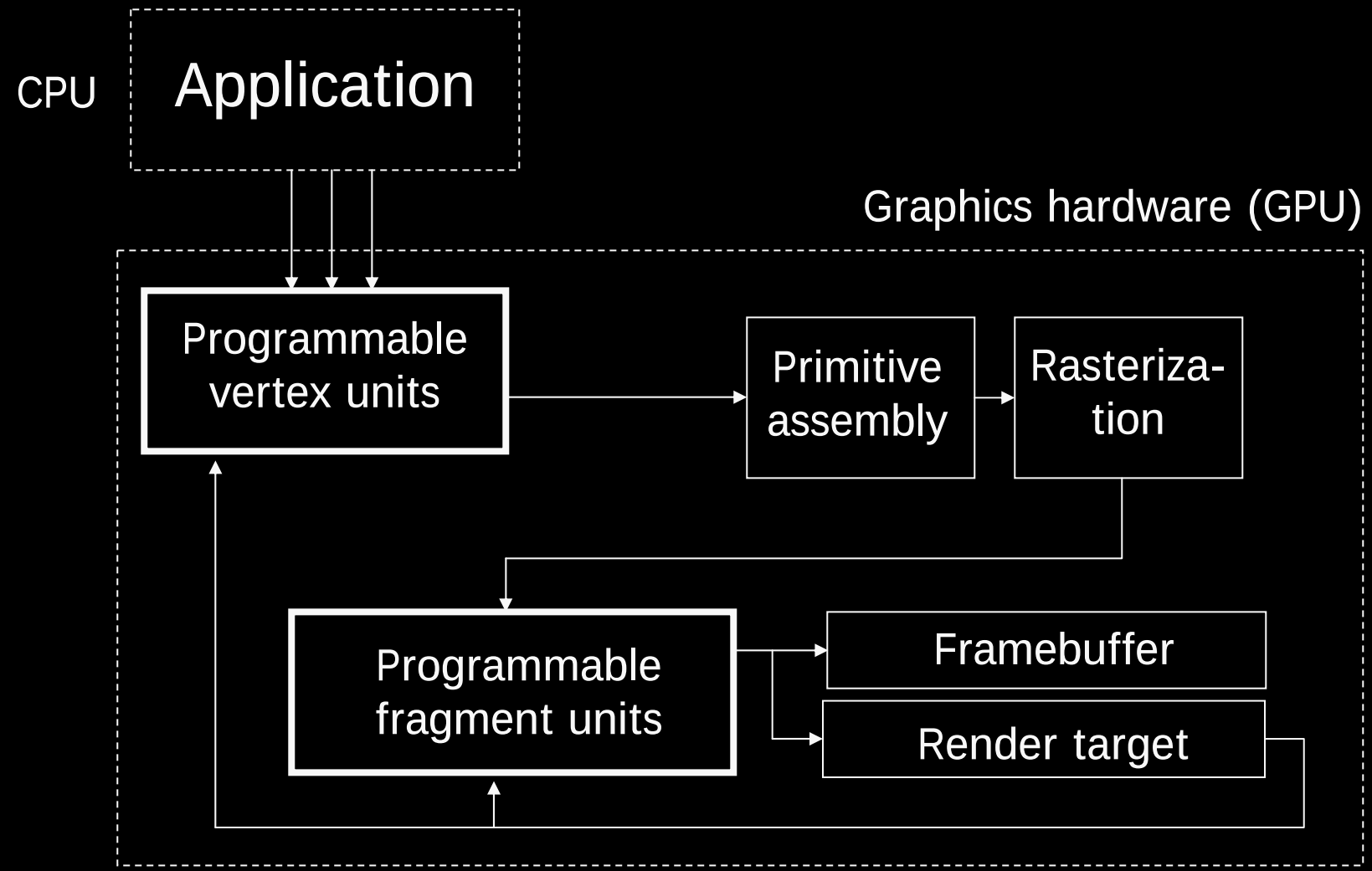
Vladimir Minguillo
http://forums.cgsociety.org/showthread.php?f=121&t=483818

Camera position
Surface position
Surface normal
Light position
Diffuse color layers
Specularity
Bump map
Environment map
...

