Informatica Instituut, Faculteit Wiskunde en Informatica, UU.
In elektronische vorm beschikbaar gemaakt door de TBC van A−Eskwadraat.
Het college INFOGR werd in 2003/2004 gegeven door René van Oostrum.

Uitwerking[1] Graphics (INFOGR) 11 november 2003

Opgave 1. Scan-line Rasterization

a) In hoofdstuk 12 van het bij het college gebruikte boek hebben we gezien hoe we geprojecteerde polygonen efficiënt kunnen tekenen met behulp van het *scan-line* algoritme. Wat is in dit verband *scan-line coherence*, en hoe maakt het scan-line algoritme hier gebruik van?

Antwoord: Scan-line coherence wil zeggen dat de x-coordinaat van een edge op een zekere scanline gerelateerd is aan de x-coordinaat van dezelfde edge op de vorige scanline. Door vooraf te bepalen wat de verhoging of verlaging van de x-coordinaat is bij het overgaan van de ene scanline naar de andere kan de x-coordinaat met een simpele optelling worden berekend (i.p.v. met een dure vermenigvuldiging en deling).

Score: De vraag bestaat uit twee delen, en het punt wordt alleen gegeven als beide delen naar voldoening zijn beantwoord.

b) Het scan-line algoritme maakt gebruik van de *Edge Table* en de *Active Edge Table*. Neem aan dat we een polygon projecteren waarvan één van de edges als eindpunten de punten $(2, 10)$ en $(5, 4)$ heeft.

1. Welke informatie (d.w.z. welke getallen, en wat stellen ze voor) wordt er van bovenstaande edge opgeslagen in de Edge Table?

   Antwoord: De getallen die worden opgeslagen zijn (1) de y-coordinaat van het hoogste eindpunt van de edge, ook wel y_2 genoemd. Waarde: 10. (2) de x-coordinaat van het laagste eindpunt van de edge, ook wel x_1 genoemd. Waarde: 5. (3) De x-increment, ofwel de verandering van de x-coordinaat bij het stappen van een scan-line naar de volgende, ook wel D_x genoemd. Waarde: $\frac{1}{2}$. De items/getallen hoeven niet noodzakelijk in deze volgorde gegeven te worden.

2. In welke entry van de Edge Table komt bovenstaande edge terecht?

   Antwoord: De entry die wordt bepaald door de y-coordinaat van het laagste punt, ook wel y_1 genaamd, dus in dit geval entry 4. Het getal 4 is hier voldoende als antwoord.

3. Hoe ziet de entry voor bovenstaande edge in de Active Edge Table eruit om het moment dat we de scanline $y = 8$ aan het tekenen zijn?

   Antwoord: De entry bevat (1) de y-coordinaat van het hoogste eindpunt van de edge, ook wel y_2 genoemd. Waarde: 10. (2) De x-coordinaat van het snijpunt van de edge met de onderhavige scanline. Waarde: 3. (3) De x-increment, ofwel de verandering van de x-coordinaat bij het stappen van een scan-line naar de volgende, ook wel D_x genoemd. Waarde: $\frac{1}{2}$.
   Bij deze vraag is het voldoende om alleen de getallen te geven, d.w.z. $(10, 3, \frac{1}{2})$ (of in een andere volgorde, corresponderend met de volgorde bij vraag 1a1.)

   Score: 1 punt als alle drie de vragen correct zijn beantwoord; 0.5 punt als er twee vragen correct zijn beantwoord; anders 0 punten.

[1] Deze uitwerkingen zijn met de grootste zorg gemaakt. In geval van fouten kan de TBC niet verantwoordelijk worden gesteld, maar wordt zij wel graag op de hoogte gesteld: `tbc@A-Eskwadraat.nl`

Opgave 2. Het z-buffer Algoritme

a) Leg de principes achter het z-buffer algoritme uit (daarbij hoef je niet in te gaan op een efficiënte implementatie m.b.v. een variant op het scan-line algoritme), en noem het grote voordeel van het z-buffer algoritme ten opzichte van de andere projectiemethode(n) die we eerder in het college hebben besproken.

