



Ein *OpenGL*-Template mit Lichtquellen und Schattierung

Manfred Brill

Oktober 2005

1 Beleuchtung und Schattierung in OpenGL

Sie haben im Praktikum bereits mit dieser Technik gearbeitet. Um Ihnen die Arbeit damit etwas zu erleichtern gibt es eine weitere Vorlage, in der bereits einige Dinge vorbereitet sind. Diese Vorlage wird in einigen weiteren *OpenGL*-Demos verwendet.

OpenGL unterstützt wie Sie wissen *Flat*- und *Gouraud*-Shading. Mit Hilfe der globalen Variable `FLAT` und der Taste `F6` kann zwischen diesen beiden Verfahren umgeschaltet werden.

In der `init`-Funktion wird ersteinmal die Beleuchtung aktiviert und dann, je nach Einstellung, ein Schattierungsverfahren eingestellt:

```
/* Beleuchtung aktivieren          */
glEnable(GL_LIGHTING);
/* Shading aktivieren            */
if (FLAT)
    glShadeModel(GL_FLAT);
else
    glShadeModel(GL_SMOOTH);
```

In der Szene sind zwei Lichtquellen diagonal gegenüber bereits definiert:

```
/* Ambiente Lichtfarbe setzen      */
GLfloat ambientLight[] = {0.1f, 0.1f, 0.1f, 1.0f};
/* Diffuse Lichtfarbe setzen      */
GLfloat diffuseLight[] = {0.7f, 0.7f, 0.7f, 1.0f};
/* Spekulare Lichtfarbe setzen    */
GLfloat specularLight[] = {1.0f, 1.0f, 1.0f, 1.0f};
/* Highlight-Farbe setzen        */
GLfloat specular[] = {1.0f, 1.0f, 1.0f, 1.0f};

// Zwei Lichtquellen, diagonal gegenüber
glLightfv(GL_LIGHT0, GL_AMBIENT, ambientLight);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuseLight);
glLightfv(GL_LIGHT0, GL_SPECULAR, specularLight);
glEnable(GL_LIGHT0);
glLightfv(GL_LIGHT1, GL_AMBIENT, ambientLight);
glLightfv(GL_LIGHT1, GL_DIFFUSE, diffuseLight);
glLightfv(GL_LIGHT1, GL_SPECULAR, specularLight);
glEnable(GL_LIGHT1);
```

In *OpenGL* gibt es zwei Methoden, um Materialeigenschaften zu definieren. Eine kennen Sie aus dem Praktikum, man verwendet die Funktion `glMaterial*` wie in

```
GLfloat gray[] = {0.75, 0.75, 0.75, 1};
glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, gray);
```

Daneben gibt es das *Color Tracking*. Ist dies aktiviert, können Sie *OpenGL* mitteilen, dass für die Materialeigenschaften mit Hilfe von `glColor` übergeben werden.

```
/* Color Tracking anschalten      */
glEnable(GL_COLOR_MATERIAL);
glColorMaterial(GL_FRONT_AND_BACK, GL_AMBIENT_AND_DIFFUSE);
```

Da als Highlight-Farbe sehr häufig weiß verwendet wird wird auch dies in der `init`-Funktion gesetzt:

```
// Highlight-Farbe setzen für alle Objekte
// Der Exponent wird bei den Objekten individuell gesetzt!
glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, specular);
```

Bei den einzelnen Objekten in `display` müssen Sie dann nur noch die Farbe mit `glColor` und den Phong-Exponenten setzen:

```
glPushMatrix();
    glTranslatef(-6.0, 1.75, 0.0);
    glColor3f(0.1, 0.9, 0.05);
    glMateriali(GL_FRONT_AND_BACK, GL_SHININESS, 60);
    glutSolidDodecahedron();
glPopMatrix();
```

Ebenfalls in `display` werden die Lichtquellen positioniert.

```
/* Lichtquellen positionieren */
GLfloat light0Pos[] = {-5.0f, 5.0f, 10.0f, 1.0f};
    glLightfv(GL_LIGHT0, GL_POSITION, light0Pos);
GLfloat light1Pos[] = {5.0f, 5.0f, -10.0f, 1.0f};
    glLightfv(GL_LIGHT1, GL_POSITION, light1Pos);
```

Achten Sie darauf, ob Sie ein Headlight haben möchten oder nicht - Sie erinnern sich, das bestimmen Sie durch die Position der Funktion `glLightfv` zu `examine`.

