

# BÀI GIẢNG ĐỒ HỌA MÁY TÍNH

Lưu hành nội bộ  
Đà Nẵng, 2025

## Chương 2. BIẾN ĐỔI ĐỒ HỌA MÁY TÍNH

### 2.1 Quá trình hiển thị đồ họa 3D

- Các đối tượng trong thế giới thực phần lớn là các đối tượng 3D
- Thiết bị đồ họa hiển thị chỉ 2D.
- Do vậy, muốn có hình ảnh 3D ta cần phải giả lập trên thiết bị.
- Thực hiện chuyển đổi từng bước => Hình ảnh sẽ được hình thành từ từ, chi tiết hơn.

### Các bước chính trong quá trình hiển thị đồ họa 3D

Modeling Transformation: Biến đổi mô hình Các phép ánh xạ tọa độ điểm hay vector thành tọa độ hay vector khác

Viewing Transformation: Biến đổi góc nhìn

Hidden Surfaces: Khử mặt khuất

Lighting & Shadow: Chiếu sáng và tô bóng đối tượng.

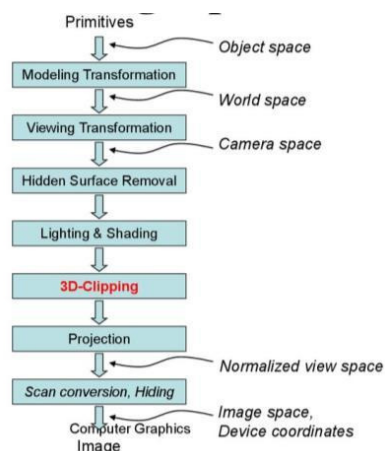
3D Clipping: Xén đối tượng

Projection Transformation: Biến đổi hình chiếu

Biến đổi khung nhìn

Rasterization: Chuyển sang dạng pixel.

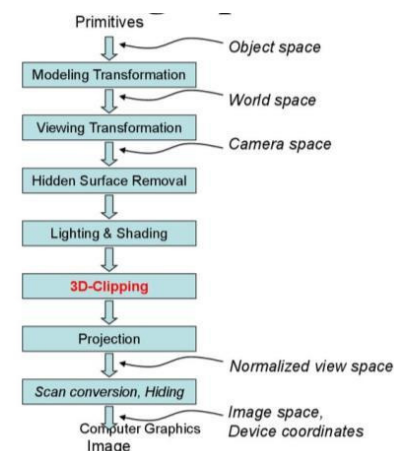
Display: Hiển thị hình ảnh trên màn hình 2D



### Các bước chính trong quá trình hiển thị đồ họa 3D

#### 1) Modelling transformation:

- Mỗi đối tượng được mô tả trong một hệ tọa độ riêng được gọi là hệ tọa độ đối tượng/Hệ tọa độ cục bộ.
- Thực hiện biến đổi từ hệ tọa độ đối tượng sang hệ tọa độ thế giới thực.
- Khi biểu diễn đối tượng, ta có thể chọn góc tọa độ và đơn vị đo lường sao cho việc biểu diễn là thuận lợi nhất.
- Thường chuẩn hóa kích thước của đối tượng khi biểu diễn.

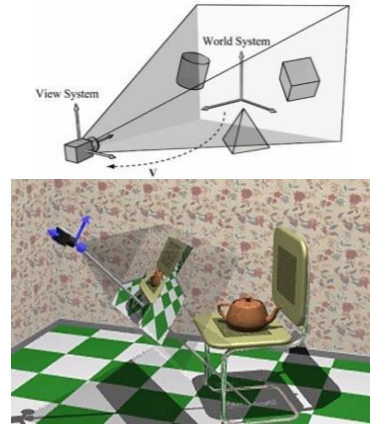




## Các bước chính trong quá trình hiển thị đồ họa 3D

### 2) Viewing Transformation

- Chuyển từ không gian thế giới thực (world space) sang không gian quan sát (eye/camera space).
- Thực hiện một phép biến đổi hệ tọa độ để đặt vị trí quan sát (viewing position) về gốc tọa độ và mặt phẳng quan sát (viewing plane) về một vị trí mong muốn.
- Hình ảnh hiển thị phụ thuộc vào vị trí quan sát và góc nhìn.
- Hệ qui chiếu có gốc đặt tại vị trí quan sát và phù hợp với hướng nhìn sẽ thuận tiện cho quá trình xử lý và hiển thị.



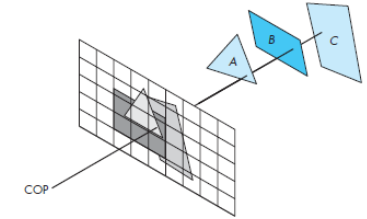
4



## Các bước chính trong quá trình hiển thị đồ họa 3D

### 3) Hidden Surface Removal:

- Loại bỏ các đối tượng không nhìn thấy được (Trivial Rejection)
- Giảm chi phí xử lý.



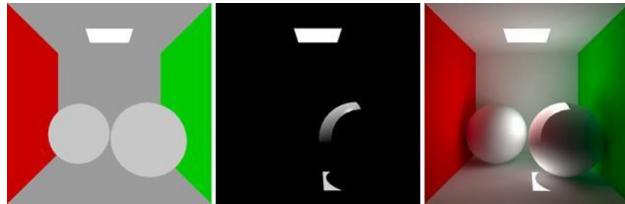
5



## Các bước chính trong quá trình hiển thị đồ họa 3D

### 4) Lighting & Shading:

- Chiếu sáng các đối tượng (Illumination).
- Gán cho các đối tượng màu sắc dựa trên các đặc tính của các chất liệu và các nguồn sáng.
- Các mô hình chiếu sáng và tạo bóng : constant-intensity, Interpolate,...

