UNIVERSITETET I AGDER

Emnekode : DAT200
Kandidatnr. : 426
Dato : 02.12.2015
Ark nr. : 1 av 21

1

a) Ja
b) Nei
c) Ja
d) Ja
e) Ja
f) Ja
g) Nei
h) Nei
i) Ja
j) Nei
k) Ja
l) Ja

2

a)

1 - i
2 - c
3 - h og e
4 - e
5 - h og b
6 - g
7 - c
8) - g

b) For å konvertere til homogene koordinater vil jeg sette $X_h = X$, $Y_h = Y$, $Z_h = Z$, $W_h = 1$ der W_h er den homogene koordinaten

for så å konvertere til 3D koordinater:

$$x = \frac{x_h}{w_h}, \quad y = \frac{y_h}{w_h}, \quad z = \frac{z_h}{w_h}$$

c) Det betyr at matrisen har en 2x2 submatrise som enkelt kan flyttes til et koordinatsystem der objektet blir 1-enhet langs og roteres der.
Dette medfører at vi kan rotere et objekt enkelt uten å måtte flytte det til omtrent av det opprinnelige koordinatsystemet først.

d

Vet ikke om det er en del av oppgaven, men hvis spillet skal fungere lengre enn 1 gang må vi transformere tilbake til origo først før vi utfører rotasjonen og så translasjonen

Kaller t_x og t_y for posisjonen for hendelsen (0 i begynnelsen)

$$\begin{bmatrix} 1 & 0 & x \\ 0 & 1 & y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$$

3 2 1

UNIVERSITETET I AGDER

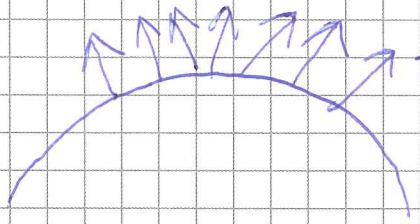
Emnekode : DAT200
Kandidatnr. : 126
Dato : 02.12.2015
Ark nr. : 5 av 21

3
a) Ray casting er en metode for å fjerne skjulte linjer og flater ved å følge en "stråle" fra en piksel og så se hvilken flate som treffes først

Ray tracing er en global lysettingsmodell som bruker ray casting i første del av prosessen, men så følger de reflekterte strålene for å sette lys på scenen.

c) Et CIE chromaticity diagram er et diagram som viser samtlige intensitetsavhengige farger ved hjelp av 3 matematisk beregnete farger med diverse positive tilegg.
Det kan brukes til å vise farge-"gamuts" til forskjellig utstyr eller finne komplementærfarger

c) Diffus refleksjon er refleksjonen over et hele det belyste materialet og går ut i alle retninger



- Diffus refleksjon

mens Spekulær refleksjon er en enkelt vektor reflektert om normalvektoren

- Spekulær refleksjon

Den spekulære refleksjonen blir over et større område og blir mindre sterk når vi har et mattere materiale fordi mindre av den

Spekulære refleksjonen blir reflektert og mere går over til diffus refleksjon.

c) Konstant intensitet:
Normalvektorene blir beregnet for hver flate først å bli brukt direkte i intensitets-likningen.
Gouraud shading:
Gjennomsnitts normalvektorer blir ~~blir~~ beregnet i knutepunktene til flatene før denne blir brukt i intensitetslikningen først så interpoleres denne intensiteten.

Phong shading:
Her vil vi igjen gjennomsnitts normalvektor beregnet og så lineærinterpolert, for den brukes til å regne ut intensiteten.