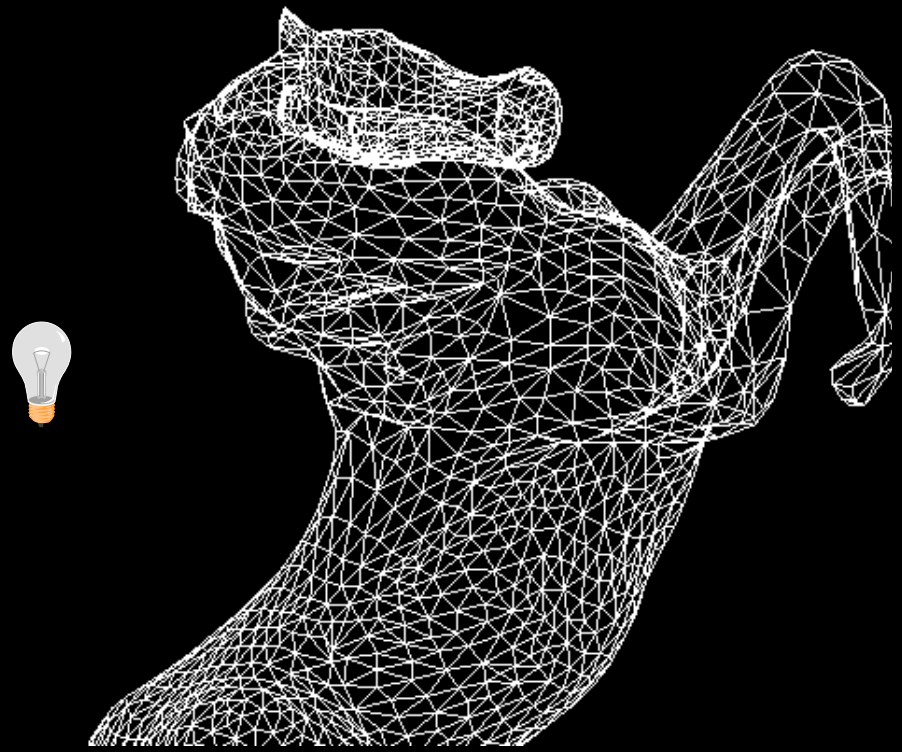
↓

Pixel color

The graphics pipeline

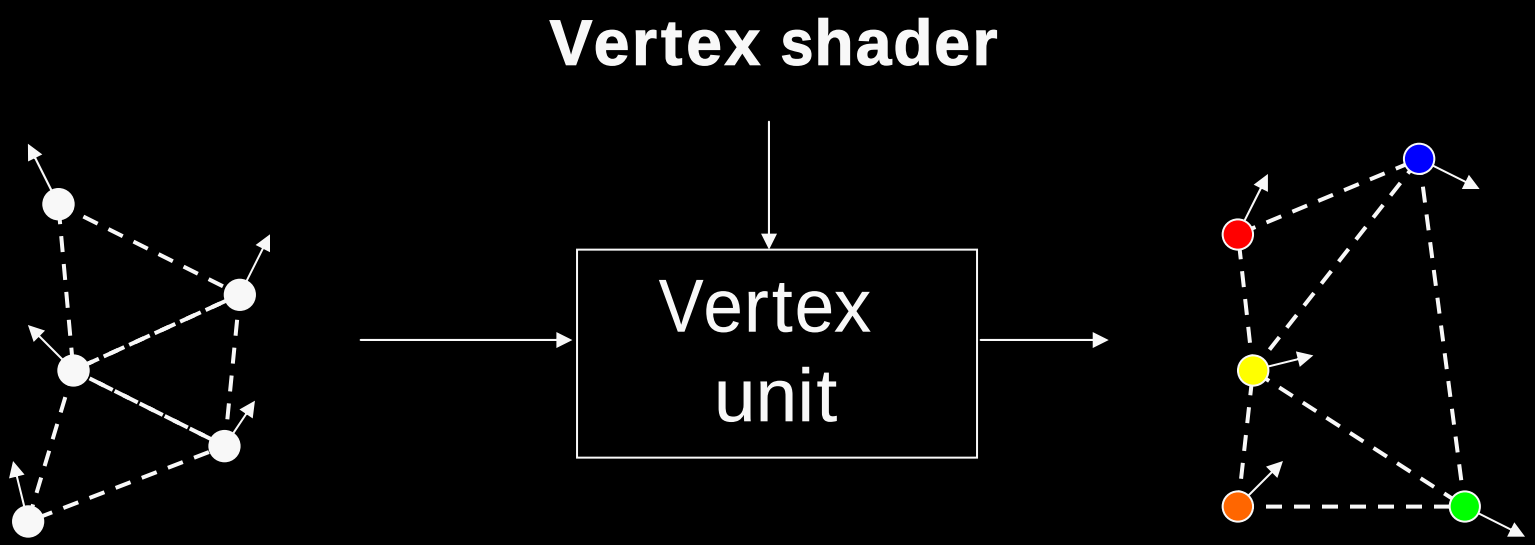


Uniform vs. varying data