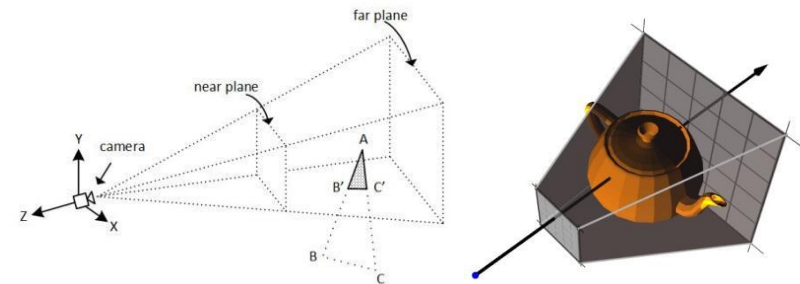


6



### 5) 3D Clipping: Loại bỏ phần nằm ngoài viewing frustum.

- Thực hiện việc xén đối tượng trong cảnh để cảnh nằm gọn trong một phần không gian hình chóp cụt giới hạn vùng quan sát (viewing frustum).
- Vùng quan sát có trục trùng với tia nhìn, kích thước giới hạn tùy theo đối tượng quan sát.



7



## Các bước chính trong quá trình hiển thị đồ họa 3D

### 6) Projection transformation

- Chiếu từ không gian quan sát (*eye space*) xuống không gian màn hình 2D (*screen space*).
- Phân loại: Chiếu song song, Chiếu phối cảnh
- Thực hiện chiếu đồng thời **khử mặt khuất** để có thể nhận được hình ảnh trung thực.

### 7) Chuyển đổi tượng sang dạng pixel (Rasterization)

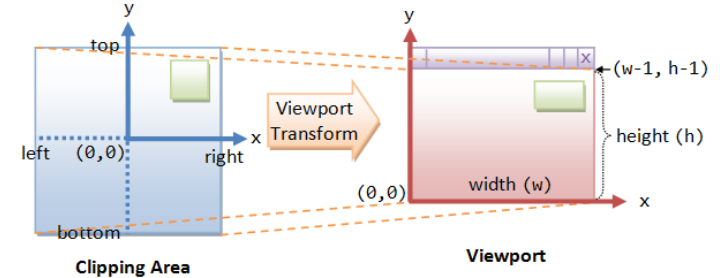


8

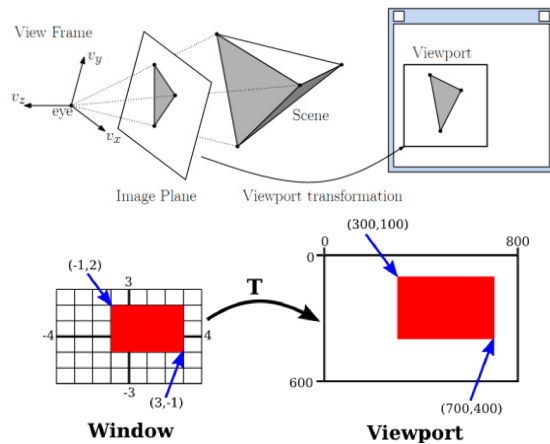


## Các bước chính trong quá trình hiển thị đồ họa 3D

### 8) Hiển thị đối tượng (Display)



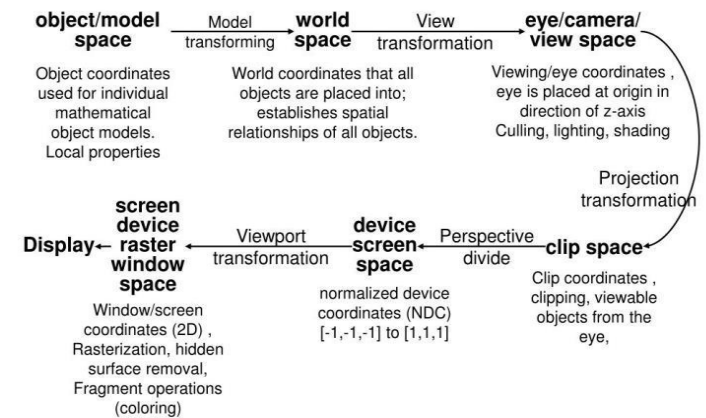
9



10



### Các bước biến đổi mô hình



11

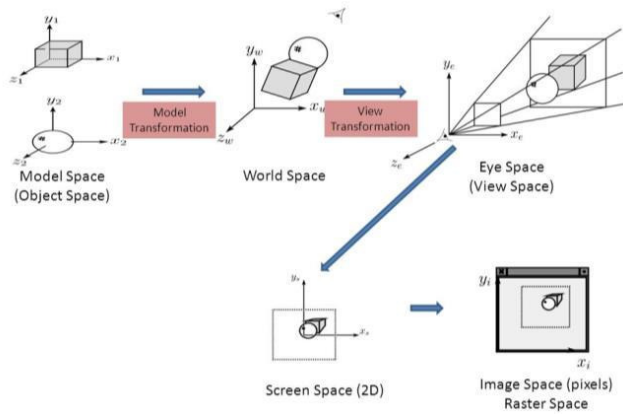


## 2.2 Biến đổi hệ tọa độ

### 1) Hệ tọa độ đối tượng/mô hình (object/model coordinates):

Khi thiết kế đối tượng, thường gốc tọa độ đặt tại tâm của đối tượng.

