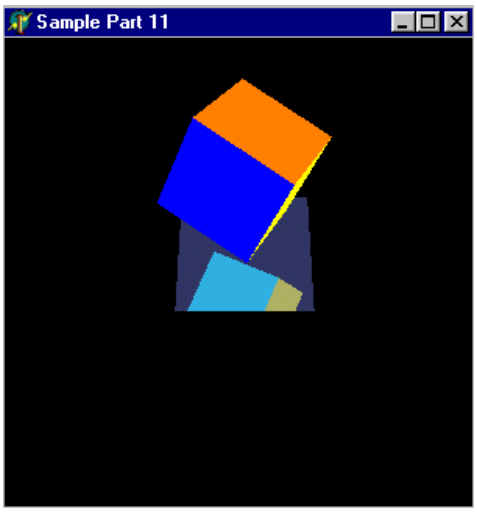
Open Gl - Delphi – Kurs Teil 11: Der Stencil-Puffer

Einleitung

Nun sitze ich hier in meinem viel zu warmen Zimmer - mich halb totschwitzend - und soll euch nun was über eines der Grauenhaftesten Dinge unseres Lebens erzählen: Masken!
Masken verfolgen uns überall: Sei es als Kind, wenn man es hasst sich zu verkleiden (ging mir immer so) oder auch wenn man sich unsere Politik ansieht, welche sich im Hinblick auf unsere Wirtschaftliche Lage hinter einer Maske der Ratlosigkeit versucht zu verstecken...
Und soll ich euch was sagen: Masken verfolgen uns bis in unsere heile Open Gl –Welt (nein, ich meine nicht die maskenhafte Gestallt, welcher meiner Mutter ähnelt und mir sagt, ich solle nachts lieber schlafen, statt am PC zu sitzen...)
Nein, ich meine Zeichenmasken, welche man über den Stencil-Puffer realisiert, womit wir bei unserem eigentlichem Thema wären...
Der Stencil-Puffer ist an sich nichts anderes, als eine Art Array, welches die Ausmaße unserer Zeichenfläche hat. In dieses Array kann man Informationen über unsere Szene schreiben... bloß leider nur recht begrenzt: Meistens ist der Stencil-Puffer pro Bildschirmpixel nur 1-8 Bit groß! Und wozu das Ganze? Na ja: ihr kennt doch den Tiefentest mittlerweile, oder? In diesem Werden doch Infos gespeichert, wie Weit weg ein Objekt ist und alles, was Auf demselben Pixel gezeichnet werden würde und weiter weg liegt, wird nicht gepaintet...
So ähnlich ist das mit dem Stencil-Test auch: Man kann in den Stencil-Puffer Werte schreiben und bei allen nachfolgenden Objekten wird geguckt (je nach Einstellung), ob an der Stelle, wo das Objekt gezeichnet werden soll, überhaupt gezeichnet werden darf... man kann also Pixel vor dem Überschreiben schützen!
Bloß wozu soll das gut sein? Ich gebe euch ein Beispiel:
Angenommen ihr baut eine Indoor-Map oder besser sogar noch einfach nur ein Geflecht aus mehreren Räumen. In diesem Falle müsst ihr doch sicherlich ein paar Verbindungsgänge zwischen den Räumen einbasteln, oder? Das Ding ist ja nun: wenn so eine Verbindung nicht die ganze Wand einnimmt (was ja recht unpraktisch wäre), dann ist man dazu verdonnert statt eines Quads für die betreffende wand 2,3 oder sogar 4 kleinere zu Zeichnen, um ein Brauchbares Loch in der wand zu hinterlassen (Wenn die Eingänge quadratisch sind, geht es ja noch... stellt euch das aber mal mit einem total unförmigem Durchgang vor! Eine Katastrophe...)
Und hier setzt unsere Maske ein: wenn wir den Stencil-Puffer verwenden, dann können wir die wand komplett am Stück zeichnen... wir müssen nur an die Stelle, wo der Eingang hin soll, zuvor eine Maske platzieren und den Bereich vor dem überschreiben durch die Wand schützen... Auf diese Art und weise braucht man für einen normalen Durchgang nur 2 Quads (Wand & Maske) statt 4: eine Extreme Erleichterung (besonders wenn man mit Lightmaps arbeitet und eventuell zig Texcoords anpassen müsste.....)
So, lange genug gelabert (mache ich ja so gerne), ich denke wir sollten mal Anfangen...