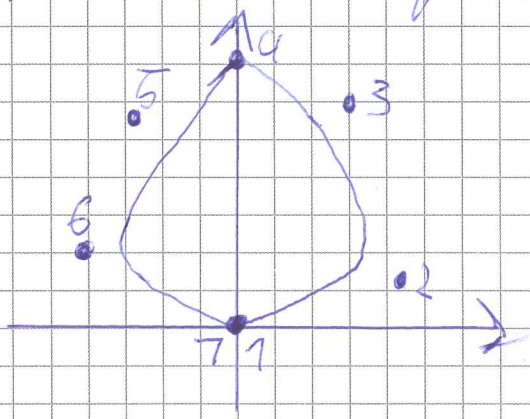
d) Back-face culling fungerer ved at vi fjerner flater på objekter som vender bort fra kamera.
N er flatens normalvektor og V er vårt vektor. Hvis $N \cdot V > 0$ oppfylles kan flaten fjernes.
Algoritmen er rask og fjerner en stor del usynlige flater, men ikke alle

Så vi bruker denne først for vi kjører en tynnere synlighetsbestemmelses-algoritme

4

a Vi starter i origo for så å sette den deriverte ganske bratt opp på siden, neste deriverte ligger høyere, men nærmere y-aksen for så å avslutte kurvesegmentet rett på y-aksen.
Neste segment begynner der forrige sluttet og deriverte går ned til venstre side, neste derivert tok ligger enda brattere til siden for så å

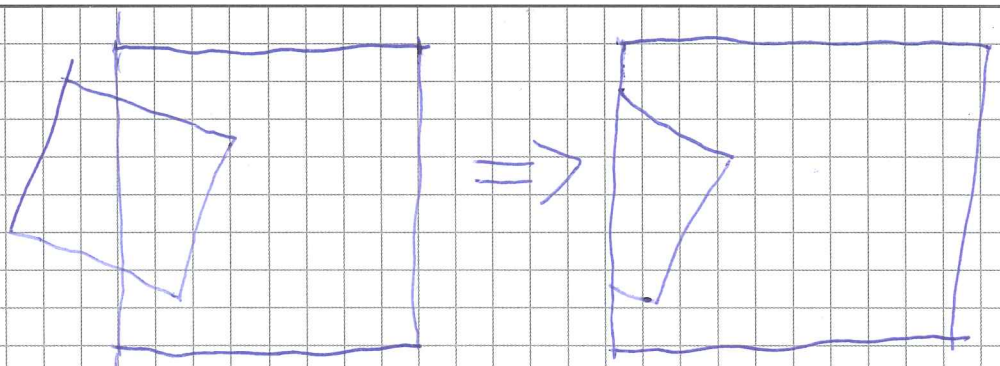
avslutte i punktet i startet (1, totalt 7 punkter



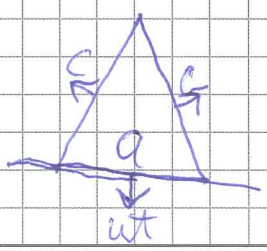
c) Vieuporten er det som til slutt vises på skjermen, etter er klipping og normalisering til vieuport transformeres koordinatene og da får vi de endelige utstyrskoordinatene som kan vises i vieuporten/skjermen.

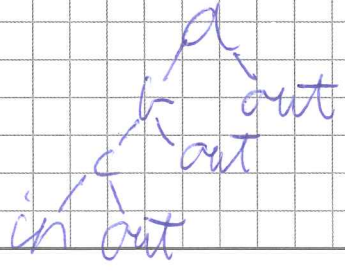
c) p_k er startpunktet til segmentet og Dp_k er dens deriverte. p_{k+1} er sluttpunktet og Dp_{k+1} er dens deriverte; altså stigningen/avtagningen.

d) Sutherland-Hodgman sin polygonklippings algoritme går ut på at vi sprite problemet opp i 4.
Vi forlenger først et av klippevinduets kanter i det uendelige og klipper bort alt som ligger på utsiden så gjør vi det samme med neste kant o.s.v. helt til alle 4 kanter er prosessert.



e) BSP-representasjonen går ut på å legge en vendelig langs kant langs en side i polygonet for så å bestemme en innside og en utside for så og ta neste kant og se på den i forhold til forrige slik at vi lager et binærtre.