Các tọa độ được biến đổi sang không gian toàn cảnh với biến đổi mô hình (modeling transformation, world transformation) với ma trận  $M_{\text{Object} \rightarrow \text{world}}$



12



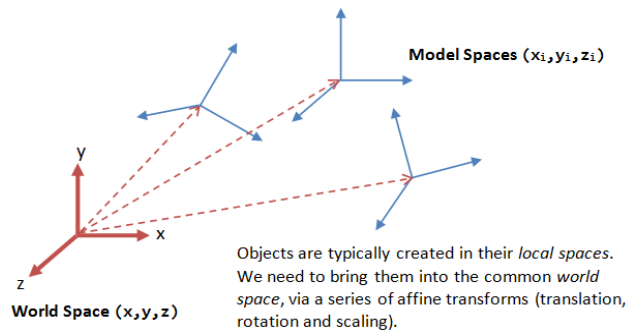
Hình 2.1. Quá trình đo tọa độ thể giới thực

13



### 2) Hệ tọa độ thể giới thực (world coordinates):

- Các đối tượng được đặt vào vị trí trong cảnh do người lập trình quy định.
- Hình ảnh phải được nhìn thấy bởi người xem dưới góc độ quan sát của họ.
- Thực hiện biến đổi view hay camera (viewing/camera transformation), với ma trận  $M_{\text{world} \rightarrow \text{view}}$



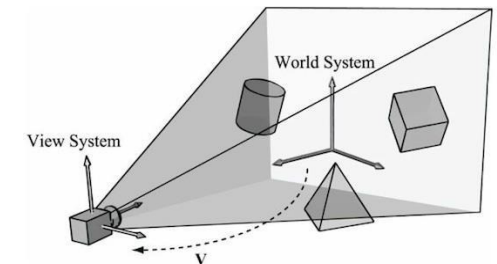
14



### 3) Hệ tọa độ quan sát (view/eye/camera coordinates):

- Gốc tọa độ đặt tại mắt người quan sát
- Một hướng ngắm (nhìn vào tâm điểm)
- Một hướng lên trên (up)
- Một hướng khởi tạo để qui định mắt người quan sát hướng về phía nào trong không gian toàn cảnh).

Đối tượng 3D được hiển thị trên màn hình 2D. Cần biến đổi phép chiếu với ma trận  $M_{\text{proj}}$

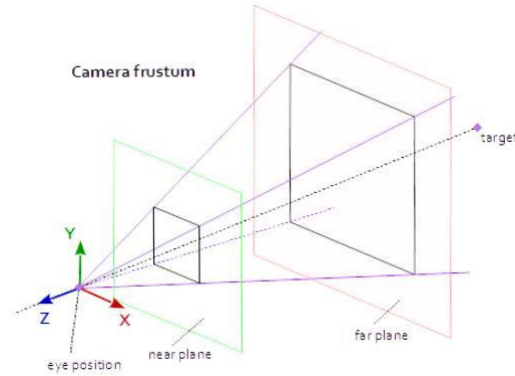


15



### 5) Hệ tọa độ clip (clip coordinates):

- Biến đổi phép chiếu thực hiện cắt và giới hạn cảnh xem vào một hình chóp cắt.
- Góc nhìn theo chiều thẳng đứng (fovy - field of view in y direction),
- Tỷ lệ màn hình (hay cửa sổ chương trình) xác định góc nhìn ngang, mặt phẳng clip gần và mặt phẳng clip xa.

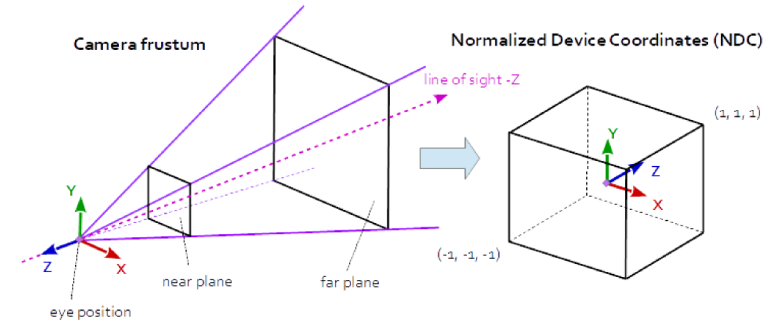


16



### 6) Hệ tọa độ thiết bị chuẩn hóa (Normalized Device Coordinates - NDC):

Các tọa độ x, y, z được ánh xạ trong khoảng [-1,1], gọi là tọa độ thiết bị chuẩn hóa.



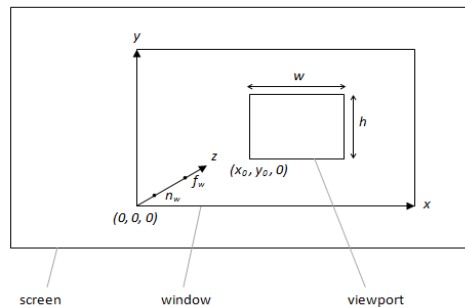
17



### 7) Hệ tọa độ màn hình/cửa sổ (screen/window coordinates):

- Tọa độ cuối cùng là tọa độ màn hình hay cửa sổ, với biến đổi cuối cùng là biến đổi khung nhìn (viewport transformation) => Ma trận biến đổi:

3D Model space => World Space  
World Space => Camera space  
Camera space => Projection space



18



## Chương 3. CÁC PHÉP BIẾN ĐỔI TRONG ĐỒ HỌA 3D

### 3.1 Tọa độ đồng nhất (Homogeneous)

- Cho phép các phép biến đổi Affine có thể được biểu diễn dễ dàng bằng một ma trận.
- Giúp cho việc tính toán có thể thực hiện trong không gian xạ ảnh, giống như là hệ tọa độ Descartes trong không gian Euclide

- Tọa độ thông thường → Tọa độ đồng nhất

$$P(x, y, z) \rightarrow P(x*w, y*w, z*w, w)$$

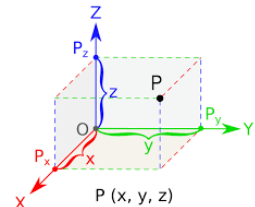
$$P(x, y, z) \rightarrow P(x, y, z, 1)$$