Der erste Kontakt

So, nun will ich euch also auch noch zeigen, wie man diese Geniale Technik überhaupt verwendet... ich könnte nun zwar das Sample von oben bringen (die Sache mit dem Loch in der Wand), aber vorerst machen wir etwas leichteres. (Wenn auch nicht weniger eindrucksvolles)
Dazu erstmal ein kleines Bildchen:



Tja, was sehen wir denn da: einen netten kleinen (im Sample rotierenden) Würfel, welcher über eine kleine Fläche gespiegelt wird... die Frage ist bloß, was das mit der Maske zu tun hat...
An sich ist es ganz einfach: Die Spiegelung erfolgt dadurch, dass ich den Würfel an sich einfach 2x zeichne, bloß einmal halt an der x / z-Fläche gespiegelt... oder anders ausgedrückt: Ich habe beim Spiegelbild einfach mit glscalef(1,-1,1) alle Y-Werte umgedreht. Die Maske kommt nun insofern zum Einsatz, dass das Spiegelbild auch wirklich nur da gespiegelt wird, wo der Spiegel ist und nicht das ganze Spiegelbild zu sehen ist. Alles klar?
Na ja, ich gebe euch hier erstmal ein wenig Source, so dass ich euch besser quälen kann ;-)

```
glEnable(GL_STENCIL_TEST);        //Stencil-Test an
glStencilFunc(GL_ALWAYS, 1, 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);
glColorMask(false, false, false, false);
gldepthmask(gl_false);

drawspiegel;          //Spiegel in den Stencil Buffer zeichnen

gldepthmask(gl_true);

glColorMask(True, True, True, True);
glStencilFunc(GL_EQUAL, 1, 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);

glpushmatrix;
glScalef(1,-1,1);
gltranslate(0,1.5,0);
glrotate(rot,1,1.5,1);

drawwuerfel;    //Gespiegelten Würfel zeichnen

glpopmatrix;

gldisable(gl_stencil_test);
glenable(gl_blend);
```

```
glblendfunc(gl_src_alpha, gl_one_minus_src_alpha);
glcolor4f(0.4,0.4,0.8,0.5);
drawspiegel;     //Spiegel zeichnen
gldisable(gl_blend);

gltranslate(0,1.5,0);
glrotate(rot,1,1.5,1);
drawwuerfel;    //Original-Würfel Zeichnen
```

Na denn wollen wir mal... also: das übel beginnt gleich am Anfang mit glenable(gl_stencil_test), welches nichts anderes zu tun hat, als den Stencil-Test zu aktivieren... soweit ja noch gut und schön, aber was nun kommt, hat es in sich:

glStencilFunc() ist dafür verantwortlich, dass der Stencil-Test richtig ausgewertet wird. Dabei meint der erste Parameter, auf welche Art und Weise der Test bestanden werden kann: Möglichkeiten sind z.B. gl_always (der Test wird immer bestanden), gl_equal (der Test wird nur bestanden, wenn der Vorhandene Stencil-Wert mit dem Referenzwert übereinstimmt) oder auch gl_less (Test nur dann bestanden, wenn der Stencil-Wert kleiner als der Referenzwert ist)
Hä? Was bitte ist der Referenzwert? Keine Panik, dazu komme ich nun... der Referenzwert ist der 2. Parameter bei glstencilfunc ;-)
Ich habe euch doch vorhin erzählt, dass der Stencil-Puffer einfache Integer bzw. Bytewerte für jedes Pixel speichert, oder? (Sollte ich es nicht getan haben: es ist so *g*)
Wenn nun ein Objekt gezeichnet werden soll, dann wird immer geguckt, welcher Wert im Stencil-Puffer steht und dieser Wert wird mit dem Referenzwert (halt der 2. Parameter von glstencilfunc) verglichen und dementsprechend Ausgewertet... Angenommen Pixel xy hat einen Stencil-wert von 1 und nun wird ein neues Obejkt gezeichnet, dessen Stencil-Funk so aussieht:
glStencilFunc(GL_EQUAL, 1, 1);
Wird es gezeichnet? Ja, das wird es, da der Referenzwert (1) identisch ist, mit dem Wert, der schon im Stencil-Buffer steht... simpel, oder?
(Ok, ich gebe zu, nicht wirklich... aber denkt mal etwas drüber nach... ist nicht allzu schwer zu verstehen... weiß bloß nicht, wie ich es erklären soll...)

