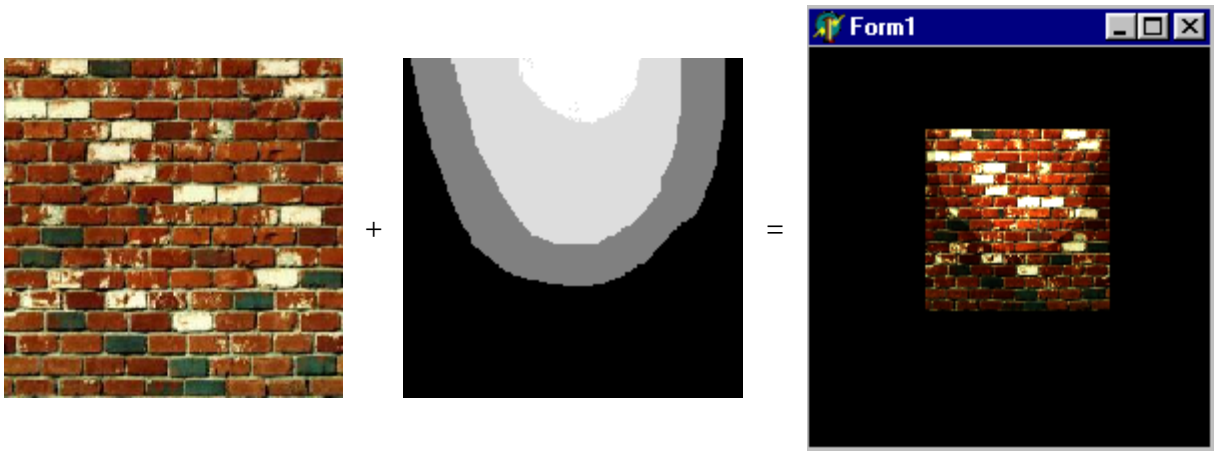
Open Gl - Delphi – Kurs Teil 7: Licht mit Blending

Einleitung

Hi! Wochen sind vergangen, das Teil 6 das Licht der Welt erblickte und nun erst kann ich euch mit dem nächsten Teil meines Kurses dienen. Das Problem lag auf der Hand und nannte sich Praktikum, wozu leider auch ein mehrseitiger Bericht gehört *hust*. Na ja ich hoffe die Wartezeit war nicht zu schlimm für euch...
Diesmal möchten wir uns also mit der (scheinbaren) Beleuchtung von Szenen befassen, welche wir mit Blending realisieren möchten. Aber wieso Blending? Gibt es keine echten Open Gl – Lichter? Doch, die gibt es schon, haben aber grobe Nachteile:
Zum einen sehen sie schlicht und einfach nicht sehr gut aus, zudem sind sie sehr rechenaufwendig (müssen ja in Echtzeit berechnet werden). Dazu kommt es, dass sie recht schwer zu handhaben sind... sie leuchten einfach quer durch alle Objekte der Szene hindurch, wenn man es ihnen nicht anders „beibringt“. Logischerweise ist somit die Schattenberechnung mit diesen Lichtern ein hoch mathematischer Akt, an welchen sich nur Läute mit mindestens 10 Punkte (= 2-) in einem Mathe Leistungskurs wagen sollten (zu denen ich nicht gehöre, da ich noch 11. Klasse bin).
Wie nennt sich also die Lösung dieses Dilemmas: Lichttexturen, die mit Blending auf die normalen Objekte gekleistert werden. Denkt nun nicht, dieses sei eine Minderwertige Lösung... wenn ich mich recht entsinne verwendeten viele 3d - Shooter der Vergangenheit dieses Prinzip (Beispiel: UT oder auch Half Life). Wie das bei neueren Sachen aussieht, kann ich leider nicht sagen, da diese Schlicht und einfach nicht auf meiner 500er – Gurke laufen.

Und Los!

Genug gefaselt, nun müssen wir auch mal was tun... na ja: die meiste Theorie kennt ihr ja schon, aber wie sieht so was in der Praxis aus? Ein Beispiel:



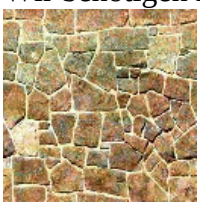
Sieht doch schon deutlich besser aus, als im Mauer - only – Zustand. Dieses einfache Beispiel werde ich mit euch nun aber nicht machen... es wird Zeit, das wir uns mit etwas Größerem

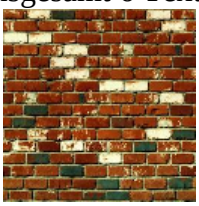
befassen. (Natürlich dürft ihr euch das kleine Sample auch downloaden) Und zwar möchte ich mit euch einen kleinen Raum beleuchten. Dieser sieht unbeleuchtet so aus (igitt):