- Tọa độ đồng nhất → Tọa độ thông thường: chia cho thành phần tọa độ thứ 4 và bỏ đi thành phần tọa độ thứ 4)

$$P(x, y, z, w) \rightarrow P(x/w, y/w, z/w, 1) \rightarrow P(x, y, z)$$

$$P(x, y, z, 1) \rightarrow P(x, y, z)$$

Ví dụ:  $P(3, 6, 2, 3) \rightarrow$  tọa độ thông thường là  $(1, 2, 2/3)$

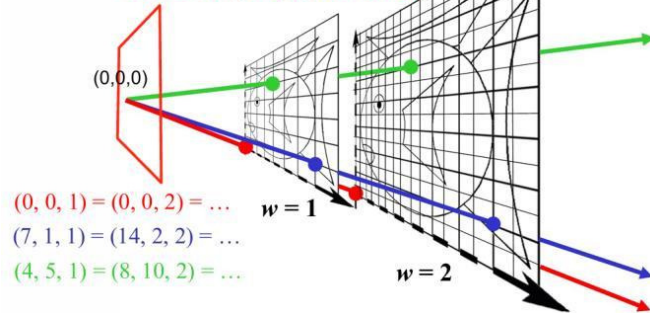


19



- Divide by  $w$  to normalize (project)

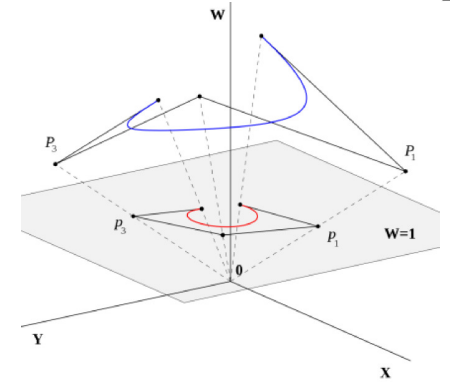
- $w = 0$ ? *Points at infinity (directions)*



20

### Ý nghĩa tọa độ đồng nhất:

- Trong phép biến đổi hình học Affine, tọa độ điểm được mô tả dưới ma trận  $\begin{bmatrix} x^* & y^* & h \end{bmatrix}$ . Trong đó  $x = x^*/h$ ,  $y = y^*/h$ ,  $z = z^*/h$  và  $h$  là một số thực tùy ý
- Cho phép biểu diễn hợp nhất của các phép biến đổi dưới phép nhân ma trận,
- Hỗ trợ cho việc xử lý bằng cả phần cứng và phần mềm: kết hợp với cả các phép biến đổi đặc biệt không tuyến tính khác (non-affine) phép chiếu phối cảnh (Perspective projections), bends, tapers...
- Biểu diễn đồng nhất  $P = (wP_x, wP_y, wP_z, w)$ , các điểm nằm trên cùng tia, điểm ở vô cùng cùng  $w=0$ .

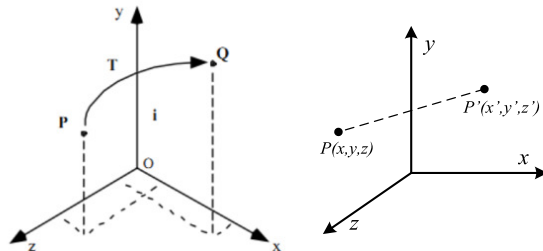


21



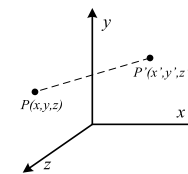
## 3.2 Các phép biến đổi đồ họa 3D (MODEL TRANSFORMATION)

### 3.2.1 Phép tịnh tiến (Translate)



- Biến đổi điểm  $P_{obj}(x, y, z, 1) \rightarrow P'_{trans}(x', y', z', 1)$  thông qua vector tịnh tiến  $T = (t_x, t_y, t_z)$

22



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_T] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$P_{world} = [M_T] * P_{obj}$$

Hình 3.1. Phép tịnh tiến 3D

Vector  $T(t_x, t_y, t_z)$ : biểu diễn giá trị tịnh tiến theo các hướng trục  $Ox, Oy, Oz$ .

23





Ví dụ:

$$\begin{array}{|c|} \hline P_{world} \\ \hline 4.00 \\ \hline 6.00 \\ \hline 8.00 \\ \hline 1.00 \\ \hline \end{array} = \begin{array}{|c|c|c|c|} \hline [T] = & (1.00 & 2.00 & 3.00) \\ \hline 1.0 & 0.0 & 0.0 & 1.0 \\ \hline 0.0 & 1.0 & 0.0 & 2.0 \\ \hline 0.0 & 0.0 & 1.0 & 3.0 \\ \hline 0.0 & 0.0 & 0.0 & 1.0 \\ \hline \end{array} \times \begin{array}{|c|} \hline P_{obj} \\ \hline 3.00 \\ \hline 4.00 \\ \hline 5.00 \\ \hline 1.00 \\ \hline \end{array}$$

- Cho  $P_{world}$ , biến đổi ngược thành  $P_{obj}$ ?

$$P_{obj} = [M_T]^{-1} * P_{world}$$

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -t_x & -t_y & -t_z & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_T]^{-1} \cdot \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

24



### 3.2.2 Phép biến đổi tỉ lệ (Scale)

#### 3.2.2.1 Phép biến đổi tỉ lệ một điểm

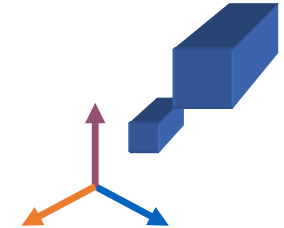
- Cho điểm  $P_{obj}(x, y, z, 1) \rightarrow P'_{scale}(x', y', z', 1)$  thông qua  $S(s_x, s_y, s_z)$ :

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_S] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$P_{world} = [M_S] * P_{obj}$$

Nếu  $s_x \neq s_y \neq s_z$ : biến đổi méo.

Nếu  $s_x = s_y = s_z$ : biến đổi đồng dạng



25



Dùng phép tỉ lệ có ma trận biến đổi đồng dạng như sau:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & s \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ s \end{bmatrix}$$

Biến đổi ma trận để cho các tọa độ (x, y, z) thành các tọa độ Descartes chuẩn.

Nếu  $s > 1$  thì sẽ giảm kích thước đối tượng và ngược lại.

