

# Course 3DC7, C2: Texture, Bauanleitung mit C++/MFC7.0

Copyright © by V. Miszalok, last update: 05-10-2002

- ↓ [Projekt texture1 mit leerem Fenster](#)
  - ↓ [Code für ein beleuchtetes OpenGL-Viereck](#)
  - ↓ [Code zum Einlesen einer Textur](#)
  - ↓ [Code für einen Oktaeder](#)
  - ↓ [Experimente](#)
  - ↓ [Beispielbilder](#)
- 

## Projekt texture1 mit leerem Fenster

Microsoft Visual Studio.NET starten

File - New - Project - Project Types: Visual C++ Projects, Templates: MFC Application

Name: texture1

Location: C:\temp

Button OK unten Mitte klicken

Links unter Overview auf Application Type klicken.

Schritt1: **single document einschalten**

Schritt 2: **Document/View architecture support** Checkbox **einschalten**

Schritt 3: **Finish**

Schritt 4: Klicken Sie im Hauptmenü von VS.NET auf den Menüpunkt **View** und dann auf den Unterpunkt **Class View** **Ctrl+Shift+C**.

Sie sehen im rechten Bereich des VS.NET-Hauptfensters das Unterfenster **Class View - texture1**.

An dessen unteren Rand die Registerkarte **Klassen** klicken.

Jetzt sehen Sie oben die Zeile **+ texture1**, klicken Sie mit der rechten Maustaste auf **texture1**.

Es öffnet sich ein Kontextmenü. Klicken Sie ganz unten auf **Properties**. Es öffnet sich ein Fenster **texture1 Property Pages**.

Auf der linken Seite der **texture1 Property Pages** öffnen Sie den Eintrag **Linker**. Klicken Sie auf den Untereintrag **Input**.

Auf der rechten Seite der **texture1 Property Pages** erscheint eine Tabelle. In deren erster Zeile steht **Additional Dependencies**.

Dahinter in das freie Feld in der zweiten Spalte schreiben Sie **opengl32.lib glu32.lib** und verlassen die **texture1 Property Pages** mit **OK**.

Schritt 5: Probeweise ausführen, beenden.

## Code für ein beleuchtetes OpenGL-Viereck

Klicken Sie im Hauptmenu von VS.NET auf den Menüpunkt `View` und dann auf den Unterpunkt `Class View` `Ctrl+Shift+C`.

Sie sehen im rechten Bereich des VS.NET-Hauptfensters das Unterfenster `Class View - texture1`.

An dessen unteren Rand die Registerkarte `Klassen` klicken.

Sie sehen im Unterfenster `Class View - texture1` die Zeile `+ texture1`, klicken Sie das Pluszeichen.

Sie sehen u.a. `+ CMainFrame`, klicken Sie das Pluszeichen.

Sie sehen u.a. `PreCreateWindow(CREATESTRUCT& cs)`, doppelklicken Sie diese Funktion.

Sie editieren jetzt das File `MainFrm.cpp`, Schreiben Sie dort in die Funktion

`PreCreateWindow(CREATESTRUCT& cs)` anstelle der beiden Kommentarzeilen:

```
cs.cx = 700;
cs.cy = 700;
```

Doppelklicken Sie im Unterfenster `Class View - texture1` die Zeile `Ctexture1View`.

Sie editieren jetzt das File `texture1View.h`.

Deklariieren Sie am Anfang (am besten gleich unter der ersten geschweiften Klammer) von `class CTexture1View : public CView` eine Variable:

```
private:
    HGLRC hglrc;
```

Doppelklicken Sie im Unterfenster `Class View - texture1` im Teilbaum von `Ctexture1View` den Konstruktor `Ctexture1View(void)`.

Sie editieren jetzt das File `texture1View.cpp`.

