UNIVERSITETET I AGDER

Emnekode : DAT-200

Kandidatnr. : 268

Dato : 18/12-12

Ark nr. : 1 av 11

A

1

a) Sant

b) Sant

c) Usant

d) Sant

e) Sant

f) Usant

g) Usant

h) Usant

i) Sant

j) Sant

k) Sant

l) Usant

(2)

a) 1. Translasjon a i x-retning, b i y-retning og c i z-retning.

2. Rotasjon θ grader om z-aksen.

3. Skalering i x, y og z-retning med skaleringsfaktor S.

b) Transformasjonsmatriser som er homogene benyttes for å enkelt kunne multiplisere sammen alle matrisene.

(3) ← OBS!

c) Utvidelsen består i å opprettholde en lenket liste for hver pikselposisjon. På denne måten kan flere flater påvirke fargen til en piksel, som er hensiktsmessig ved transparente flater.

(2)

c) 1. Translasjon til origo

2. Rotere slik at speilingslinjen ligger langs x-aksen

3. Speile om x-aksen

4. Invers rotasjon (nr. 2)

5. Translere tilbake til original posisjon.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -3 \\ 0 & 0 & 1 \end{bmatrix}$$

hvor $\theta = \tan^{-1}\left(\frac{1}{2}\right)$

(3)

a) Cyan, magenta og yellow kan brukes for å fremstille sort. Sort er derimot også tatt med i skriveren for å sikre en god gjengivelse av sort og for å spare blekk.

Jeg kan ikke se noen fordeler ved å innføre flere enn de fire nevnte fargene.

③

b) Painter's algorithme, eller dybdesorteringsalgoritmen, sorterer først alle flatene som skal projiseres etter dybde. Deretter skankonverteres flate-for-flate, størst dybde først. Flater som ligger lenger fremme (mindre dybde) vil overskrive flater lenger bak (større dybde).

c) Se side 2.

d)

$x_1 = 1$ $y_1 = 1$

$x_2 = 1$ $y_2 = 1$

```
WHILE (ikke alle piksler prosessert) DO
    i = intensitet(gråskalabilde(x1, y1))
    sorthvitt bilde(x2, y2) = maske(i)
    x1 = x1 + 1
    y1 = y1 + 1
    x2 = x2 + 8
    y2 = y2 + 8
ENDWHILE
```

Forklaring side 11

(4)

a) Algoritmen deler et større "problem" inn i 4 mindre "problem" ved å tilegne et klippevindu 4 klippere. En for over, under, til høyre og til venstre. En til hver, altså.

En liste holder rede på alle punktene som utgjør polygonet, og hvor disse befinner seg i forhold til klippevinduet. Denne listen sendes som input til en klipper, listen oppdateres og sendes til neste klipper.

Hver klipper jobber parvis med punktene. 4 forskjellige betingelser bestemmer om punktene fortsatt skal være med i lista, og evt. skjæringspunkter som skal med.

b) Først klippes det mot venstre side av klippevinduet. Finner skjæringspunktet mellom venstresiden av klippevinduet og linjen mellom punktene P1 og P2, og linjen mellom P1 og P4. P1 erstattes av de to skjæringspunktene og utgjør nå sammen med de andre punktene et polygon der alt til venstre for klippevinduet er klippet. Tilsvarende prosedyre gjøres for de andre sidene i klippevinduet.

(4)

c) Parallell projeksjon er hensiktsmessig ved modellering, slik at avstander og størrelsesforhold er enkle å forholde seg til. Perspektivisk projeksjon ville jeg brukt for å vise frem et resultat på en mest mulig realistisk måte, siden perspektivisk projeksjon etterligner hvordan øyet vårt ser.

d) Dette er tilfelle fordi vektene til kurven ~~vil alltid være ~~ ~~aldri er større enn~~ en. I 2D kan man forestille seg at x- og y-aksen "drar" på kurven. Summen av den vekten x og y "drar" med vil alltid være 1.

Kubiske B-spline er et annet eksempel.

I en linjeklippingsalgoritme kan man først sjekke om omhyllingslegemet er innenfor klippevinduet. Hvis dette er sant kan hele kurven aksepteres. Trivielt.

(4)

e) Den største grunnen for dette er at naturlige kubiske splines ikke har lokal kontroll. Det betyr at hele splinen må prosesseres på nytt ved en ~~liten~~ endring i et kontrollpunkt. Formen på kurven blir også mindre håndterbar av denne grunn.

f) ~~Det~~ Ved å utvide fra 4 til 6 bits per endepunkt. De to nye bitene står for "hear" og "tar". Kan oversettes til foran og bak.

(5)

a) Ville ha modellert en tynn sylinder med en bend-modifier, deretter ~~multiplisert~~ duplisert og rotert disse.

Beholder en rett sylinder i midten. På toppen av inspen er en enhet for å holde de andre delene sammen. For denne ville jeg modellert enda en sylinder, men gjort dette ved å lage en 2D sirkel og deretter bruke en bevel-modifier for å lage de avrundede kantene.

