Michis Prüfungsprotokoll
Computer Graphik und Interaktive Computer Graphik

11. April 2007

Ein paar Tips und Hinweise vorab.

Die Prüfer (oder auch: Die Meister der Graphik)

Was ihr inzwischen ohnehin in zich Prüfungsprotokollen gelesen habt: Professor Stamminger ist ein ultra-freundlicher Prüfer, der sich alle Mühe gibt, den Prüfling nicht zu verunsichern oder zu beunruhigen. Er stellt sehr faire Fragen und hilft sofort, wenn man nicht weiter weiß, ohne diese Hilfestellung am Ende anzukreiden. Was ich gehört und gelesen habe und aus eigener Erfahrung auch bestätigen kann (Scheinprüfung), verhält es sich mit Professor Greiner genau gleich.

Vorbereitung (oder wie man die letzte Henkersmahlzeit so richtig genießt)

Als Vorbereitung für die Prüfung empfehle ich euch:

- ein ehrliches Interesse am Stoff, das nicht bei den Sachen der Vorlesung auffhört (Dann kann eigentlich nichts schief gehen)
- die beiden super guten Bücher: Real-Time Rendering (auch für die in CG behandelten Bereiche) und Fundamentals of Computer Graphgics (für alles was in Real-Time Rendering nicht erklärt wird)
- sich vor allem komplexere Zusammenhänge anhand von Apllets im Internet oder per Rumspielen mit 3D Programmen klar zu machen
- sich beim Lernen einen Fragen Katallog erstellen mit Sachen, die man noch nicht wirklich verstanden hat, diesen dann gegen Ende mit Freunden durch zu gehen und eventuell bei Prof. Stamminger oder Greiner um eine Gelgenheit bitten die hartnäckigen Fragen den Meistern selbst zu stellen

- ganz wichtig: lernt denn Stoff nicht nur alleine, sondern geht ihn mit Freunden (am besten Leute, die auch vor der Prüfung stehen durch). Es gibt keine bessere Vorbereitung als die Sachverhalte anderen Menschen zu erklären. Im Notfall tun es auch eure Eltern oder Geschwister, wenn ihr sadistisch veranlagt seid versteht sich. Einerseits beweist ihr so Grundlagenwissen, da ihr natürlich alles super ausführlich erklären müsst, andererseits werden sie aber nicht wie eure Conlernenden fiese Querfragen stellen können.

- last but not least empfehle ich euch natürlich mein ultra-cooles[1] Studentenskript, in dem ich versucht habe alle Zusammenhänge so zu erklären, wie ich sie schließlich verstanden habe. Ihr könnt es euch bei der FSI besorgen[2].

So nun aber zum eigentlichen Protkoll.
Nachdem Prof. Stamminger mich mit Fragen nach meinem kommenden Jahr in Japan beruhigte und mich mit dem Statement: "Also schön, sie wollen sich also über Geometrische Modellierung und Visualisierung prüfen lassen", wieder beunruhigte,
malte er ein Viereck auf ein Blatt und in dieses ein kompliziertes konkaves Polygon. Dann fragte er mich was am Computer passiere, wemm man dieses rasterisieren möchte. Er wollte scheinbar auf

[1] Polygon Clipping

hinaus, doch ich habe zuerst gesagt, dass man vorher eventuell komplett außerhalb liegende Objekte culled. Dann ging ich zum eigentlichen Clipping über und überlegte kurz, weil das Polygon konkav war. Die konkave Eigenschaft macht jedoch nichts aus; denn nur falls die Clip-Fläche selbst konkav ist, hab ich Probleme. Also Polygon Clipping erklärt und anhand einer Edge exemplarisch vorgespielt. Dann ließ er mich

[2] Scanline und Seed-Fill

mit den Edge Tables, den inkrementellen Updates und den Interpolationsvorteilen, erklären.
Hab allerdings fälschlicherweise gesagt, dass man beim Scanlien Algorithmus zuerst die Edges mit Bresenham zeichnet. Das muss man natürlich nicht machen (Außer bei Seed-Fill).

Was ich zusätzlich nicht gewusst habe, ist dass anstelle nur 1/slope als x-Update zu nehmen, man eine Abart der Bresenham-Idee für das Update der aktuellen x-Position benutzt. Dazu wollte er dann wissen, welche Besonderheiten gegenüber der standard Bresenham Methode ich dabei berücksichtigen muss. Da bin ich aber leider nicht draufgekommen. Irgendwie hat er ne Linie

[1] **Leser**: Mensch! Schon mal was von Bescheidenheit gehört?
Autor: Na komm Leser, dafür hab ich's doch als letzten Punkt aufgeführt!
[2] no pun intended

gezeichnet die schräg im Bildschirm lag, um mir das Problem zu illustrieren. Ich habe allerdings nur genickt, aber im Nachhinein nicht wirklich verstanden, warum man die x-Werte der Linie nicht mit dem standard Bresenham-Update hätte berechnen können :(. Am besten ihr schaut das nochmal nach, wenn ihr das hier lest. Anschließend wollte er, dass ich

[3] Gourad und Phong Shading

anhand der Scanline Zeichnung erkläre. Ich habe einfach erzählt, was man jeweils mit der Scanline interpoliert, und dass bei Phong-Shading die eigentliche Lichtberechneung erst nachher kommt. Außerdem wurde ich gefragt, woher die Normalen an den Edges kommen (Durschnitt der anliegenen Face-Normalen). Nun nimmt er ein neues Blatt, malt 'ne Zeichnung hin und läßt mich