- Cho  $P_{world}$ , biến đổi ngược thành  $P_{obj}$ ?

$$P_{obj} = [M_S]^{-1} \cdot P_{world}$$

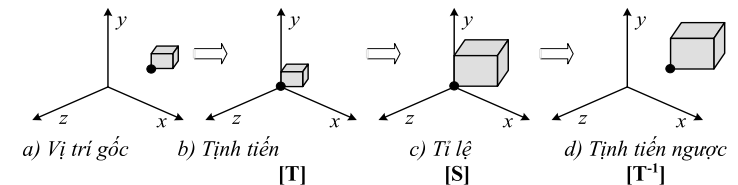
$$[M_S]^{-1} = \begin{bmatrix} 1/s_x & 0 & 0 & 0 \\ 0 & 1/s_y & 0 & 0 \\ 0 & 0 & 1/s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

26



#### 3.2.2.2 Phép biến đổi tỉ lệ một đối tượng

- Tịnh tiến đối tượng sao cho điểm xác định trùng với điểm gốc.
- Biến đổi tỉ lệ đối tượng theo tọa độ gốc;
- Dịch chuyển ngược đối tượng về vị trí cũ.



Hình 3.2. Phép biến đổi tỉ lệ tại một điểm trên đối tượng

$$P_{world} = [T^{-1}][S][T] * P_{obj}$$

$$P_{world} = [M] * P_{obj}$$

$$[M] = [T^{-1}][S][T]$$

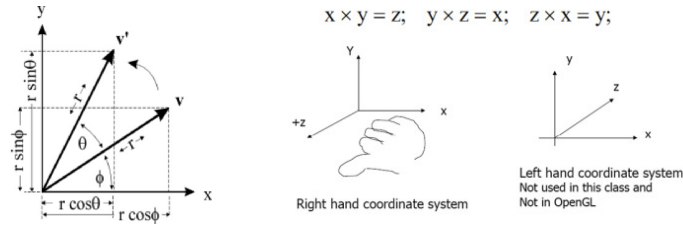
27



### 3.2.3 Phép quay (Rotate)

- Phép quay làm thay đổi hướng của đối tượng. Một phép quay cần có tâm quay, góc quay.
- Góc  $\theta > 0$ : quay theo chiều dương (chiều ngược chiều kim đồng hồ).  $\theta < 0$ : clockwise
- Xét phép **quay dương** (ngược chiều kim đồng hồ) trong hệ trục tọa độ  $Oxyz$  theo quy tắc **bàn tay phải**.

- Quay quanh các trục  $Ox$ ,  $Oy$ ,  $Oz$ ;
- Quay quanh vectơ  $u$
- Quay quanh một trục  $P_0P_1$  bất kì

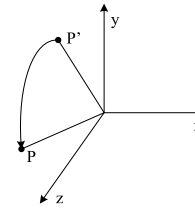


28



### 3.2.3.1 Phép quay quanh trục $Ox$

- Điểm  $P(x, y, z, 1)$  khi quay quanh trục  $Ox$  một góc  $\theta \rightarrow P'(x', y', z', 1)$
- Góc quay dương ngược chiều kim đồng hồ.



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [R_x] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$P_{\text{world}} = [M_{R_x}] * P_{\text{obj}}$$

$$x' = x$$

$$y' = y \cdot \cos \theta - z \cdot \sin \theta$$

$$z' = y \cdot \sin \theta + z \cdot \cos \theta$$

Hình 3.3. Phép quay quanh trục  $Ox$

29



Ví dụ:

Pworld		[Rx]	$\alpha =$	45.00°	0.79rad		Pobj
4.00	=	1.00	0.00	0.00	0.0	x	4.00
-1.41		0.00	0.71	-0.71	0.0		6.00
9.90		0.00	0.71	0.71	0.0		8.00
1.00		0.00	0.00	0.0	1.0		1.00

30



- Cho  $P_{\text{world}}$ , biến đổi ngược thành  $P_{\text{obj}}$ ?

$$P_{\text{obj}} = [M_{R_x}]^{-1} * P_{\text{world}}$$

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-\theta) & -\sin(-\theta) & 0 \\ 0 & \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_{R_x}]^{-1} \cdot \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -\cos(\theta) & \sin(\theta) & 0 \\ 0 & -\sin(\theta) & -\cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_{R_x}]^{-1} \cdot \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

Ma trận quay là trực giao  $\Rightarrow$  ma trận nghịch đảo của ma trận quay là ma trận chuyển vị

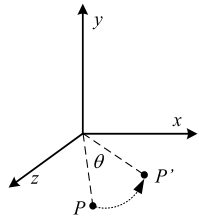
31





### 3.2.3.2 Phép quay quanh trục Oy

- Điểm  $P(x, y, z, 1)$  khi quay quanh trục  $Oy$  một góc  $\theta \rightarrow P'(x', y', z', 1)$
- Góc quay dương ngược chiều kim đồng hồ (counterclockwise)



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_{Ry}] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

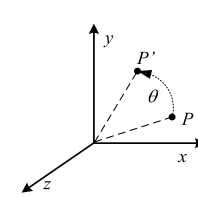
$\mathbf{P}_{\text{world}} = [\mathbf{M}_{Ry}] * \mathbf{P}_{\text{obj}}$

Hình 3.4. Phép quay quanh trục Oy



### 3.2.3.3 Phép quay quanh trục Oz

- Điểm  $P(x, y, z, 1)$  khi quay quanh trục  $Oz$  một góc  $\theta \rightarrow P'(x', y', z', 1)$ .