Aber für was steht denn der allzu mysteriöse 3. Parameter? Nun, das ist einfach:
Der 3. Parameter gibt an, was in den Stencil-Puffer geschrieben wird, wenn der Stencil-Test bestanden wird... einfach, oder?

So, weiter im Quelltext. Die nächste Zeile sollte euch zwar momentan noch fremd sein, ist aber verdammt einfach zu beschreiben:
Mit dem Open Gl - Befehl glcolormask kann man nämlich ganz simpel einzelne Farbspuren (RGBA) einfach Abschalten... in diesem Falle habe ich überall false stehen, was bedeutet, dass die nächsten Objekte nicht grafisch dargestellt werden. Dieses Brauche ich deshalb, weil ich ja zunächst einmal nur meine Maske zeichnen will, aber ja noch nicht den Spiegel seber, weil wir diesen später blenden müssen.

Was dann passiert ist relativ Simpel zu beschreiben:
Zunächst sage ich Open Gl, dass von dem kommenden Objekten keine Tiefeninfos gespeichert werden sollen (wäre auch widersinnig, wenn ein Objekt hinter dem Spiegel den Stencil -Test besteht, aber dann durch den Tiefentest rauscht...), dann zeichne ich die Spiegelfläche in den Stencil Buffer (habe ich in eine externe Prozedur ausgelagert... sollte für

euch aber nicht schwierig sein, ein einfaches Quadrat zu zeichnen), sie wird ja aber nicht angezeigt, und zu guter Letzt aktiviere ich die Speicherung von Tiefeninfos wieder.
So, damit hätten wir also unsere Spiegelfläche als Maske im Stencil-Puffer... nun müssen wir unseren gespiegelten Würfel zeichnen.
Um die nötigen Voreinstellungen dafür zu Treffen sind die nächsten 3 Zeilen da:
Zunächst aktivieren wir wieder das Zeichen aller Farbkanäle – schließlich wollen wir unseren Würfel ja auch sehen...
Dann bearbeiten wir die Einstellung das Stencil-Puffers. Wenn ihr oben gut aufgepasst habt, dann wisst ihr auch, was die Zeile „glStencilFunc(GL_EQUAL, 1, 1);“ bewirkt:
Alle Objekte, die nun gezeichnet werden, werden nur dann dargestellt, wenn der Wert im Stencil-Puffer genau „1“ ist. (bekanntermaßen ist der wert im Puffer genau dort „1“, wo wir unsere Spiegelflächenmaske hinplatziert haben)
Das letzte, was wir nun noch machen müssen ist, dass wir nun mit Hilfe von „glStencilOp(GL_KEEP, GL_KEEP, GL_KEEP);“ Open Gl sagen, dass der Stencil-Puffer selber nicht geändert werden soll... komme, was wolle :-P

Was nun kommt, hat mit der Stencil-Technik nicht viel zu tun:
Wir puschen zunächst unsere Matrix (damit wir sie später wieder unverändert parat haben), missbrauchen dann den glscale-Befehl, um alles Folgende nach unten zu Spiegeln, verschieben dann den Würfel an die richtige Stelle und lassen diesen rotieren.

