**Cho điểm P(x, y, z) trong hệ tọa độ thế giới thực (world space)**

**1) Trình bày ma trận biến đổi và xác định điểm Pw(xw, yw, zw, 1) khi thực hiện các phép biến đổi sau:**

**a. Phép tịnh tiến với dx, dy, dz**

**b. Phép quay quanh trục Ox, Oy, Oz**

**c. Phép quay quanh một trục PQ có P(xp, yp, zp), Q(xq, yq, zq)**

Giải

a)

Mmodel­=[Mt ] =

Pworld =Mmodel \* P = \*

Code:

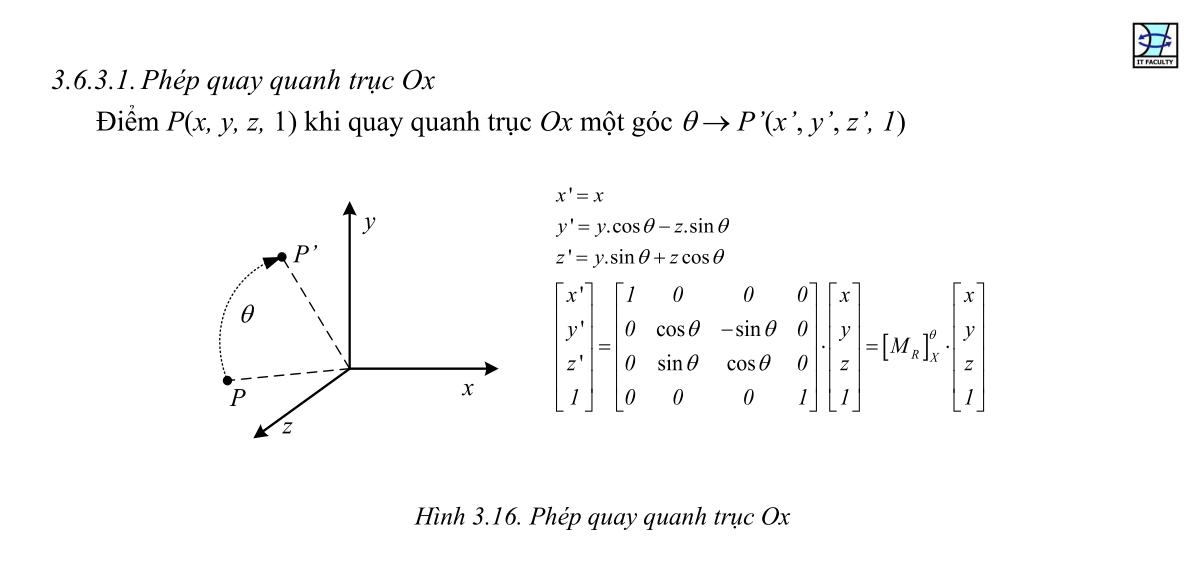
glMatrixModel(GL\_MODELVIEW);

glLoadIdentity()

glTranslatef(dx,dy,dz);

b) Phép quay quanh trục Ox, Oy, Oz

-Trục Ox:



Mmodel­=[Mt ] =

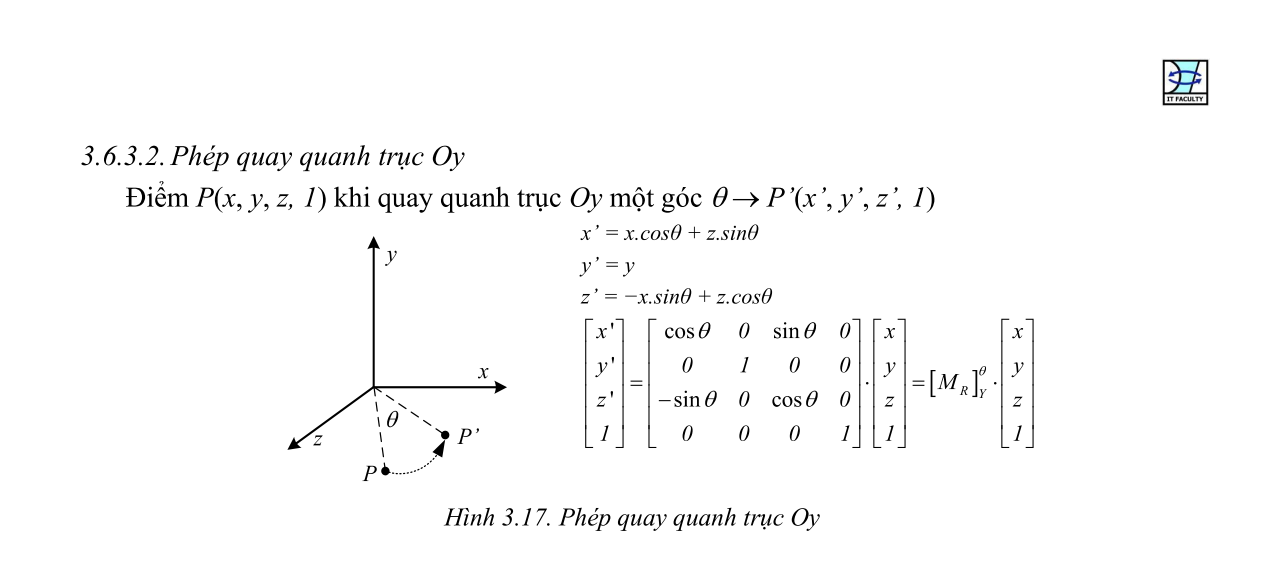
PW=MModel\*P=

CODE:

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glRotatef(theta,1.0,0.0,0.0);

* Trục Oy:
* 

Mmodel­=[Mt ] =

PW=MModel\*P=

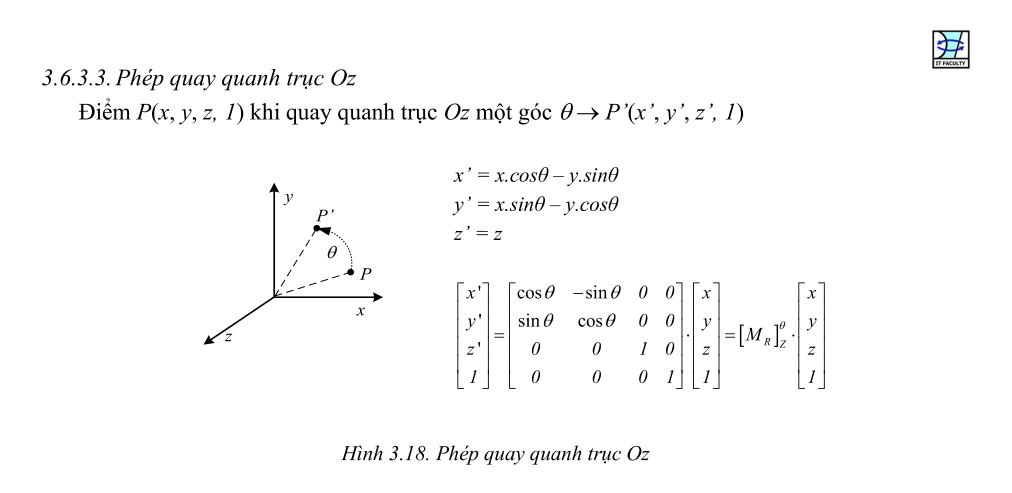
CODE:

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glRotatef(theta,0.0,1.0,0.0);

* Trục Oz



Mmodel­=[Mt ] =

PW=MModel\*P=

CODE:

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glRotatef(theta,0.0,0.0,1.0);

c) Phép quay quanh một trục PQ có P(xp, yp, zp), Q(xq, yq, zq):

Ta có: vecto (xQ-xP, yQ-yP, zQ-zP)

Bước 1: Tịnh tiến P O:

[MT] =

Bước 2: Quay PQ quanh trục Ox góc x để PQ nằm trên mặt xz:

[MR]X===

Bước 3: Quay PQ quanh trục Oy một góc -để PQ nằm trên trục Oz

[MR]Y= =

Bước 4: Quay quanh trục Oz một góc

[MR]Z=

Bước 5: Quay lại PQ ngược lại quanh trục Oy một góc y

[MR]Y= =

Bước 6: Quay PQ ngược lại quanh trục Ox một góc -

[MR]X= =

Bước 7: Tịnh tiến PQ về lại vị trí ban đầu

[MT]=



CODE:

glModeMatrix(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(-xp, -yp, -zp);

glRotatef(theta, xq-xp, yq-yp,zq-zp);

glTranslatef(xp,yp,zp);

**2) Trình bày ma trận biến đổi và xác định điểm Pc(xc, yc, zc, 1) trong hệ tọa độ camera space khi thực hiện:**

**Phép biến đổi camera đặt tại t = (x1, y1, z1), nhìn vào c = (x2, y2, z2), hướng lên k = (0, 1, 0)]**

Giải

Hướng nhìn của camera là d = c - t = (xd, yd, zd)

Ta có:

z = -

x =

y = z \* x

M =

Mworld -> view =

Pc = Mworld -> view  \* Pworld =

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

gluLookAt(x1,y1,z1,x2,y2,z2,0,1,0);

Ví dụ mẫu:

Input: *Pworld(22,4,102,1)*

Cho camera đặt tại điểm quan sát *e(xe, ye, ze)=(4, 4, 4)*, nhìn vào tâm *c(xc, yc, zc)=(0, 1, 4)*, hướng lên *up: u(xu, yu, zu)= (0, 1, 0).*

Chọn vị trí camera: *e = (4,4,4)* nhìn vào *c = (0,1,4)*

Hướng nhìn của camera là d = c – e = (0,1,4) – (4,4,4) = (-4, -3, 0).

Vậy ta có các vecto e=(4,4,4), c=(0,1,4),d = (-4,-3,0), PWorld=(22,4,102,1),u=(0,1,0)

Z=- = (4/5,3/5,0)

X==

Y== x = (-3/5, 4/5, 0)

M =

MT=

-MT\*e=-\* =

Mworld -> view=

Pcam= Mworld->view\* Pworld  =

CODE:

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

gluLookAt(4.0f, 4.0f, 4.0f, 0.0f, 1.0f, 4.0f, 0.0f, 1.0f, 0.0f);

**3) Trình bày ma trận biến đổi và xác định điểm Pclip(xclip, yclip, zclip, wclip) trong clipping/projection space khi thực hiện:**

**a) Phép chiếu song song**

**b) Phép chiếu xiên**

**c) Phép chiếu trực giao**

**d) Phép chiếu phối cảnh**

Giải

1. **Phép chiếu song song**

[MTILT]=

Trong đó, w,d h là các thông số liên quan đến cửa số nhìn và tọa độ của mặt phẳng chiếu, và (x,y,z) là tọa độ của điểm quan sát. Các thông số này cần được thiết lập trước khi thực hiện phép chiếu.

1. **Phép chiếu xiên**

[MISO] = [MTILT].

1. **Phép chiếu trực giao**

* Ma trận phép chiếu trên các mặt phẳng Oxy (Z=0)

[MZ]=

* Ma trận phép chiếu trên các mặt phẳng Oxz (Y=0)

[MY]=

* Ma trận phép chiếu trên các mặt phẳng Oyz (X=0)

[MX]=

1. **Phép chiếu phối cảnh**

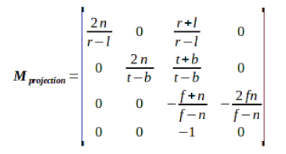
Mprojection=

Ví dụ mẫu:

Trình bày công thức tính toán, ma trận biến đổi Mproj, tính tọa độ điểm ảnh Pproj sau khi thực hiện phép chiếu phối cảnh có các thông số: b=3.0; t=-3.0; l=-4.0; r=4.0; near=0.1; far=80.

Ta có Pcam=

Ta có ma trận phép chiếu bối cảnh tổng quát:



=

làm tròn =

*Ppro=-\*Mpro\*Pcam=***(49/288,-1/40,73/72,1)**

*CODE*

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluLookAt( x,y,z ); //set up góc nhìn camera

glFrustum(l,r,b,t,n,f);