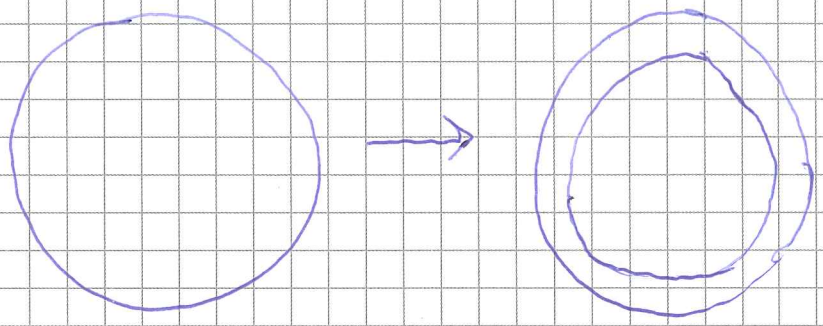
Emnekode : DAT200
Kandidatnr. : 426
Dato : 02.12.2015
Ark nr. : 14 av 21

Dette er ikke den mest brukte metoden for definering av 3D-solider.

f) ~~i parallell så~~ hovedforskjellen er at bilde vil vise forskjellige størrelser på objekter og linjer vil ~~forsvi~~ ikke bevege seg parallellt ~~men~~ men mot et forsvinningspunkt avhengig av ~~hvor~~ hvor du ser objektet fra når du har perspektivisk projeksjon i motsetning til parallell. Perspektivisk er best å bruke i en reklamefilm ettersom det etterligner hvordan øyet vårt ser, altså hvordan slutt-resultatet blir.

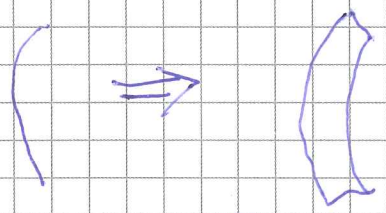
5

a) For å modellere dekk lager jeg først en sirkel spline for så å lofte den for å lage en slags rund tube.



Felgen defineres ved å lage en sylinder, fjerne topp og bunn og gi den en spell-effekt for litt tykkelse.
Så definerer jeg en mindre sylinder som skal være i midten.

Lager den som bruker dem sammen ved å modellere en spline og så lotte inn formen



Setter så sammen alle delene og tar en boolsk operasjon mellom felg og dekk for å fjerne undersiden av dekket siden den ikke skal være rund hele veien rundt

c) Til dekka velger jeg et litt matt materiale, men ikke så veldig for den skal ha litt spekulær refleksjon, det bør også være merket

Til felger velger jeg et lysere og blankere her metallisk materiale.

