Roterende firkant

Ekspert Python

Introduksjon

I denne oppgaven skal du få en enkel introduksjon til hvordan man kan lage 3D-grafikk ved hjelp av python og OpenGL. Målet med denne oppgaven er å lage en kube som roterer, enkelt og greit.

Ting du trenger for å komme i gang

- [ ] pygame
- [ ] OpenGL

Steg 1: Lage et tomt pygame-program

- [ ] Det første vi må gjøre er å importere en del fra diverse kodebiblioteker. Pygame er et bibliotek for å lage spill i python, og OpenGL kan brukes til å tegne grafikk både i 2D og 3D.

```
import pygame
from OpenGL.GL import *
from OpenGL.GLU import *
from pygame.locals import *
```

- [ ] Når vi har importert alt vi trenger så er det på tide å opprette et vindu som vi kan tegne i. Før vi kan bruke pygame så må vi kalle på funksjonen som heter pygame.init().

- [ ] Den neste linjen kaller på funksjonen set_mode() som vi bruker for å sette opp et vindu som vi kan tegne i. Den første parameteren til set_mode sier hvor mange pixler vi skal ha i bredden og høyden på vinduet vårt. Siden vi skal bruke denne informasjonen senere i programmet så lagrer vi den i en variabel som heter display. I eksempelet er vinduet vårt 800 pixler bredt og 600 pixler høyt.

```
pygame.init()
display = (800,600)
pygame.display.set_mode(display, OPENGL | DOUBLEBUF)
```

- [ ] Hvis du kjører programmet ditt nå så vil du trolig få opp et vindu på skjermen som umiddelbart lukkes, og programmet ditt vil avslutte. Det vi trenger nå er noe som sørger for at programmet vårt ikke avslutter før vi vil at det skal avslutte.

```
while True:
  for event in pygame.event.get():
      if event.type==pygame.QUIT:
          pygame.quit()
          quit()

  pygame.display.flip()
  pygame.time.wait(10)
```

Dette er en while-løkke som kjører helt til noen krysser ut vinduet. Det er inne i denne løkken at vi kommer til å legge koden som tegner og roterer kuben vår.

Steg 2: Tegne en firkant

Til nå har vi bare et tomt vindu som ikke viser noe fornuftig. Nå er det på tide å faktisk tegne en firkant. Det første du må gjøre er å legge til tre linjer rett over while-loopen din.

```
gluPerspective(45, display[0]/display[1], 0.1, 50)
glTranslatef(0, 0, -5)
glColor3fv((0, 127, 127))
```

Det er ikke lett å registrere at disse linjene gjør noe spesielt før vi faktisk begynner å tegne noe på skjermen. I korte trekk så beskriver vi hvordan vi skal vise de tre dimensjonene på dataskjermen, flytter kameraposisjonen vår, og til slutt sier hvilken farge vi skal tegne med.

- [ ] Gå inn i while-loopen og legg til de følgende linjene.

```
glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
glBegin(GL_QUADS)
glEnd()
```

Fikk du tidligere en del rare ting tegnet i vinduet ditt når du kjørte det? Dette slipper vi når vi kaller glClear slik som det er vist over. Denne linjen sier enkelt og greit i fra om at vi skal ha en tom skjerm, det eneste som vil bli vist er det vi sier at skal tegnes i etterkant.

OpenGL er biblioteket som vi bruker for å tegne 3D-grafikk. OpenGL fungerer stort sett slik at man oppgir en rekke punkter og så vil OpenGL tegne mellom disse punktene på en gitt måte. Når vi kaller glBegin(GL_QUADS) så vil OpenGL ta hver gruppe på fire punkter og tegne en flate mellom dem. glEnd sier ifra om at vi er ferdig med å tegne det vi skulle med GL_QUADS, og vi kan eventuelt tegne noe annet på en annen måte.

For å sende inn et punkt som OpenGL skal bruke til å tegne med så bruker vi funksjonen glVertex3fv. Denne funksjonen tar inn et punkt, siden vi jobber i 3D så vil det bety at vi må ha tre tallverdier, en for x-aksen, y-aksen, og z-aksen.

- [ ] Det vi skal tegne er en firkant, derfor trenger vi å oppgi fire punkter. Legg følgende mellom glBegin og glEnd.

```
glVertex3fv((-1,-1, 0))
glVertex3fv((-1, 1, 0))
glVertex3fv(( 1, 1, 0))
glVertex3fv(( 1,-1, 0))
```

Hvis alt har gått rett for seg så skal du nå ha en nydelig og turkis firkant på skjermen din. Dessverre er det lite som avslører at dette er 3D og ikke bare 2D. Det fikser vi enkelt og greit ved å legge til litt rotasjon. Legg til den følgende linjen inne i while-løkken din.

```
glRotatef(1, 1, 0, 0)
```