Antwoord: Bij het z-buffer algoritme worden polygonen in een willekeurige volgorde getekend. Wanneer we polygon P tekenen, bepalen we voor iedere pixel waarop P wordt afgebeeld wat de corresponderende z waarde van het te projecteren punt op P is. Als die z waarde kleiner is dan de z-waarde die in de met de huidige pixel corresponderende entry in de z.g. z-buffer is opgeslagen, tekenen we de pixel opnieuw in de kleur van P, want P ligt dan voor het polygon dat het laatst op deze pixel is afgebeeld. Tevens updaten we dan de entry in de z-buffer met de huidige, zojuist bepaalde z waarde. In het andere geval (als de z-waarde van P voor de huidige pixel groter is dan de in de z-buffer opgeslagen z-waarde) hoeven we niets te doen voor deze pixel, omdat er dan een eerder getekend polygon voor P ligt.

Het voordeel van z-buffering boven het gebruik van BSP-bomen (of andere datastructuren) om een correcte projectievolgorde te bepalen is de eenvoud (bij z-buffering hebben we geen ingewikkelde datastructuren en preprocessing nodig).

Score: het antwoord van de studenten hoeft niet zo uitgebreid te zijn als hierboven, maar elementen die er beslist in moeten zitten om 0.5 punt te scoren zijn (1) test tegen z-buffer; (2) beslissing om òf zowel pixel als z-buffer te updaten, òf niets te doen. De andere helft van het punt wordt gescoord als de eenvoud van het z-buffer algoritme t.o.v. de alternatieven genoemd wordt.

b) Wanneer we een variant op het scan-line algoritme gebruiken om het z-buffer algoritme te implementeren, in welk(e) coördinatenstelsel(s)/ruimte(n) kunnen we dan werken?

1. In *World Coordinates*
2. In *Camera Coordinates*
3. In het *Canonical Perspective Frame*
4. In *Canonical Projection Space*
5. In alle bovenstaande ruimtes
6. In geen van de bovenstaande ruimtes

Antwoord: Score: Het enige goede antwoord is antwoord 4: in Canonical Projection Space. Eventueel mag dat worden toegelicht, maar het hoeft niet.

Opgave 3. Texture Mapping

a) Bij texture mapping zal het dikwijls voorkomen dat een een pixel wordt gemapt naar een gebied in de texture dat meerdere texels bevat. De kleur van de pixel moet dan worden bepaald aan de hand van de kleur van de texels in dat gebied; dit wordt *filtering* genoemd. Beschrijf twee filtering-methoden die in het bij het college gebruikte boek zijn beschreven.

Antwoord: Volgens het boek: (1) de kleur van de pixel wordt de kleur van de texel die het dichtst bij het middelpunt van de (in de texture map afgebeelde) pixel ligt, en (2) de kleur van de pixel is het gemiddelde van de kleuren van de vier texels die het dichtst bij het middelpunt liggen. Het antwoord wordt ook goed gerekend als er onderscheid gemaakt wordt tussen 1 texel vs. meerdere texels (dus met een ander aantal dan vier, danwel een ongespecificeerd aantal.

Score: om het punt te verdienen moeten minstens twee methoden worden gegeven. Het geven van slechts één methode levert niets op.

b) Wat is *mipmapping*?

Antwoord: In het kort: het bijhouden van textures op verschillende resoluties, en het kiezen van de resolutie aan de hand van de schermgrootte van de afgebeelde polygonen.

Score: het vermelden van het bijhouden van de verschillende resoluties alleen is niet voldoende om het punt te scoren. Er moet vermeld zijn hoe de keuze voor de resolutie wordt gemaakt; dat hoeft slechts een rudimentaire verwijzing te zijn.

c) Stel dat we een cilinder hebben gemodelleerd als een getrianguleerd polyhedron, waarbij alle driehoeken min of meer hetzelfde formaat hebben. We willen het zijvlak (d.w.z.: het gekromde oppervlak) van de cilinder texturen met een tegelpatroon van witte en zwarte vierkantjes. We kunnen daarbij texture mapping uitvoeren voor elke driehoek afzonderlijk, maar dat geeft niet al te fraaie resultaten. Wat is hier het probleem, en hoe kunnen we dit oplossen?

Antwoord: Het probleem is dat de texture niet "smooth" over het polyhedron loopt; de overgangen tussen driehoeken blijven zichtbaar. Dit is op te lossen door de vertices van het polyhedron te projecteren op een (echte, gladde) cylinder die het polyhedron omvat, en waarop we de texture mapping gemakkelijk kunnen berekenen. De texture coordinaten voor de omvattende cylinder gebruiken we dan als texture coordinaten voor de vertices van het polyhedron.