(5)

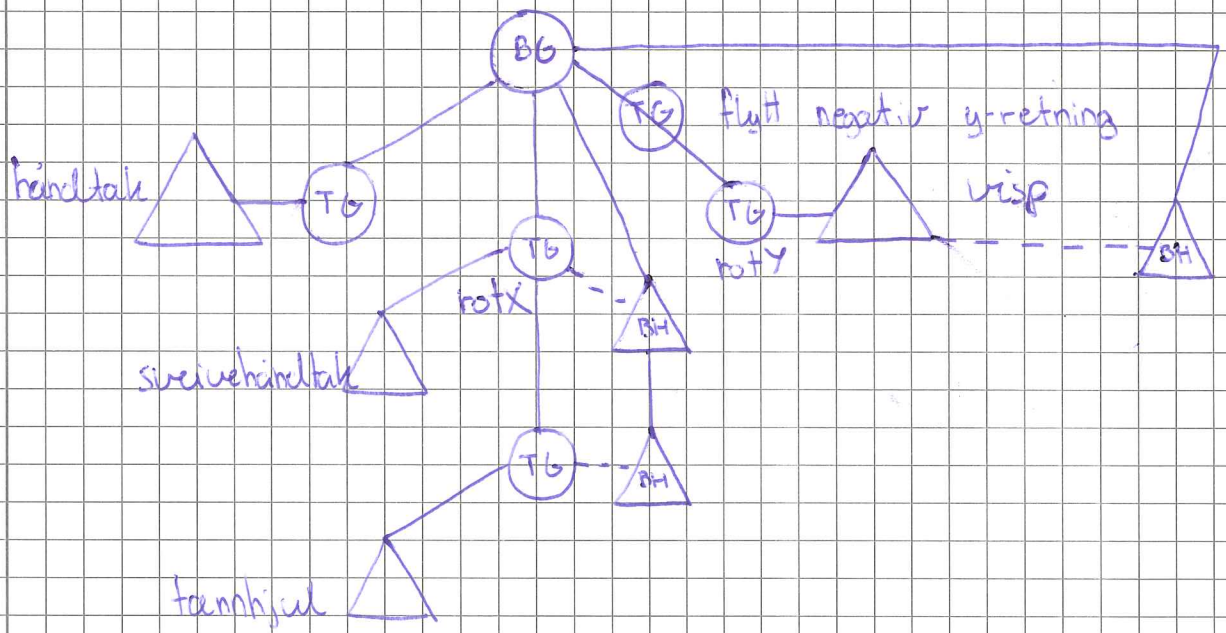
b) Ville lagd en liten bit av tannhjulet og "loftet" denne rundt en sirkel.

c) Antar at håndtaket er laget av gummi for godt grep. Definerer et hvitt, matt materiale (lite spekulærrefleksjon). De to vispene gir jeg et grått materiale med høy spekulær-refleksjon.

6

a) Laster vispen inn i Java3D slik at det første bildet i oppgaven viser vispen i et 2D koordinatsystem med y oppover og x bortover. Sveivehåndtak og tannhjul roterer om x-aksen, mens vispene roterer om y-aksen.

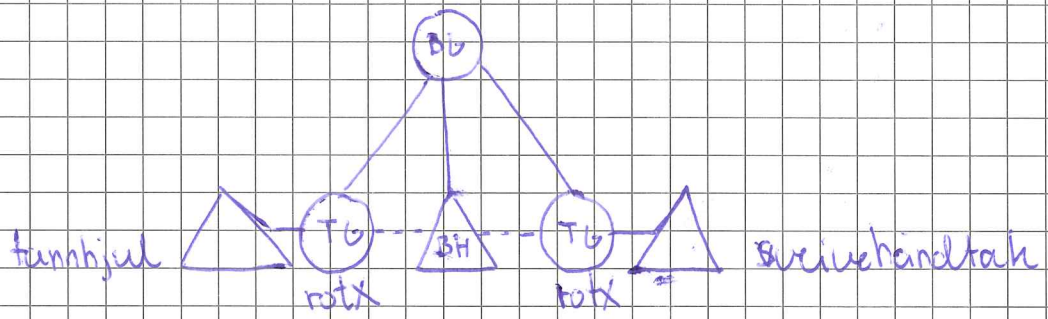
Håndtak, sveiv og tannhjul er modellert med origo i samme punkt.



b)

b) I treet mitt har jeg valgt å la sveivehåndtaket styre rotasjonen til seg selv og tannhjulet. Tannhjulet har dermed ikke mulighet til å rotere uten at sveivehåndtaket roterer. Hvis tannhjulet også kunne rotere ville dette vanskelig ikke påvirket sveivehåndtaket.

Hvis jeg ikke hadde gjort det på denne måten ville jeg definert treet slik som nedenfor og la samme behaviour-objekt påvirke begge TG'ene. - denne delen av



(3)

d) "intensitet" er en funksjon som tar imot en pikselposisjon og returnerer en verdi for intensiteten til denne pikselen.

"maske" er en funksjon som tar imot en verdi for intensitet og returnerer et mønster på 64 piksler hvor antall piksler ~~med~~ farget med sort gjenspeiler intensiteten til den prosesserte pikselen.