Schreiben Sie oben in den Kopf des Files unterhalb der Zeile `#endif`, jedenfalls noch vor dem Konstruktor folgende Zeilen:

```
#include < gl/gl.h >
float f[4][3] = { { -0.5f, -0.5f, 0.f },
                 {  0.5f, -0.5f, 0.f },
                 {  0.5f,  0.5f, 0.f },
                 { -0.5f,  0.5f, 0.f } };

int   face[4] = { 0, 1, 2, 3 };
float n   [3] = { 0.f, 0.f, 1.f };
float diffuseLight [4] = { 1.5f, 1.5f, 1.5f, 1.0f };
float materialDiffuse[4] = { 1.0f, 1.0f, 1.0f, 0.0f };
float LightPosition [4] = { 1.0f, 0.0f, 1.0f, 1.0f };
```

Klicken Sie im Klassenbaum mit der rechten Maustaste auf `Ctexture1View`.

Es öffnet sich ein Kontextmenü. Klicken Sie ganz unten im Kontextmenü auf `Properties`.

Es öffnet sich unter dem `Class View - texture1 - Fenster` ein `Properties - Fenster` mit der Überschrift `Ctexture1View VCCodesClass`.

Im `Toolbar` dieses Fensters klicken Sie rechts neben dem gelben Blitz auf das `Messages - Symbol`, das aussieht wie ein weißer Topf mit blauem Deckel.

Sie sehen nun die Liste der `Windows-Messages` von `Ctexture1View`.

Klicken Sie auf `WM_CREATE` und dann auf das schwarze Abwärtsdreieck rechts in dieser Zeile und dann auf `Add OnCreate`.

Klicken Sie auf `WM_TIMER` und dann auf das schwarze Abwärtsdreieck rechts in dieser Zeile und dann auf `Add OnTimer`.

Programmieren Sie die beiden eben generierten Funktionen bis sie so aussehen:

```
int Ctexture1View::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CView::OnCreate(lpCreateStruct) == -1) return -1;
    CClientDC dc( this );
    PIXELFORMATDESCRIPTOR pfd = {
        sizeof( PIXELFORMATDESCRIPTOR ),
        1,
        PFD_DRAW_TO_WINDOW | PFD_SUPPORT_OPENGL | PFD_DOUBLEBUFFER,
        PFD_TYPE_RGBA,
        24,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,32,0,0,
        PFD_MAIN_PLANE,0,0,0 };
    int i = ChoosePixelFormat( dc.m_hDC, &pfd );
    SetPixelFormat( dc.m_hDC, i, &pfd );
    hglrc = wglCreateContext( dc.m_hDC );
    SetTimer(1, 1, NULL);
    return 0;
}

void Ctexture1View::OnTimer(UINT nIDEvent)
{
    CClientDC dc( this );
    wglMakeCurrent( dc.m_hDC, hglrc );
    glDrawBuffer( GL_BACK );
    glClearColor( 0.1f, 0.1f, 0.1f, 0.f );
    glClear( GL_COLOR_BUFFER_BIT );
    glMatrixMode( GL_MODELVIEW );
    glRotatef(1.f, 1.f, 1.f, 1.f );
    glPushMatrix();
    glLoadIdentity();
    glLightfv( GL_LIGHT0, GL_POSITION, LightPosition );
    glPopMatrix();

    glLightfv( GL_LIGHT0, GL_DIFFUSE, diffuseLight );
    glMaterialfv( GL_FRONT, GL_DIFFUSE, materialDiffuse );

    glEnable(GL_LIGHTING);
    glEnable(GL_LIGHT0);

    glBegin(GL_POLYGON);
        glNormal3f( n[0], n[1], n[2] );
        glVertex3f( f[0][0], f[0][1], f[0][2] );
        glVertex3f( f[1][0], f[1][1], f[1][2] );
        glVertex3f( f[2][0], f[2][1], f[2][2] );
        glVertex3f( f[3][0], f[3][1], f[3][2] );
    glEnd();
    SwapBuffers(dc.m_hDC);
}
```

Ausführen, experimentieren Sie mit den jeweils ersten 3 Parametern von:

```
float diffuseLight [4] = { 1.5f, 1.5f, 1.5f, 1.0f };
```

```
float materialDiffuse[4] = { 1.0f, 1.0f, 1.0f, 0.0f };
```

```
float LightPosition [4] = { 1.0f, 0.0f, 1.0f, 1.0f };
```

## Code zum Einlesen einer Textur

Doppelklicken Sie im Unterfenster Class View - texture1 die Zeile Ctexture1Doc.