Nicht gerade das, was unsere shooter - verwöhnten Augen als hübsch betrachten. Aber mit ein wenig Licht, sieht das ganze schon besser aus:

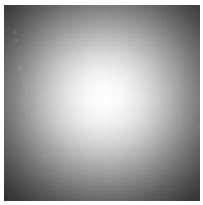
Man erkennt den kleinen Raum fast nicht wieder, gelle? Zugegeben, das Licht scheint einfach aus der decke zu kommen und ist etwas arg gelb geraten, ich finde es für den Anfang aber ganz Ok. Falls man eine realistische Lichtquelle einbauen wollte wäre das auch nicht allzu schwer: Man bastle sich an die Decke n' kleinen Kasten, welchen man hübsch texturiert und somit wie ne Lampe aussehen lässt (falls es nur ne Glühbirne sein soll, muss man das ganze halt rundlich machen...) Ach ja: die Farbe des Lichtes kommt nicht von der Lichttextur! Man kann ja mit „glcolor3f“ nicht nur Objekte einfärben, sondern auch Texturen... nur das ihr es wisst.

Diese Szene wollen wir also nun nachbauen. Dazu muss ich nun aber noch was sagen: wir sind nun schon im 7. Teil dieses Kursen. Daher werde ich ab nun bei größeres Sachen nicht mehr alles haarklein aufdröseln und einzeln erklären. Falls ihr Fragen habt z.b., wie man mit Texturen umgeht oder wie man die Initialisierung am besten macht, dann seht in den vorherigen Kursteilen nach. Alles noch mal durchzukauen würde den Rahmen dieses Teils sprengen und sowohl mir, als auch einigen Usern den letzen Nerv kosten.
Fangen wir also an...
Wir benötigen insgesamt 6 Texturen:


boden.bmp


stein.bmp


decke.bmp


licht1.bmp


licht2.bmp


licht3.bmp

(Alle 6 Texturen sind auch in dem Package mit dem Sample in Originalgröße enthalten)

Dafür benötigen wir also auch 6 gluints:

```
var
  Form1: TForm1;
  bodenlicht : gluInt;
  wandlicht : gluInt;
  deckenlicht : gluInt;
  mauer : gluint;
  boden : gluint;
  decke : gluint;
```

Und auch 6 Laderoutinen:

```
procedure InitTextures;
var
  textur: PTAUX_RGBImageRec;
begin
  textur := auxDIBImageLoadA('stein.bmp');
  if not Assigned( textur ) then begin
    showmessage('Fehler beim Laden der Textur');
  end;

  glGenTextures(1, mauer);
  glBindTexture(GL_TEXTURE_2D, mauer);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_linear);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_linear);
  glTexImage2D(GL_TEXTURE_2D, 0, 3, textur^.sizeX, textur^.sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
textur^.data);


  textur := auxDIBImageLoadA('boden.bmp');
  if not Assigned( textur ) then begin
    showmessage('Fehler beim Laden der Textur');
  end;

  glGenTextures(1, boden);
  glBindTexture(GL_TEXTURE_2D, boden);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_linear);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_linear);
  glTexImage2D(GL_TEXTURE_2D, 0, 3, textur^.sizeX, textur^.sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
textur^.data);



  textur := auxDIBImageLoadA('decke.bmp');
```

```
  if not Assigned( textur ) then begin
    showmessage('Fehler beim Laden der Textur');
  end;

  glGenTextures(1, decke);
  glBindTexture(GL_TEXTURE_2D, decke);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_linear);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_linear);
  glTexImage2D(GL_TEXTURE_2D, 0, 3, textur^.sizeX, textur^.sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
textur^.data);

  textur := auxDIBImageLoadA('licht1.bmp');
  if not Assigned( textur ) then begin
    showmessage('Fehler beim Laden der Textur');
  end;

  glGenTextures(1, bodenlicht);
  glBindTexture(GL_TEXTURE_2D, bodenlicht);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_linear);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_linear);
  glTexImage2D(GL_TEXTURE_2D, 0, 3, textur^.sizeX, textur^.sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
textur^.data);



  textur := auxDIBImageLoadA('licht2.bmp');
  if not Assigned( textur ) then begin
    showmessage('Fehler beim Laden der Textur');
  end;

  glGenTextures(1, wandlicht);
  glBindTexture(GL_TEXTURE_2D, wandlicht);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_linear);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_linear);
  glTexImage2D(GL_TEXTURE_2D, 0, 3, textur^.sizeX, textur^.sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
textur^.data);



  textur := auxDIBImageLoadA('licht3.bmp');
  if not Assigned( textur ) then begin
    showmessage('Fehler beim Laden der Textur');
  end;

  glGenTextures(1, deckenlicht);
  glBindTexture(GL_TEXTURE_2D, deckenlicht);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MAG_FILTER,GL_linear);
  glTexParameteri(GL_TEXTURE_2D,GL_TEXTURE_MIN_FILTER,GL_linear);
  glTexImage2D(GL_TEXTURE_2D, 0, 3, textur^.sizeX, textur^.sizeY, 0, GL_RGB, GL_UNSIGNED_BYTE,
textur^.data);


end;
```