Soweit angekommen zeichnen wir diesen dann auch (ich habe auch dieses in eine neue Prozedur ausgelagert, weil ich so schreibfaul bin) und denn holen wir zu guter letzt noch unsere Matrix Mithilfe von glpopmatrix zurück.

Ab nun wird alles nur noch Formsache:
Ich deaktiviere den Stencil-Test (das Spiegelbild ist nun ja schon korrekt da), zeichne Mithilfe von Blending die eigentliche Spiegelfläche und zu guter Letzt muss ja auch noch der Original-Würfel her.

Doch, wenn man das ganze nun startet, dann sieht das nicht ganz so aus, wie ihr euch das vorgestellt habt, oder? Ist auch kein Wunder, da wir bei der Initialisierung noch was ändern müssen. Dazu nehmt euch mal eure SetupPixelFormat-Prozedur vor und sucht dort nach der Zeile „cDepthBits = “. Darunter müsst ihr nun eine neue Zeile einfügen:

```
cstencilbits := 8;
```

Damit weisen wir Open Gl an, einen 8 Bit großen Stencil-Buffer anzulegen. (Für unsere zwecke würden hier auch 1 oder 2 reichen)
So... nun müssen wir noch den Stencil-Puffer irgendwie lehren – etwa so wie den Tiefenpuffer auch. Dazu modifizieren wir unser „glclear“ einfach etwas, so dass es nun so aussieht:

```
glClear(GL_COLOR_BUFFER_BIT or GL_DEPTH_BUFFER_BIT or
GL_STENCIL_BUFFER_BIT);
```

Fertig!

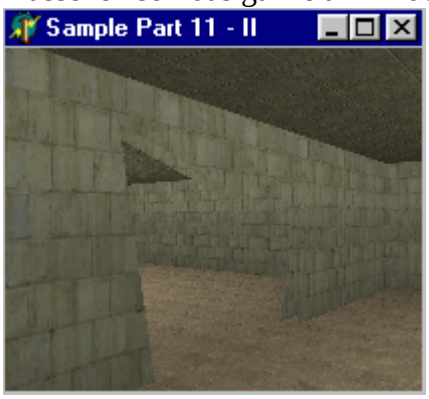
Ach ja: ich habe im Sample ja einen bestimmten Winkel zur Kamera, den ihr bislang nicht gesetzt habt... die Verschiebung und den Winkel habe ich so eingebastelt:

```
gltranslate(0,0,-7);
glrotate(60,1,0,0);
```

Mit diesen Infos solltet ihr - sofern ihr dazu in der Lage seit, ein Quadrat und einen Würfel zu painten - das Sample gut nachbauen können.

Und Cut!

So, nun kommen wir zu einem etwas größerem Beispiel, bei dem man die Arbeit mit dem Stencil-Puffer sehr schon sehen kann.
Mein Ziel ist recht einfach:
Ich will einen Raum basteln, welcher durch eine Mittelwand durchtrennt ist. Diese Mittelwand hat nun aber ein durch den Stencil-Puffer ausgeschnittenes, leicht unförmiges Loch, welches man sonst nur mit viel Mühe aus vielen Dreiecken und Vierecken zusammenbekommt...
Aussehen soll das ganze am Ende etwa so:



Wie ihr den Raum hinbekommt ist klar, oder? An sich ist es nur ein großer Quader, der 16 Einheiten lang, 8 Einheiten breit und 2 Einheiten hoch ist.

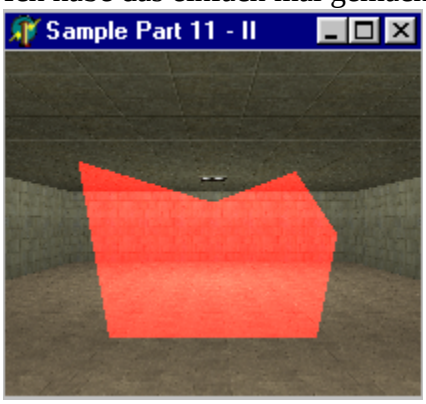
Wir haben also unseren Raum (Bild unten), in den wir unsere Wand einbauen möchten.