Sie editieren jetzt texture1Doc.h.

Schreiben Sie im Kopf von texture1Doc.h vor die Klasse CTexture1Doc aber unter #pragma once:

```
#include < vector >
#include < gl/gl.h >
#include < gl/glu.h >
```

Deklariieren Sie in texture1Doc.h in der Klasse CTexture1Doc (am besten gleich unter der ersten geschweiften Klammer) folgende Variablen:

```
public:
    BITMAPFILEHEADER    FH;
    BYTE                IBytes[1200]; //bytes for BitmapInfoHeader+Palette
    BITMAPINFOHEADER*   pIH;           //pointer on BitmapInfoHeader
    BITMAPINFO*         pI;           //pointer on BitmapInfo
    std::vector< BYTE > OriginalImage; //dyn. array for pixel
    BOOL                APictureHasBeenLoaded;
```

Klicken Sie im Unterfenster Class View - texture1 das Pluszeichen in der Zeile + Ctexture1Doc.

Doppelklicken Sie auf den Konstruktor Ctexture1Doc(void). Sie editieren jetzt texture1Doc.cpp.

Initialisieren Sie die folgenden Variablen im Konstruktor:

```
Ctexture1Doc::Ctexture1Doc()
{
    pIH = (BITMAPINFOHEADER*) IBytes;
    pI  = (BITMAPINFO*)      IBytes;
    APictureHasBeenLoaded = FALSE;
}
}
```

Fügen Sie wie gewohnt in der Klasse CTexture1Doc die Funktion void forget\_it() ein.

```
void Ctexture1Doc::forget_it()
{
    OriginalImage.clear();
    APictureHasBeenLoaded = FALSE;
    for ( int i=0; i < 10; i++ ) MessageBeep(-1);
}
}
```

Deklariieren Sie in texture1View.h in der Klasse CTexture1View : public CView direkt unterhalb von HGLRC hglrc; einen dynamischen Array:

```
std::vector< BYTE > ScaledImage; //width/height must be 32|64|128|256|512|1024
```

Schreiben Sie in texture1Doc.cpp in die Funktion void Ctexture1Doc::Serialize(CArchive& ar) unter else:

```
{
    ar.Read( &FH, sizeof(BITMAPFILEHEADER) );
    if ( FH.bfType != 'MB' ) { forget_it(); return; }
    if ( FH.bfSize  <= 54 ) { forget_it(); return; }
    if ( FH.bfOffBits < 54 ) { forget_it(); return; }
    int nBytesInfo = FH.bfOffBits - sizeof(BITMAPFILEHEADER);
    int nBytesPixel = FH.bfSize - FH.bfOffBits;
    ar.Read( IBytes, nBytesInfo ); //BitmapInfoHeader+Palette
    OriginalImage.resize( nBytesPixel );
    ar.Read( &OriginalImage.front(), nBytesPixel );
    APictureHasBeenLoaded = TRUE;
    if ( pIH->biBitCount == 24 ) return; //ok
    if ( pIH->biBitCount == 8 ) return; //ok
    forget_it(); //not ok
}
}
```