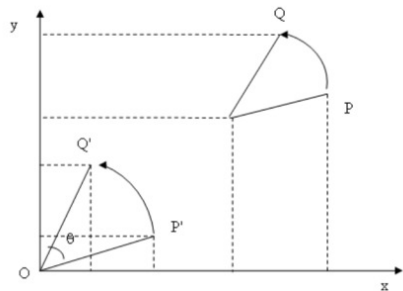
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [M_{Rz}] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$\mathbf{P}_{\text{world}} = [\mathbf{M}_{Rz}] * \mathbf{P}_{\text{obj}}$

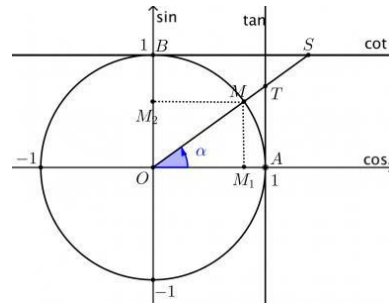
Hình 3.5. Phép quay quanh trục Oz



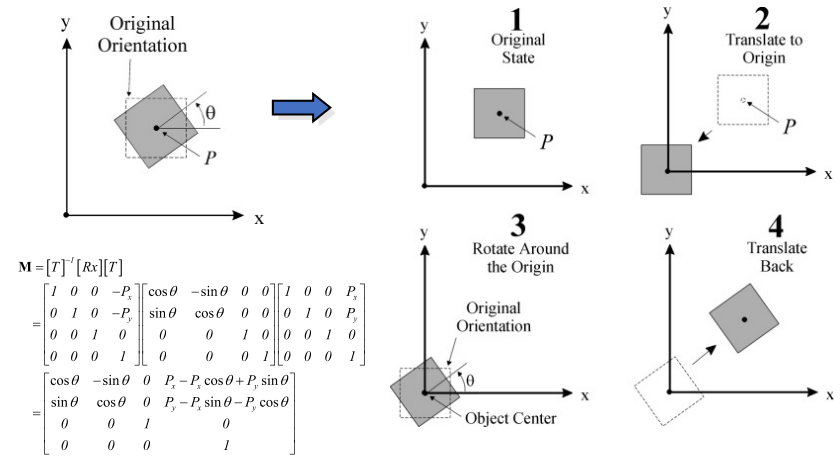
### 3.2.3.4 Quay trong 2D một đối tượng quanh tâm $O'(x, y)$



Quay counterclockwise một đối tượng quanh tâm  $O'(x, y)$

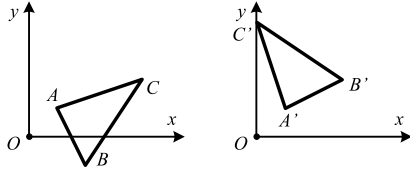


Đường tròn lượng giác





**Ví dụ 3.3.** Quay trong 2D tam giác ABC có tọa độ các đỉnh A(1, 1), B(2, -1), C(4, 2) một góc  $90^\circ$  quanh A.



Hình 3.6. Quay tam giác ABC một góc  $90^\circ$

Quá trình quay counterclockwise tam giác ABC bao gồm các bước như sau:

- Tịnh tiến tam giác ABC về gốc tọa độ sao cho  $A \equiv O$
- Quay quanh gốc tọa độ một góc  $\alpha = 90^\circ$ :
- Tịnh tiến tam giác ABC để đưa tâm quay về vị trí ban đầu.

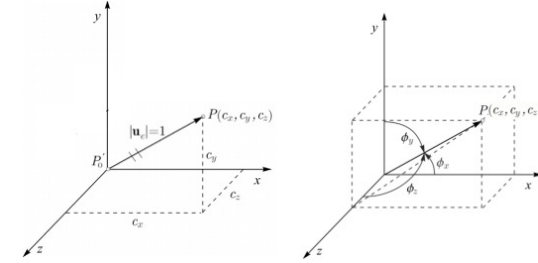


### 3.2.3.5 Phép quay quanh một vector $u$ ( $u_x, u_y, u_z$ )

Bước 1: Chuẩn hóa vector  $u$  (normalize vector) :

$$u_e = \frac{u}{|u|} = (c_x, c_y, c_z) \quad h = \sqrt{u_x^2 + u_y^2 + u_z^2} = 1, c_x = \frac{u_x}{h}, c_y = \frac{u_y}{h}, c_z = \frac{u_z}{h}$$

Vector chuẩn hóa  $u_e$  được xác định:  $c_x^2 + c_y^2 + c_z^2 = 1$   $\cos \theta_x = c_x, \cos \theta_y = c_y, \cos \theta_z = c_z$



Hình 3.7. Quay quanh trục là vecto  $u$



Bước 2: Quay quanh trục  $Ox$  một góc  $\theta_x$  để vector  $u$  nằm trên mặt  $Oxz$ : ma trận quay  $[M_{Rx}]^{\theta_x}$

$$[Rx]^{\theta_x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x & 0 \\ 0 & \sin \theta_x & \cos \theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{c_z}{d} & \frac{-c_y}{d} & 0 \\ 0 & \frac{c_y}{d} & \frac{c_z}{d} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$d = \sqrt{c_y^2 + c_z^2}, \cos \theta_x = \frac{c_z}{d}, \sin \theta_x = \frac{c_y}{d}$$

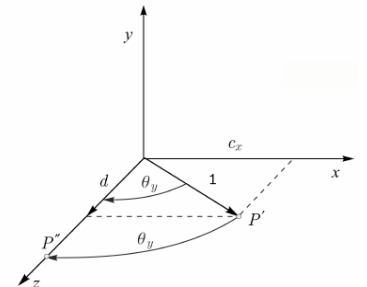
Hình 3.8. Quay  $P_0P_1$  quanh counterclockwise quanh trục  $Ox$



Bước 3: Quay quanh trục  $Oy$  một góc  $-\theta_y$  để vector  $u$  nằm trên trục  $Oz$ : ma trận quay  $[Ry]^{-\theta_y}$

$$[Ry]^{-\theta_y} = \begin{bmatrix} \cos \theta_y & 0 & \sin \theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta_y & 0 & \cos \theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} d & 0 & -c_x & 0 \\ 0 & 1 & 0 & 0 \\ c_x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$h = \sqrt{c_x^2 + c_y^2 + c_z^2} = 1, \cos \theta_y = \frac{d}{h} = d, \sin \theta_y = \frac{-c_x}{h} = -c_x$$

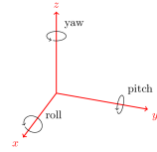


Hình 3.9. Quay quanh trục  $Oy$  để  $P$  nằm trên  $Oz$ .