Ich liebe Seitenlange Code - Posts... Na ja: nützt ja nun nichts... ich denke aber mal, das sich das, was dort steht von selber erklärt (insbesondere, wenn man die Teile 5 und 6 gelesen hat, dürfte einem dieser Code nichts neues sein).

Wir erzeugen nun also unseren Raum:

```
procedure render;
begin
glClear(GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT); //Farb und Tiefenpuffer löschen
glLoadIdentity;
```

```
glcolor3f(1,1,1);
glBindTexture(GL_TEXTURE_2D, mauer);

glbegin(gl_quads);                  //Wände
gltexcoord2f(0,0);                  //Vorne (auf den Shots nicht sichtbar)
glvertex3f(-1,-0.5,1);
gltexcoord2f(0,1);
glvertex3f(-1, 0.5,1);
gltexcoord2f(1,1);
glvertex3f( 1, 0.5,1);
gltexcoord2f(1,0);
glvertex3f( 1,-0.5,1);

gltexcoord2f(0,0);                  //Hinten
glvertex3f(-1,-0.5,-5);
gltexcoord2f(0,1);
glvertex3f(-1, 0.5,-5);
gltexcoord2f(1,1);
glvertex3f( 1, 0.5,-5);
gltexcoord2f(1,0);
glvertex3f( 1,-0.5,-5);

gltexcoord2f(0,0);                  //Rechte Seite
glvertex3f( 1,-0.5,-5);
gltexcoord2f(0,1);
glvertex3f( 1, 0.5,-5);
gltexcoord2f(2,1);
glvertex3f( 1, 0.5,1);
gltexcoord2f(2,0);
glvertex3f( 1,-0.5,1);

gltexcoord2f(0,0);                  //Linke Seite
glvertex3f(-1,-0.5,-5);
gltexcoord2f(0,1);
glvertex3f(-1, 0.5,-5);
gltexcoord2f(2,1);
glvertex3f(-1, 0.5,1);
gltexcoord2f(2,0);
glvertex3f(-1,-0.5,1);
glend;

glBindTexture(GL_TEXTURE_2D, boden);
glbegin(gl_quads);                  //Der Fußboden
gltexcoord2f(0,0);
glvertex3f(-1,-0.5,-5);
gltexcoord2f(1,0);
glvertex3f( 1,-0.5,-5);
gltexcoord2f(1,3);
glvertex3f( 1,-0.5,1);
gltexcoord2f(0,3);
glvertex3f(-1,-0.5,1);
glend;

glBindTexture(GL_TEXTURE_2D, decke);    //Die Decke
glbegin(gl_quads);
gltexcoord2f(0,0);
glvertex3f(-1, 0.5,-5);
gltexcoord2f(1,0);
glvertex3f( 1, 0.5,-5);
gltexcoord2f(1,2);
glvertex3f( 1, 0.5,1);
gltexcoord2f(0,2);
glvertex3f(-1, 0.5,1);
glend;

SwapBuffers(form1.myDC);                //scene ausgeben
end;
```

Soweit ebenfalls noch nicht viel neues (kleiner Tip: Saugt euch das Sample, anstatt diesen ganzen Code abzuschreiben).
Das einzige, was evtl. neu sein könnte ist, das bei „gltexcoord“ manchmal Werte >1 auftauchen. Dies hat einen sehr einfachen Grund: wird ein Wert >1 angegeben, wird die Textur in diese Richtung wiederholt. Wenn ich nämlich in diesem Raum immer nur „1“ genommen hätte, dann sähen die Texturen sehr gestreckt aus, was nicht gerade hübsch aussieht.
Wenn wir nun unsere Szene starten, dann bekommen wir etwa ein Bild, welches dem ersten, unbeleuchteten Bild unserer Szene arg gleicht (besser: Es *ist* die unbeleuchtete Szene ;-))