Ergänzen Sie folgende Zeilen in `texture1View.cpp` in der Funktion `void Ctexture1View::OnTimer(UINT nIDEvent)` nach der Zeile `glEnable(GL_LIGHT0);` und vor der Zeile `glBegin(GL_POLYGON);`:

```

Ctexture1Doc* pDoc = GetDocument();
int cx, cy, format;
if ( pDoc->APictureHasBeenLoaded )
{
    cx = cy = 128;
    if ( pDoc->pIH->biBitCount == 24 )
    {
        ScaledImage.resize( cx*cy*3 );
        format = GL_BGR_EXT;
    }
    else if ( pDoc->pIH->biBitCount == 8 )
    {
        ScaledImage.resize( cx*cy );
        format = GL_LUMINANCE;
    }
    gluScaleImage( format,
                  pDoc->pIH->biWidth,
                  pDoc->pIH->biHeight,
                  GL_UNSIGNED_BYTE, &(pDoc->OriginalImage.front()),
                  cx, cy, GL_UNSIGNED_BYTE, &ScaledImage.front() );
    glTexParameteri( GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST );
    glTexParameteri( GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST );
    glTexImage2D ( GL_TEXTURE_2D, 0, GL_RGB, cx, cy,
                  0, format, GL_UNSIGNED_BYTE,
&ScaledImage.front() );
    glBindTexture( GL_TEXTURE_2D, 1 );
    pDoc->OriginalImage.clear();
    pDoc->APictureHasBeenLoaded = FALSE;
}

glEnable      ( GL_TEXTURE_2D );
glBindTexture( GL_TEXTURE_2D, 1 );
glTexEnvf    ( GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE );

```

und schreiben Sie vor die vier `glVertex3f`-Befehle zwischen `glBegin(GL_POLYGON)` und `glEnd()` je einen `glTexCoord`-Befehl:

```

glTexCoord2f(0.0f, 0.0f); glVertex3f( f[0][0], f[0][1], f[0][2] );
glTexCoord2f(1.0f, 0.0f); glVertex3f( f[1][0], f[1][1], f[1][2] );
glTexCoord2f(1.0f, 1.0f); glVertex3f( f[2][0], f[2][1], f[2][2] );
glTexCoord2f(0.0f, 1.0f); glVertex3f( f[3][0], f[3][1], f[3][2] );

```

Wenn Sie einen langsamen Rechner haben, schreiben Sie hinter `glEnd()` noch die Zeile:

```
glFlush();
```

und als letzten Befehl der Timer-Funktion unter `SwapBuffers( dc.m_hDC );`:

```
dc.TextOut( 0, 0, "Please open a 8/24-Bit *.BMP file!" );
```

`texture1` ist vorläufig fertig, ausführen, 8- oder 24 Bit \*.bmp Datei öffnen, beenden.

## Code für einen Oktaeder

Ersetzen Sie im Kopf von `texture1View.cpp` die Definitionen der Arrays `f[4][3]`, `face[4]` und `n[3]` mitsamt deren Initialisierungen durch:

```
float f[6][3] = { { 0.0f, 0.0f, 1.0f },
                 { 0.7f, 0.7f, 0.0f },
                 { -0.7f, 0.7f, 0.0f },
                 { -0.7f, -0.7f, 0.0f },
                 { 0.7f, -0.7f, 0.0f },
                 { 0.0f, 0.0f, -1.0f } };

int face[8][3] = { { 0, 4, 1 },
                  { 0, 1, 2 },
                  { 0, 2, 3 },
                  { 0, 3, 4 },
                  { 5, 1, 4 },
                  { 5, 2, 1 },
                  { 5, 3, 2 },
                  { 5, 4, 3 } };

#define NX 0.8192f
#define NY 0.8192f
#define NZ 0.5735f
float n[8][3] = { { NX, 0.f, NZ },
                 { 0.f, NY, NZ },
                 { -NX, 0.f, NZ },
                 { 0.f, -NY, NZ },
                 { NX, 0.f, -NZ },
                 { 0.f, NY, -NZ },
                 { -NX, 0.f, -NZ },
                 { 0.f, -NY, -NZ } };
```

Die drei folgenden Zeilen mit den Definitionen von `diffuseLight`, `materialDiffuse` und `LightPosition` müssen unverändert bleiben.