Was machen wir also?
Nun ja: nachdem der Raum komplett gezeichnet wurde, zeichnen wir zunächst die Maske in den Stencil-Puffer. Ich habe dazu einfach ein unförmiges Polygon genommen. Dieses geschieht so:

```
glEnable(GL_STENCIL_TEST);
glStencilFunc(GL_ALWAYS, 1, 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_REPLACE);
glColorMask(false, false, false, false);
gldepthmask(gl_false);

glbegin(gl_polygon);            //Maske zeichen
glvertex3f(-1,0,0);
glvertex3f(1,0,0);
glvertex3f(1.2,1,0);
glvertex3f(0.8,1.6,0);
glvertex3f(0,1.3,0);
glvertex3f(-1.3,1.7,0);
glend;
```

Damit hätten wie die Maske im Puffer... um zu testen, wie die Maske nun genau aussieht, bevor man weitermacht, kann man sich die ja mal anzeigen lassen. Dazu muss man vor diesem Code die Texturen deaktivieren, die Farbkanäle an lassen und eine Farbe wählen (sowie den Stencil-Test nach diesem Code beenden).
Ich habe das einfach mal gemacht:



Das ist also unsere Maske. Überall dort, wo nun dieses rote etwas auf dem Bild ist (ihr werdet es wahrscheinlich nicht so sehen... habe den Shot wie gesagt nur zur Verdeutlichung gemacht) befindet sich nun im Stencil-Puffer ne „1“... an allen anderen Stellen eine „0“.
Nun sollte es ein leichtes sein, da noch ne Wand mit dem nötigen Licht hinzumalen. Der Code ist nicht schwer zu verstehen:

```
gldepthmask(gl_true);
glColorMask(True, True, True, True);
glStencilFunc(GL_Equal, 0, 1);
glStencilOp(GL_KEEP, GL_KEEP, GL_Keep);

gldepthmask(gl_false);
wall.bind;
glbegin(gl_quads);          //Mittelwand
gltexcoord2f(0,0);
glvertex3f(-4,0,0);
gltexcoord2f(8,0);
glvertex3f(4,0,0);
gltexcoord2f(8,1);
glvertex3f(4,2,0);
```

```
gltexcoord2f(0,1);
glvertex3f(-4,2,0);
glend;
gldepthmask(gl_true);


glenable(gl_blend);
glblendfunc(gl_dst_color, gl_zero);
light2.bind;
glbegin(gl_quads);          //MittelWand - Licht
gltexcoord2f(0,0);
glvertex3f(-4,0,0);
gltexcoord2f(1,0);
glvertex3f(4,0,0);
gltexcoord2f(1,1);
glvertex3f(4,2,0);
gltexcoord2f(0,1);
glvertex3f(-4,2,0);
glend;
gldisable(gl_blend);

gldisable(GL_STENCIL_TEST);
```

Was wir an sich machen ist hoffentlich klar: wir stellen den Stencil-Test so um, dass nun nur noch dort gezeichnet werden können, wo „0“en im Puffer sind nachdem wir die Wand dahingezeichnet haben, konnte ich es mit nicht verkneifen, das ganze auch noch zu beleuchten. Easy, oder? Ich finde das ganze macht schon was her: (auf jeden mehr als ein schnöder, gerader Durchgang *g*)

Nachwort

So, damit wären wir auch schon wieder fertig. Sorry an alle, die solange auf diesen Teil warten mussten, aber der Schulbeginn hat die Arbeit an diesem Kursteil doch verlangsamt. Merkt man besonders daran, dass der Anfang zu einer Zeit geschrieben wurde, als es noch richtig heiß hier in Norddeutschland war und wir nun schon Mitte September haben.
Des weiteren gab es während des Schreibens dieses Teils meinen 18. zu feiern, feiert mit mir!
Hoffe, wenn ihr auch beim nächsten mal wieder reinguckt. Das Thema steht noch nicht 100%, da ich noch etwas zwischen Zwei Themen schwanke... ihr werdet schon sehen :-)

Mr_T