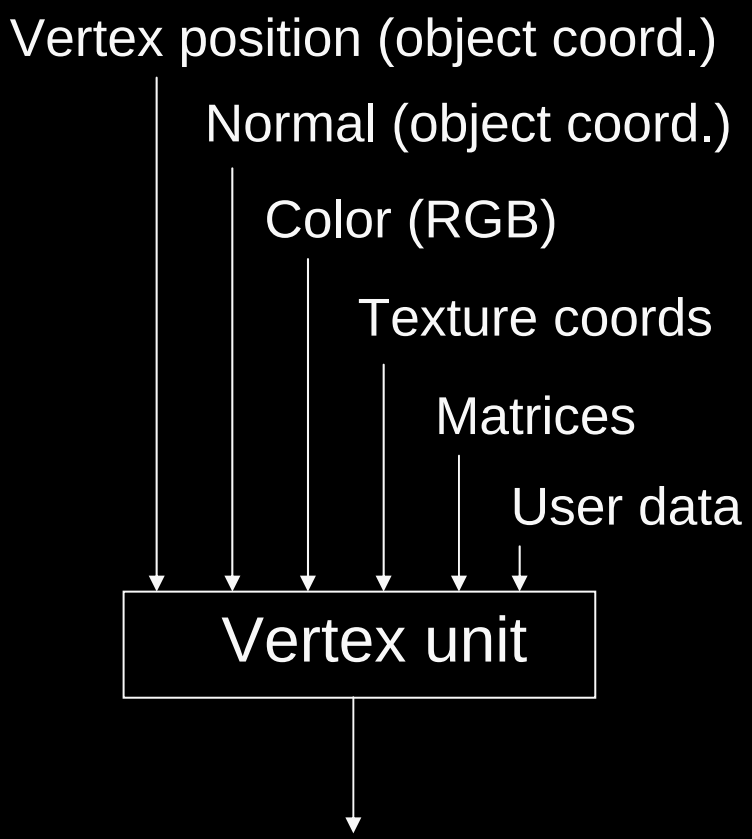


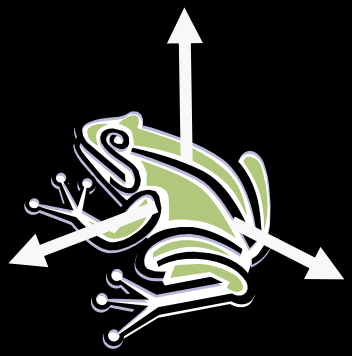
Varying data: data associated with vertices
Uniform data: everything else