In der `Timer`-Funktion müssen Sie hinter `glEnable(GL_LIGHT0)` eine Zeile einfügen:

```
glEnable( GL_CULL_FACE );
```

und nunmehr 8 Dreiecke zeichnen. Vorschlag:

Ersetzen Sie die gesamte Klammer zwischen `glBegin(GL_POLYGON)` und `glEnd()` inclusive `glBegin(GL_POLYGON)` und `glEnd()` durch:

```
#define VERTEX glVertex3f( f[j][0], f[j][1], f[j][2] );
int i, j;
for ( i = 0; i < 8; i++ )
{ glBegin( GL_POLYGON );
  glNormal3f( n[i][0], n[i][1], n[i][2] );
  glTexCoord2f( 0.5f, 1.0f ); j=face[i][0]; VERTEX
  glTexCoord2f( 0.0f, 0.0f ); j=face[i][1]; VERTEX
  glTexCoord2f( 1.0f, 0.0f ); j=face[i][2]; VERTEX
  glEnd();
}
```

Die Endersion von `texture1` ist fertig, ausführen, 8- oder 24-Bit \*.bmp-Datei öffnen, weitere Texturen zur Laufzeit öffnen, beenden.

Wenn Ihnen die Animation zu langsam läuft, schalten Sie das Projekt um auf `release` - mode. Gehen Sie dazu im Hauptmenü von VS auf `Build -> Configuration Manager`. Es öffnet sich ein Fenster `Configuration Manager`. Sie setzen das Feld `Active Solution Configuration` auf `"Release"` und die Spalte `Configuration` auch auf `"Release"`. Verlassen mit `Close`.

Dann müssen Sie den Schritt 4 aus dem ersten Absatz dieser Bauanleitung wiederholen, um die beiden Bibliotheken `openg32.lib` und `glu32.lib` auch in das `release`-Projekt einzubinden.

Dann alles neu übersetzen, linken und ausführen mit `Debug -> Start Without Debugging`.

Erproben, beenden, löschen, neu programmieren.

## Experimente

- (1) Variieren Sie die Farbe des gerichteten diffusen Lichts.
- (2) Variieren Sie die Reflektionseigenschaften des Materials für diffuses Licht.
- (3) Variieren Sie Position der Quelle für diffuses Licht.
- (4) Erhöhen Sie die Ortsauflösung der Textur von 128\*128 auf 256\*256.
- (5) Variieren Sie die Spitze des Dreiecksausschnitts der Textur von `glTexCoord2f(0.5f, 1.0f)` auf `glTexCoord2f(0.9f, 1.0f)`.
- (6) Variieren Sie die Drehungsachsen.
- (7) Variieren Sie die Drehungsgeschwindigkeit.
- (8) Lassen Sie die Darstellung der ersten und der letzten Dreiecksfläche weg: `for ( i = 1; i < 7; i++ )`.
- (9) Variieren Sie die Beträge und Vorzeichen der 8 Normalenvektoren (Nur keinen in allen Richtungen auf Null setzen, sonst gibt es einen Crash.).
- (10) Setzen Sie je eine Markierungspunkt auf die 3 Austrittsstellen der x-, y- und z-Achse aus der Oktaeder.
- (11) Variieren Sie die Abmessungen des Oktaeders durch verschieben einzelner Vertices.
- (12) Verwenden Sie zusätzliches richtungsloses Umgebungslicht (ambient light) und sorgen Sie dafür, dass das Material solches Licht auch reflektiert.

## Beispielbilder

Wenn Sie keine 8-Bit \*.bmp - Dateien auf Ihrer Harddisk finden, benutzen Sie folgende Beispielbilder:

Download: [Madonna.bmp 18 kB 8Bit-Grauwert-Bild](#)

Download: [Lena256.bmp 66 kB 8Bit-Grauwert-Bild](#)

Download: [Lena512.bmp 258 kB 8Bit-Grauwert-Bild](#)

Download: [Angiography.bmp 66 kB 8Bit-Grauwert-Bild](#)

Download: [Butterfly.bmp 217 kB 24Bit-Echtfarb-Bild](#)