Nun müssen wir also noch Licht ins Dunkel bringen. Hierzu verwende ich den Blending-Modus „glBlendFunc(gl_src_color,gl_src_color);“ Dieser hat dann bei uns folgende Wirkung: Es wird von der bereits vorhandenen Szene nur soviel genommen, wie Helligkeit auf der Lichttextur ist, das bedeutet: wo die Lichttextur schwarz ist, sieht man nichts. Hingegen in der Mitte würde bei „voller“ Farbe einfach nur ein heller Flecken sein. Um diesem vorzubeugen nehme ich für das Gelb nicht „glcolor3f(1,1,0)“ sondern „glcolor3f(0.9 , 0.9, 0.5“ für den Boden bzw. „glcolor3f(0.8 , 0.8, 0.4)“ für den Rest. Damit sollte auch der restliche Code verständlich werden (zwischen dem „SwapBuffers“ und der Decke einzufügen):

```
glenable(gl_blend);                          //Blending einschalten
gldisable(gl_depth_test);                    //Tiefentest aus
glBlendFunc(gl_src_color,gl_src_color);      //Blending-funk setzen
glcolor3f(0.9,0.9,0.5);                      //Lichtfarbe für den Boden

glBindTexture(GL_TEXTURE_2D, bodenlicht);    //die Bodenbeläuchtung

glbegin(gl_quads);
gltexcoord2f(0,0);
glvertex3f(-1,-0.5,-5);
gltexcoord2f(1,0);
glvertex3f( 1,-0.5,-5);
gltexcoord2f(1,2);
glvertex3f( 1,-0.5,1);
gltexcoord2f(0,2);
glvertex3f(-1,-0.5,1);
glend;

glcolor3f(0.8,0.8,0.4);                      //Lichtfarbe für den Rest der Szene

glBindTexture(GL_TEXTURE_2D, wandlicht);     //Die Wände beläuchten
glbegin(gl_quads);
gltexcoord2f(0,0);                  //Vorne
glvertex3f(-1,-0.5,1);
gltexcoord2f(0,1);
glvertex3f(-1, 0.5,1);
gltexcoord2f(1,1);
glvertex3f( 1, 0.5,1);
gltexcoord2f(1,0);
glvertex3f( 1,-0.5,1);

gltexcoord2f(0,0);                  //Hinten
glvertex3f(-1,-0.5,-5);
gltexcoord2f(0,1);
glvertex3f(-1, 0.5,-5);
gltexcoord2f(1,1);
glvertex3f( 1, 0.5,-5);
gltexcoord2f(1,0);
glvertex3f( 1,-0.5,-5);

gltexcoord2f(0,0);                  //Rechts
glvertex3f( 1,-0.5,-5);
gltexcoord2f(0,1);
```

```
glvertex3f( 1, 0.5,-5);
gltexcoord2f(2,1);
glvertex3f( 1, 0.5,1);
gltexcoord2f(2,0);
glvertex3f( 1,-0.5,1);

gltexcoord2f(0,0);                    //Links
glvertex3f(-1,-0.5,-5);
gltexcoord2f(0,1);
glvertex3f(-1, 0.5,-5);
gltexcoord2f(2,1);
glvertex3f(-1, 0.5,1);
gltexcoord2f(2,0);
glvertex3f(-1,-0.5,1);
glend;


glBindTexture(GL_TEXTURE_2D, deckenlicht);  //Die Decke beleuten
glbegin(gl_quads);                          //bzw verdunkeln
gltexcoord2f(0,0);
glvertex3f(-1, 0.5,-5);
gltexcoord2f(1,0);
glvertex3f( 1, 0.5,-5);
gltexcoord2f(1,2);
glvertex3f( 1, 0.5,1);
gltexcoord2f(0,2);
glvertex3f(-1, 0.5,1);
glend;


glenable(gl_depth_test);            //Tiefentest an
gldisable(gl_blend);                //Belending aus
```

Das war's auch „schon". Damit bekommen wir unsere schon deutlich schickere Szene. Im übrigen: man kann die Lichtexturen logischerweise auch wie Scheinwerfer verschieben. Dieses Funktionier ganz normal wie bei anderen Texturen über die Texturmatrix. Ein schickes Beispiel dafür werde ich auch bald schreiben (findet ihr dann auf der Homepage).

Nachwort

Damit hätten wir also das Thema Lichter mehr oder weniger abgeschlossen. Ich weiß, das dieses nicht gerade mehr so einfach war, wie bislang, aber ihr packt das schon. Ladet euch einfach die Samples runter und versucht zu verstehen, was ich dort alles gemacht habe und ich hoffe, das ihr Teil 8 auch noch lest ;-). Im 8. Teil wird es wahrscheinlich um Kamerabewegungen gehen... oder evtl. auch noch was zum Thema Texturen... mal sehen.

Mr_T