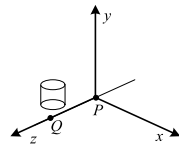
Bước 4: Quay điểm  $Q$  quanh trục  $Oz$  một góc  $\theta_z$

$$[R_z]^{\theta_z} = \begin{bmatrix} \cos \theta_z & -\sin \theta_z & 0 & 0 \\ \sin \theta_z & \cos \theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Hình 3.10. Quay quanh trục  $Oz$  một góc  $\theta$ .

Bước 5: Quay lại vector  $u$  lại quanh trục  $Oy$  một góc  $\theta_y$  :

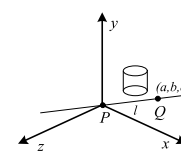


$$[R_y]^{-\theta_y} = \begin{bmatrix} \cos \theta_y & 0 & \sin \theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta_y & 0 & \cos \theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} d & 0 & -c_x & 0 \\ 0 & 1 & 0 & 0 \\ c_x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Hình 3.11. Quay ngược  $PQ$  quanh trục  $Oy$



Bước 6: Quay vector  $u$  ngược lại quanh trục  $Ox$  một góc  $-\theta_x$  :



$$[R_x]^{-\theta_x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x & 0 \\ 0 & \sin \theta_x & \cos \theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{c_z}{d} & \frac{-c_y}{d} & 0 \\ 0 & \frac{c_y}{d} & \frac{c_z}{d} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Hình 3.12. Quay ngược  $PQ$  quanh trục  $Ox$



Ma trận biến hình là tích các ma trận thành phần.

$$[R]_u^\theta = [R_x]^{-\theta_x} [R_y]^{-\theta_y} [R_z]^{\theta_z} [R_y]^{\theta_y} [R_x]^{\theta_x} =$$

$$\begin{bmatrix} \cos \theta + c_x^2 (1 - \cos \theta) & c_x c_y (1 - \cos \theta) - c_z \sin \theta & c_x c_z (1 - \cos \theta) + c_y \sin \theta & 0 \\ c_y c_x (1 - \cos \theta) + c_z \sin \theta & \cos \theta + c_y^2 (1 - \cos \theta) & c_y c_z (1 - \cos \theta) - c_x \sin \theta & 0 \\ c_z c_x (1 - \cos \theta) - c_y \sin \theta & c_z c_y (1 - \cos \theta) + c_x \sin \theta & \cos \theta + c_z^2 (1 - \cos \theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = [R]_u^\theta \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$x' = (\cos(\theta) + c_x^2 * (1 - \cos(\theta))) * x + (c_x * c_y * (1 - \cos(\theta)) - c_y * \sin(\theta)) * y + (c_x * c_z * (1 - \cos(\theta)) + c_x * \sin(\theta)) * z$$

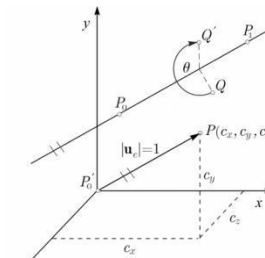
$$y' = (c_x * c_y * (1 - \cos(\theta)) + c_z * \sin(\theta)) * x + (\cos(\theta) + c_y^2 * (1 - \cos(\theta))) * y + (c_y * c_z * (1 - \cos(\theta)) - c_x * \sin(\theta)) * z$$

$$z' = (c_x * c_z * (1 - \cos(\theta)) - c_y * \sin(\theta)) * x + (c_y * c_z * (1 - \cos(\theta)) + c_x * \sin(\theta)) * y + (\cos(\theta) + c_z^2 * (1 - \cos(\theta))) * z$$



### 3.2.4 Phép quay quanh một đoạn thẳng trong 3D

- Cho hai điểm  $P_0(x_0, y_0, z_0)$ ,  $P_1(x_1, y_1, z_1)$ .
- Quay điểm  $Q(x, y, z, l)$  một góc  $\theta$  quanh đường thẳng  $P_0P_1$  theo hướng nhìn từ  $P_0 \rightarrow P_1$

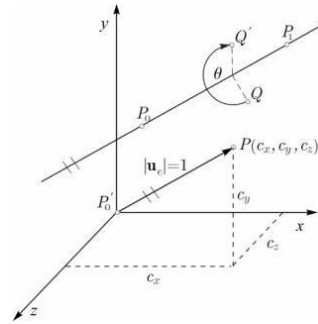


Hình 3.13. Quay quanh trục là đường thẳng  $PQ$



1) Tịnh tiến đường  $P_0P_1$  để điểm  $P_0$  trùng với gốc  $O$ : ma trận tịnh tiến  $[M_T]$

$$[T] = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Hình 3.14. Tịnh tiến đoạn  $P_0P_1$  về gốc tọa độ

44



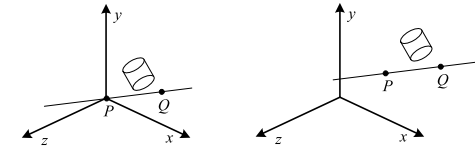
2) Tính  $V = P_1 - P_0$ . Tính vectơ đơn vị  $u$  của  $V$

$$V = P_1 - P_0 = (x_1 - x_0, y_1 - y_0, z_1 - z_0)$$

$$\text{Với } u \text{ là vector đơn vị theo } V: u = \frac{V}{|V|} = (a, b, c)$$

3) Thực hiện quay quanh vector  $u$ .

