```
Listing 4.1. Built-in uniform variables
//
// Matrix state
//
uniform mat4  gl_ModelViewMatrix;
uniform mat4  gl_ProjectionMatrix;
uniform mat4  gl_ModelViewProjectionMatrix;
uniform mat4  gl_TextureMatrix[gl_MaxTextureCoords];

//
// Derived matrix state that provides inverse and transposed versions
// of the matrices above. Poorly conditioned matrices may result
// in unpredictable values in their inverse forms.
//
uniform mat3  gl_NormalMatrix; // transpose of the inverse of the upper
                               // leftmost 3x3 of gl_ModelViewMatrix

uniform mat4  gl_ModelViewMatrixInverse;
uniform mat4  gl_ProjectionMatrixInverse;
uniform mat4  gl_ModelViewProjectionMatrixInverse;
uniform mat4  gl_TextureMatrixInverse[gl_MaxTextureCoords];

uniform mat4  gl_ModelViewMatrixTranspose;
uniform mat4  gl_ProjectionMatrixTranspose;
uniform mat4  gl_ModelViewProjectionMatrixTranspose;
uniform mat4  gl_TextureMatrixTranspose[gl_MaxTextureCoords]

uniform mat4  gl_ModelViewMatrixInverseTranspose;
uniform mat4  gl_ProjectionMatrixInverseTranspose;
uniform mat4  gl_ModelViewProjectionMatrixInverseTranspose;
uniform mat4  gl_TextureMatrixInverseTranspose[gl_MaxTextureCoords]

//
// Normal scaling
//
uniform float gl_NormalScale;

//
// Depth range in window coordinates
//
struct gl_DepthRangeParameters
{
    float near;        // n
    float far;         // f
    float diff;        // f - n
};
uniform gl_DepthRangeParameters gl_DepthRange;

//
// Clip planes
//
uniform vec4  gl_ClipPlane[gl_MaxClipPlanes];

//
// Point Size
//
struct gl_PointParameters
```

```
{
    float size;
    float sizeMin;
    float sizeMax;
    float fadeThresholdSize;
    float distanceConstantAttenuation;
    float distanceLinearAttenuation;
    float distanceQuadraticAttenuation;
};

uniform gl_PointParameters gl_Point;

//
// Material State
//
struct gl_MaterialParameters
{
    vec4 emission;     // Ecm
    vec4 ambient;      // Acm
    vec4 diffuse;      // Dcm
    vec4 specular;     // Scm
    float shininess;   // Srm
};
uniform gl_MaterialParameters  gl_FrontMaterial;
uniform gl_MaterialParameters  gl_BackMaterial;
//
// Light State
//

struct gl_LightSourceParameters
{
    vec4  ambient;              // Acli
    vec4  diffuse;              // Dcli
    vec4  specular;             // Scli
    vec4  position;             // Ppli
    vec4  halfVector;           // Derived: Hi
    vec3  spotDirection;        // Sdli
    float spotExponent;         // Srli
    float spotCutoff;           // Crli
                                // (range: [0.0,90.0], 180.0)
    float spotCosCutoff;        // Derived: cos(Crli)
                                // (range: [1.0,0.0],-1.0)
    float constantAttenuation;  // K0
    float linearAttenuation;    // K1
    float quadraticAttenuation; // K2
};

uniform gl_LightSourceParameters   gl_LightSource[gl_MaxLights];

struct gl_LightModelParameters
{
    vec4 ambient;        // Acs
};

uniform gl_LightModelParameters gl_LightModel;

//
```

```
// Derived state from products of light and material.
//

struct gl_LightModelProducts
{
    vec4 sceneColor;      // Derived. Ecm + Acm * Acs
};

uniform gl_LightModelProducts gl_FrontLightModelProduct;
uniform gl_LightModelProducts gl_BackLightModelProduct;

struct gl_LightProducts
{
    vec4 ambient;         // Acm * Acli
    vec4 diffuse;         // Dcm * Dcli
    vec4 specular;        // Scm * Scli
};
uniform gl_LightProducts gl_FrontLightProduct[gl_MaxLights];
uniform gl_LightProducts gl_BackLightProduct[gl_MaxLights];

//
// Texture Environment and Generation
//
uniform vec4  gl_TextureEnvColor[gl_MaxTextureUnits];
uniform vec4  gl_EyePlaneS[gl_MaxTextureCoords];
uniform vec4  gl_EyePlaneT[gl_MaxTextureCoords];
uniform vec4  gl_EyePlaneR[gl_MaxTextureCoords];
uniform vec4  gl_EyePlaneQ[gl_MaxTextureCoords];
uniform vec4  gl_ObjectPlaneS[gl_MaxTextureCoords];
uniform vec4  gl_ObjectPlaneT[gl_MaxTextureCoords];
uniform vec4  gl_ObjectPlaneR[gl_MaxTextureCoords];
uniform vec4  gl_ObjectPlaneQ[gl_MaxTextureCoords];

//
// Fog
//
struct gl_FogParameters
{
    vec4 color;
    float density;
    float start;
    float end;
    float scale;   // 1.0 / (gl_Fog.end - gl_Fog.start)
};

uniform gl_FogParameters gl_Fog;
```