[4] Ray Tracing

mit all den ganzen lustigen Strahlen erklären. Gezögert habe ich ein bischen, weil eine Stelle, welche im Schatten lag, durch einen zurückkommenden Reflektionsstrahl dann doch beleuchtet wurde. Daraufhin wollte er wissen wie man Shading mit RayTracing realisiert, insbesondere die dabei notwendige Interpolation. Das Problem ist, dass man keine Scanline zur Verfügung hat. Die Lösung für Dreiecke ist es, Baryzentrische Koordinaten für die Interpolation zu verwenden (Hier bin ich nur mit Hilfestellung drauf gekommen).
Nun sollte ich auf das Polygon mittels

[5] Texture Mapping

eine Textur aufkleben. Hab' dann erklärt das man ein Mapping zwischen den Objektkoordinaten und den Texturkoordinaten finden muss und Beispiele dafür gegeben (siehe mein Skript für ausführliche verständliche :) Erklärungen).

Dann hat er gefragt was dabei für Probleme auftreten: Vor allem Aliasing. Habe dann als Beispiel diese zu beheben, MipMapping erklärt. Als ich ihm zusätzlich erklärt habe wie man das MipMap-Level bestimmt (grob: Pixel auf Textur projezieren - ergibt ein Parallelogramm - MipMap level anhand der längeren Seite auswählen. Die Idee ist, dass man so sieht wie viele Texel einen Pixel beeinflussen.) war er ziemlich beeindruckt. Als ich dann mit der kürzere Paralellogram Seite weitergemacht habe und Anistropic Filtering erklärt habe (siehe mein Skript) und dabei als Ausgangspunkt die Probleme die bei MipMapping immer noch auftreten verwendete (nur quadratische Texture Maps, problematisch wenn man schräg auf das Object schaut), hat er sogar gestaunt und sich gefreut =).
Schließlich kam er zu seinem Lieblingsthema, das er praktisch in jeder Prüfung abfragt, den

[6] Schatten

<a> Planare Schatten

Zuerst sollte ich alle Einschränkungen aufzählen. Ich wusste aber nur, dass eine Ebene für den Schatten benötigt wird (PLANAR), scheinbar gibt's allerdings noch mehr. Ich weiß nicht mehr genau, was er noch gemeint hat. Ich glaube etwas mit der Lichtquelle und der Perspektive. Anschließend die beiden standard Probleme erklärt: Z-Buffer Fighting, da der Schatten denselben Tiefenwert hat wie die Ebene, auf der er gezeichnet wird. (Durch numerische Instabilitäten der Maschinengenauigkeit werden Werte immer leicht verutschen) Dann als zweites das Zeichnen des Schattens außerhalb der Ebene. Eine Lösung für das erste Problem ist das Verwenden eines Polygon Offsets (die Tiefenwerte des Schattens mit einem Offset anheben, aber im FrameBuffer in der tatsächlichen Tiefe zeichnen). Eine Lösung für beides ist der Stencil Buffer, indem ich mir die Ebene markiere und nur darauf zeichne (Wenn man dazu noch den Tiefentest ausschaltet, löst man auch das erste Problem).

<b> Shadow Maps

Ich setzte mein Auge in die Lichtquelle und speichere die Tiefenwerte in einer Textur. Alles was das Lichtauge sieht ist nicht im Schatten. Dann wollte er das *genaue* Mapping von einem Punkt in der Zeichnung zu einem Punkt in der Shadow Map (Also zeigen welcher Punkt mit welchem verglichen wird). Weiter wollte er wissen was den Werten entspricht, welche dort verglichen werden (Abstand zur Lichtquelle). Das ganze habe ich zum Glück anhand einer Zeichnung zusammengezimmert. Er meinte dann auch anschließend: "Sie glauben ja gar nicht wie viele daran scheitern!".

Dannach kamen wir zu Problemen bei Shadow Maps. Das sind im Prinzip dieselben wie bei Texturing, und das heißt vor allem Aliasing. Lösungen hier sind z.B. Adaptive Shadow Maps (ähnlich der MipMap Idee) oder Percentage Closest Filtering (nur Texel mit ähnlichen Tiefenwerten für die Interpolation verwenden).

<c> Shadow Volumes

Dann hat er gemeint solle ich den Raum, in dem wir uns gerade befinden mit Shadow Volumes und dann mit Shadow Maps rendern, und die Unterschiede und Probleme dabei erläutern. In Fortuna Gratia meinte er zusätzlich, dass sich auf dem Tisch zusätzlich noch ein kleiner Bonsaibaum steht.

Ich: "Mit vielen Blättern?"

Stamminger: "Ja mit vielen Blättern!"

Ich: "Dankeschön!"

Naja dann habe ich eben erklärt, dass bei Shadow Volumes einerseits die Komplexität (Runtime) verdammt krass ist, weil ich für jedes Bonsaibaumblatt ein

eigenes Volume rendere, und dass andererseits der Stencilbuffer most likely überlaufen wird (der hat ja nur 8 bit!). Bei Shadow Maps auf der anderen Seite habe ich in diesem Fall praktisch keine Probleme, da die Lichtquelle in Stamminergs Büro nämlich direkt über dem Prüfungstisch hängt (Miner's Lamp). Also der in der Vorlesung besprochene Idealfall, bei dem fast kein Aliasing auftritt. :-)

Dann durfte ich 30 Sekunden raus gehen, dann wieder rein und er meinte, Ich habe auf jedenfall bewiesen, dass ich die Konzepte und Ideen der Computer Graphik verstanden und durchschaut habe (In so einem Fall ist es dann scheinbar auch egal wenn man ein paar Sachen nicht genau wusste).

Note: the best there is