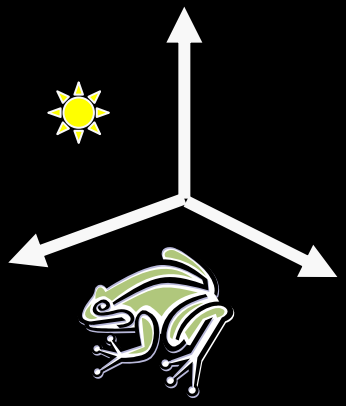
Vertex units



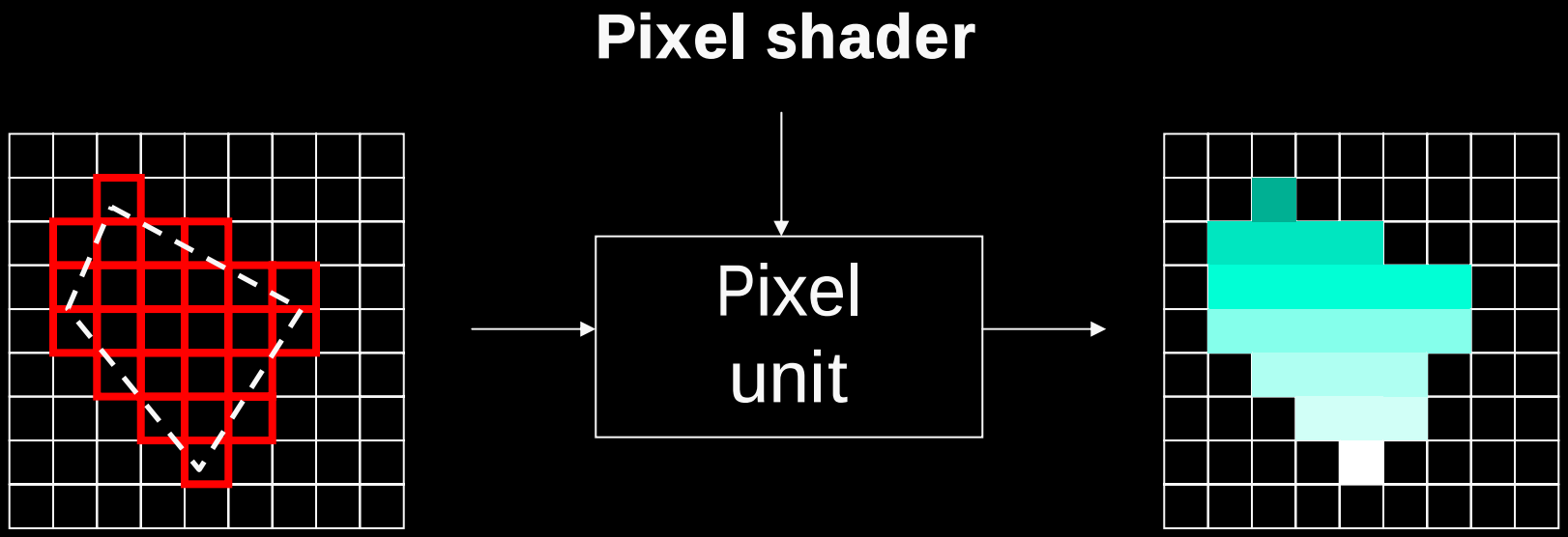
Vertex units





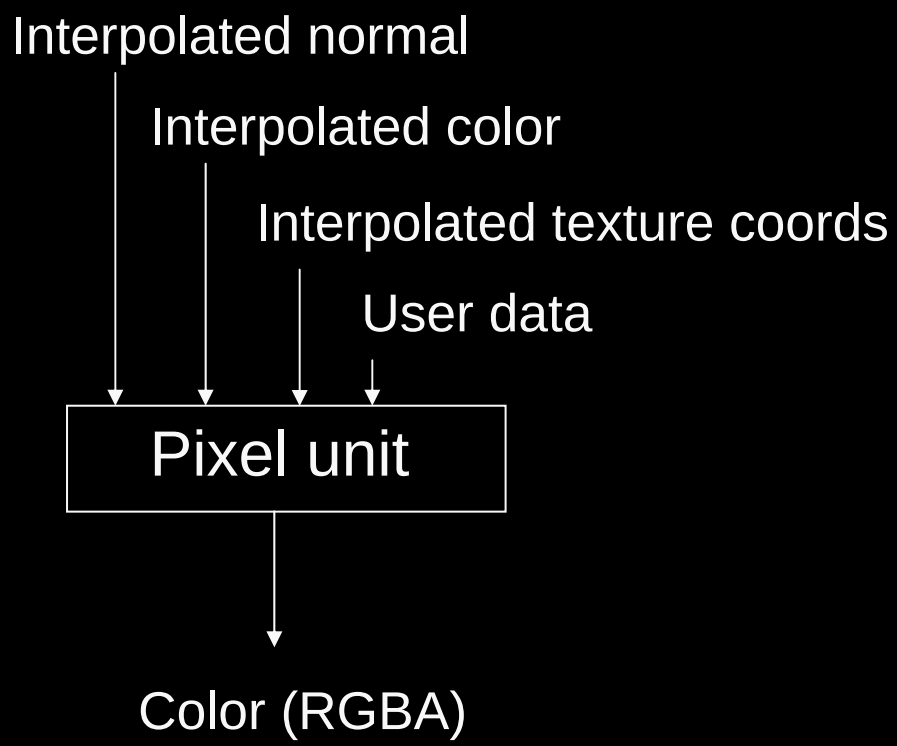


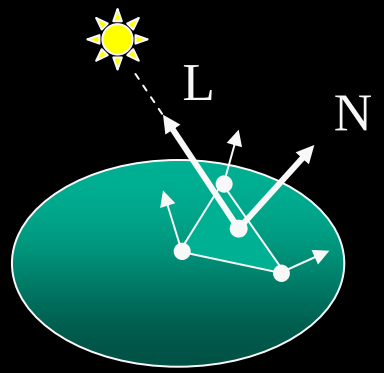
Fragment/pixel units



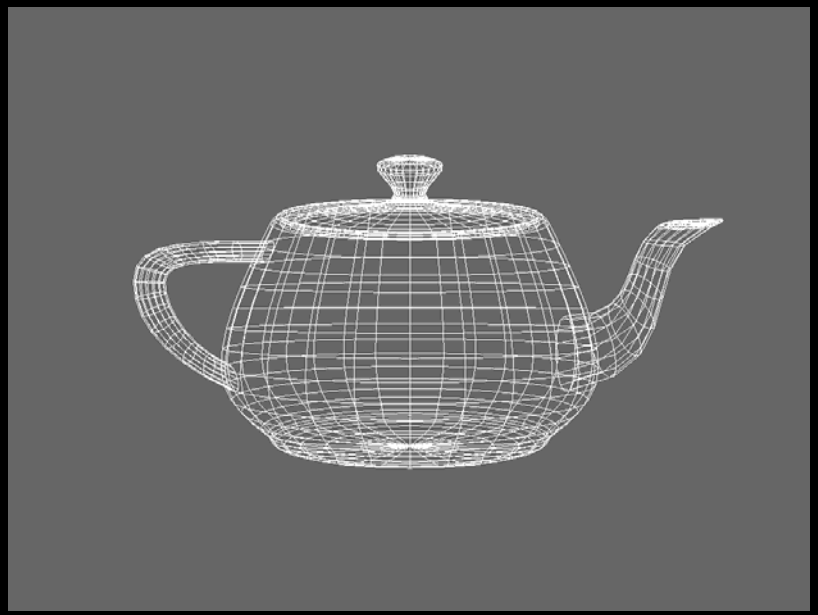
Varying data is interpolated linearly across the primitive

Fragment/pixel units





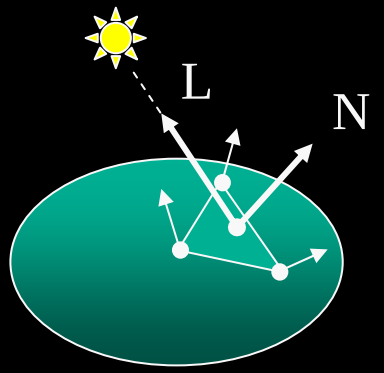
Example: Vertex shader



```
varying vec3 L;
varying vec3 N;

void main()
{
  vec3 L = vec3(gl_LightSource[0].position);
  vec3 N = gl_NormalMatrix * gl_Normal;
  gl_Position = ftransform();
}
```

Example: Fragment shader




```
varying vec3 N, L;

void main()
{
  N = normalize(N);
  L = normalize(L);
  float n_dot_l = dot(N, L);
  float c = clamp(n_dot_l, 0.0, 1.0);
  gl_FragColor = vec4(c, c, c, 1.0);
}
```

!!!

Shader languages

D3D Vertex Shader Assembler

```
dp3 r7.w, VECTOR_VERTEXTOLIGHT, VECTOR_VERTEXTOLIGHT
rsq VECTOR_VERTEXTOLIGHT.w, r7.w
dst r7, r7.wwww, VECTOR_VERTEXTOLIGHT.wwww
mul r6, r5, ATTENUATION.w
```

OpenGL Vertex Program Assembler

```
MOV   R1, R2.yzwx;
MUL   R1.xyz, R2, R3;
ADD   R1, R2, -R3;
RCP   R1, R2.w;
```

Shading languages

- Direct3D High-Level Shading Language (**HLSL**)
- nVidia **Cg**
- OpenGL Shading Language (**GLSL**)

- Pixar RenderMan
- Stanford Real-Time Shading Language
- ...

Hardware capabilities

- For HLSL, function set is defined by the DirectX version
- Cg specifies a number of **profiles** with different function sets
- GLSL programs use **`#extension`** tags to specify the capabilities they need

Shader language shoot-out

- GLSL
  - Pros: Simple to get started, integrates well with OpenGL
  - Cons: Semantics tied to specific syntax
- Cg
  - Pros: D3D/OpenGL cross-compability, used in games industry
  - Cons: Awkward syntax
- HLSL
  - Pros: Integrates well with D3D, used in games industry
  - Cons: Awkward syntax, Windows only

GLSL demos