6
a

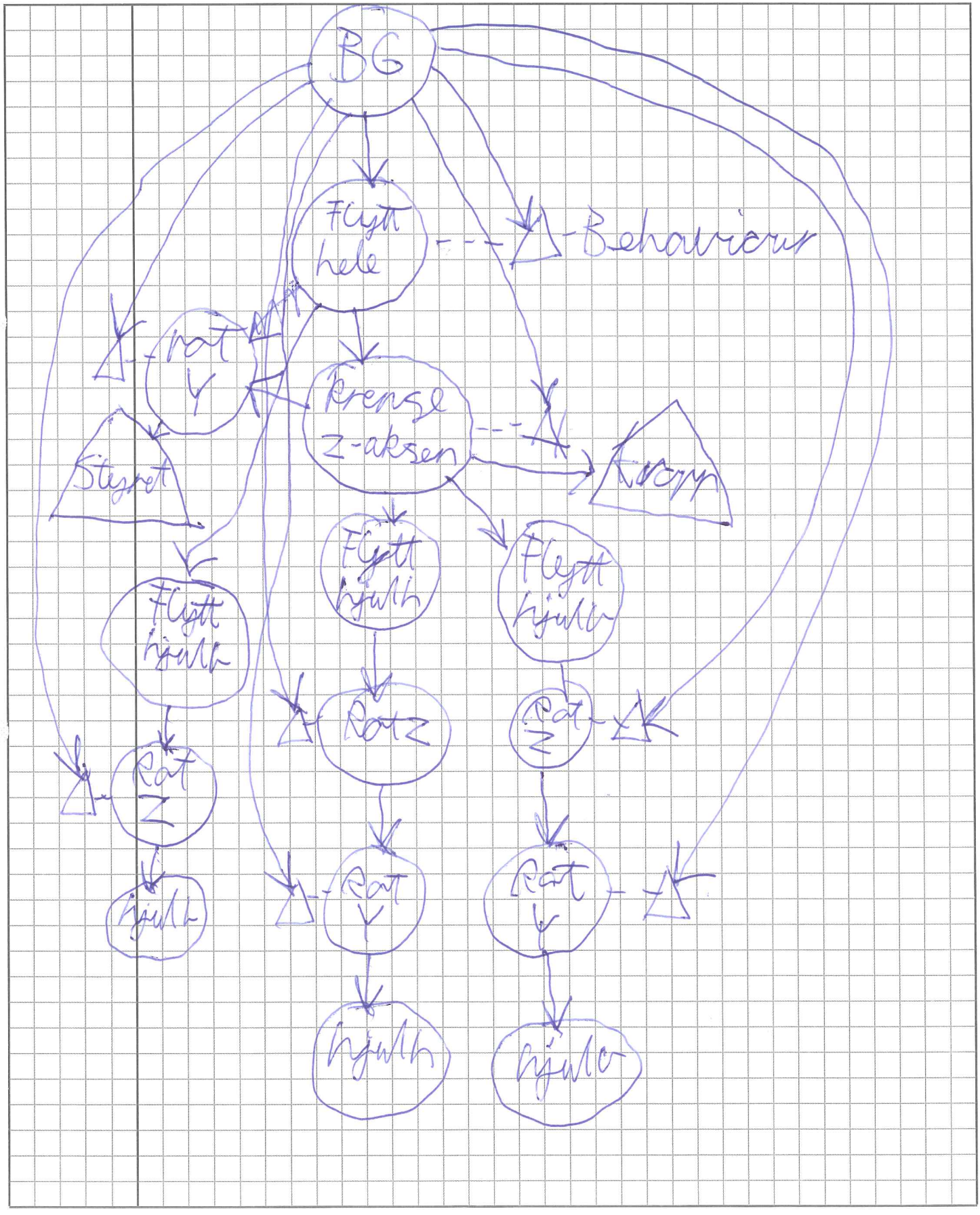
```
public BranchGroup createSceneGraph(){
    BranchGroup objRoot = new BranchGroup();
    Inspector3DS loader = new Inspector3DS("");
    loader.parseIt();
Transformgroup kropp = loader.getModel
// bruker samme metode og laster
inn alle objektene
Transformgroup hjulb = loader.getModel
                ,,          hjultr = 0    ,,
                ,,          hjulth =      ,,
                ,,          stolpet =     ,,
```

```
Transform3D a = new Transform3D();
"    "    b =    "
"    "    c =    "
"    "    d =    "

//Resten med at styret lastes inn i arrays og knappen også ligger under, hjulene lastes ved også inn i arrays så de må flyttes

a.set(new Vector3f(0.0f, -1.0f, 1.0f)
TransformGroup TGhjulV =
new TransformGroup(a);

b.set(new Vector3f(0.0f, -1.0f, -1.0f)
c. ———— " ———— (-1.0f, -1.0f, 0.0f)
//vektorene for å flytte de 3 hjulene.
```

```
TransformGroup TGhjulh = new
  TransformGroup(b)

_____ | | ____ TGhjulb = "
  newtransformGroup(c);
TGhjulh.addChild(hjulfh);
TGhjulv.addChild(hjulfv);
TGhjulb.addChild(hjulb);
objRoot.addChild(stasjon);
  "  "  (kropp);
  "  "  (TGhjulfh);
  "  "  (TGhjulv);
  "  "  (TGhjulb);
return objRoot;
```

UNIVERSITETET I AGDER

Emnekode : DAT200
Kandidatnr. : 426
Dato : 02.12.2015
Ark nr. : 20 av 21



UNIVERSITETET I AGDER

Emnekode : DAT200
Kandidatnr. : 426
Dato : 02.12.2015
Ark nr. : 21 av 21

flutter altså hele fraseset på
inputt, altså kropp, hjul v og
hjul h krenger det samme
disse kan også rotere
rundt z og y ~~for så~~
å i origo for så å
flyttes ut, styret roteres
om y og bakerste hjul
roterer kun om z for det
flyttes (regner med at bakerste
hjul ikke krenger).