Das Gitter in der Funktion `grid` wird nach wie vor ausgegeben; darüberhinaus wird das Rechteck in einer grauen Farbe ausgegeben.

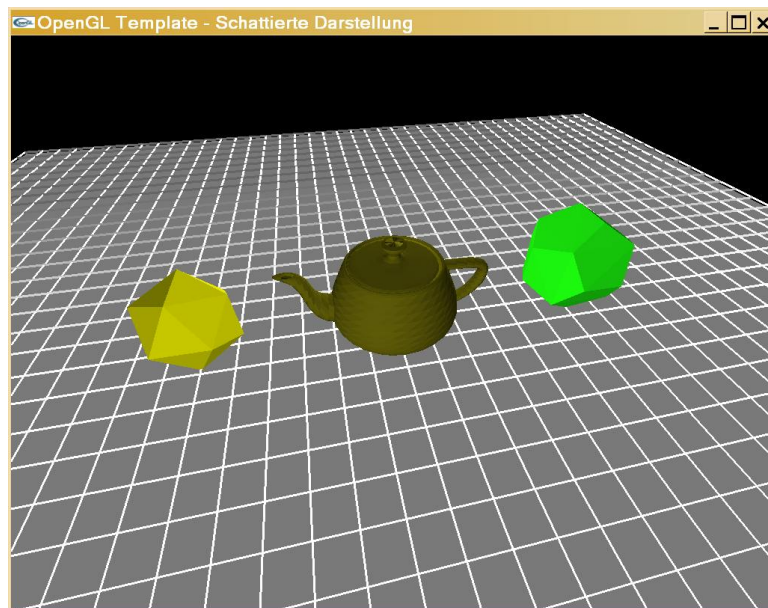


Abbildung 1: Das schattierte OpenGL-Template mit Flat-Shading

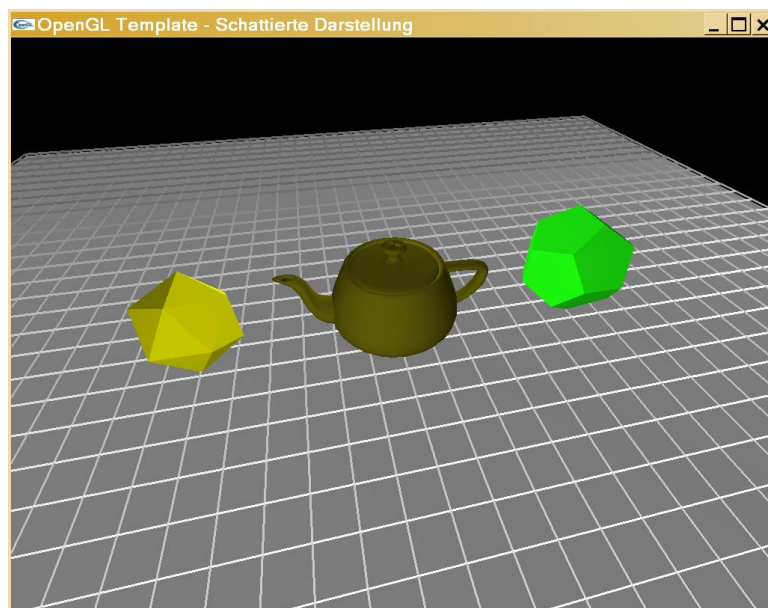


Abbildung 2: Das schattierte OpenGL-Template mit Gouraud-Shading