Score: Om het punt te scoren moet het probleem beschreven zijn, alsmede een globale beschrijving van de oplossing.

Opgave 4. Shadows

Eén van de methoden om bij real-time projectietechnieken alsnog schaduwen te introduceren is de zogenaamde *fake-shadow* methode. Daarbij projecteren we, na de scene te hebben gerenderd, alle objecten (of een selectie van belangrijke objecten) op een relevant vlak. Wanneer we bijvoorbeeld schaduwen op een tafel willen hebben, projecteren we op het vlak waarin het tafelblad ligt. Met gebruikmaking van de z-buffer kunnen we de geprojecteerde objecten vervolgens in donkergrijs tekenen, maar dan verliezen we de texture van het onderliggende tafelblad. Fraaier is het om te *blenden*, door bijvoorbeeld de kleurwaarden van de pixels waarop de schaduw-polygonen worden geprojecteerd 50% omlaag te brengen.

a) In het college hebben we twee problemen met deze methode besproken (ze kwamen ook aan de orde in het *stencil buffer tutorial* van NVIDIA). Noem deze twee problemen.

Antwoord: (1) schaduwen kunnen "over de tafelrand" uitsteken (i.h.a.: over het polygon waarop wordt geprojecteerd), en (2) het probleem van "double blending", waarbij de kleurwaarde van een pixel twee of meer keren wordt verlaagd, als er twee of meer schaduwen op vallen.

Score: 0.5 punt voor ieder juist vermelde probleem, waarbij het voor het tweede probleem al voldoende is als de term "double blending" wordt genoemd.

b) Leg uit hoe de problemen waarnaar bij opgave a. wordt gevraagd, kunnen worden opgelost met gebruikmaking van de stencil buffer.

Antwoord: Teken eerst de scene zonder schaduwen, en zet de entry in de stencil buffer die correspondeert met een pixel waarop de tafel (i.h.a.: het polygon waarop we shaduw willen hebben) op een unieke waarde. Voer in een tweede stap de blending uit voor de geprojecteerde schaduw-polygonen, maar alleen voor pixels waarvoor de corresponderende stencilwaarde gelijk is aan de eerder bepaalde unieke waarde (zo zullen schadowen dus niet "uitsteken"). Wanneer je blending hebt uitgevoerd op een pixel, zet je de stencilwaarde op 0 (zo voorkom je double blending.

Score: om het punt te verdienen moet tenminste melding gemaakt worden van (1) het zetten van een unieke stencilwaarde voor de relevante pixels, en (2) het resetten van de stencilwaarde voor behandelde (geblende) pixels.

Opgave 5. Radiosity

a) Vergelijk *radiosity* met *recursive ray tracing*, en ga daarbij in op:

1. lokaal vs. globaal
2. view-dependant vs. view-independant
3. de soorten reflecties waar beide methoden rekening mee houden.

Antwoord: (1) Zowel radiosity als recursive ray tracing zijn globale methoden; (2) radiosity is view-independent, terwijl ray tracing view-dependent is; (3) radiosity beschouwt alleen diffuse reflecties, terwijl ray tracing (a) in de lokale stap met zowel diffuse als speculaire/glossy reflecties rekent, en (b) in de recursie alleen perfect speculaire reflecties beschouwt.

Score: een volledig goed antwoord levert 1 punt op; wanneer (3a) ontbreekt, en gezegd wordt dat ray tracing alleen naar speculaire reflecties kijk, kost dat een half punt. Wanneer het antwoord op een ander onderdeel incorrect is (onafhankelijk van het al dan niet correct zijn van 3a) levert deze vraag niets op.

b) Leg uit wat *progressive refinement* is.

Antwoord: Progressive Refinement is het iteratief benaderen van de radiosities van patches door telkens de invloed van de patch met de hoogste "unshot radiosity" op de overige patches te bepalen, en daarmee de radiosities van die overige patches te updaten. Let op: adaptive subdivision (het steeds fijner verdelen van patches in subpatches) is iets essentieel anders!

Score: om het punt te verdienen is het voldoende wanneer er gesproken wordt van iteratief updaten van de radiosities (niet noodzakelijk met deze woorden).