4) Tịnh tiến  $P_0P_1$  về lại vị trí ban đầu: ma trận tịnh tiến  $[M_T]^{-1}$



a) Quay trục  $P_0P_1$  trở về hướng ban đầu b) Tịnh tiến  $P_0P_1$  trở về vị trí cũ

Hình 3.23. Quay và tịnh tiến trục  $P_0P_1$  về vị trí ban đầu

Tích các ma trận thành phần:  $[R_{PQ}]^{\theta} = [T]^{-1} [R_{Ox}]^{-\theta_x} [R_{Oy}]^{-\theta_y} [R_{Oz}]^{\theta_z} [R_{Oy}]^{\theta_y} [R_{Ox}]^{\theta_x} [T]$

45



**Ví dụ:** Cho  $P_0(6, -2, 0)$ ,  $P_1(12, 8, 0)$ . Quay điểm  $Q(10, 6, 0, 1)$  một góc  $60^\circ$  quanh trục  $P_0P_1$  theo hướng từ  $P_0 \rightarrow P_1$ .

Ta có:

- $u_x = 12 - 6 = 6$ ;  $u_y = 8 - (-2) = 10$ ;  $u_z = 0 - 0 = 0$
- $h = \sqrt{u_x^2 + u_y^2 + u_z^2} = \sqrt{6^2 + 10^2 + 0^2} = 11.6619$
- $c_x = u_x/h = 6/11.6619$ ;  $c_y = u_y/h = 10/11.6619$ ;  $c_z = 0$
- $d = \sqrt{10^2 + 0^2} = 10$

B1. Tịnh tiến điểm P về gốc tọa độ O

1.0	0.0	0.0	-6.00
0.0	1.0	0.0	2.00
0.0	0.0	1.0	0.00
0.0	0.0	0.0	1.0

[T]

46



B2. Quay  $PQ$  quanh trục  $Ox$  góc alpha để  $PQ$  nằm trên mặt phẳng  $Oxz$

1.0	0.0	0.0	0.0
0.0	0.00	-1.00	0.0
0.0	1.00	0.00	0.0
0.0	0.0	0.0	1.0

[Rx]

B3. Quay  $PQ$  quanh trục  $Oy$  góc theta để  $PQ$  nằm trên trục  $Oz$

0.857	0.0	-0.514	0.0
0.0	1.0	0.0	0.0
0.514	0.0	0.857	0.0
0.0	0.0	0.0	1.0

[Ry]

47



B4. Quay PQ quanh trục Oz góc

0.500	-0.866	0.0	0.0
0.866	0.500	0.0	0.0
0.0	0.0	1.0	0.0
0.0	0.0	0.0	1.0

[Rz]

B5. Quay PQ quanh trục Oy góc ngược

0.857	0.0	0.514	0.0
0.0	1.0	0.0	0.0
-0.514	0.0	0.857	0.0
0.0	0.0	0.0	1.0

[Ry\*]

48



B6. Quay PQ quanh trục Ox góc ngược

1.0	0.0	0.0	0.0
0.0	0.00	1.00	0.0
0.0	-1.00	0.00	0.0
0.0	0.0	0.0	1.0

[Rx\*]

B7. Tịnh tiến ngược điểm P về tọa độ ban đầu

1.0	0.0	0.0	6.00
0.0	1.0	0.0	-2.00
0.0	0.0	1.0	0.00
0.0	0.0	0.0	1.0

[T\*]

[T\*][Rx\*]

49



1.00	0.00	0.00	6.00
0.00	0.00	1.00	-2.00
0.00	-1.00	0.00	0.00
0.00	0.00	0.00	1.00

[T\*][Rx\*][Ry\*]

0.857	0.00	0.514	6.00
-0.514	0.00	0.857	-2.00
0.00	-1.00	0.00	0.00
0.00	0.00	0.00	1.00

[T\*][Rx\*][Ry\*][Rz]

0.429	-0.743	0.514	6.00
-0.257	0.446	0.857	-2.00
-0.866	-0.500	0.00	0.00
0.00	0.00	0.00	1.00

[T\*][Rx\*][Ry\*][Rz][Ry]

[T\*][Rx\*][Ry\*][Rz][Ry][Rx]

50



0.632	-0.743	0.221	6.00
0.221	0.446	0.868	-2.00
-0.743	-0.500	0.446	0.00
0.00	0.00	0.00	1.00

0.632	0.221	0.743	6.00
0.221	0.868	-0.446	-2.00
-0.743	0.446	0.500	0.00
0.00	0.00	0.00	1.00

P<sub>new</sub>

9.140  
0.316  
7.401  
1.00

[T\*][Rx\*][Ry\*][Rz][Ry][Rx][T]

0.632	0.221	0.743	2.647
0.221	0.868	-0.446	-1.588
-0.743	0.446	0.500	5.347
0.00	0.00	0.00	1.00

x

P<sub>world</sub>

3.00
4.00
5.00
1.00

51



### ▪ Ví dụ phép quay trong OpenGL

Các lệnh OpenGL biểu diễn phép quay 45-degree quanh đường thẳng OP với O là gốc tọa độ, P(1,2,3) tại điểm quay (4, 5, 6) như sau:

```
void myinit(void){
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glTranslatef(4.0, 5.0, 6.0);
    glRotatef(45.0, 1.0, 2.0, 3.0);
    glTranslatef(-4.0, -5.0, -6.0);
}
```

Ma trận kết hợp như sau:

$$C = T(4,5,6)R(45,1,2,3)T(-4,-5,-6).$$

$$P_{\text{new}} = C * P$$

52



### 3.2.5 Ứng dụng của phép quay

- Đồ họa máy tính: biến đổi và hiển thị các đối tượng 3D trên màn hình, dùng để xoay, di chuyển và tỉ lệ các đối tượng một cách linh hoạt và hiệu quả.
- Robot học: mô phỏng và điều khiển chuyển động của các bộ phận robot, giúp robot di chuyển, xoay và tương tác với môi trường xung quanh.
- Định vị và định hướng: sử dụng trong hệ thống định vị toàn cầu (GPS) và các phương pháp định vị khác để xác định vị trí và hướng di chuyển của các vật thể. Giúp xác định vị trí và hướng của các phương tiện đi lại, máy bay, tàu thủy và thiết bị di động.
- Công nghệ hình ảnh: Xoay hình ảnh, điều chỉnh góc nhìn, loại bỏ biến dạng và thực hiện các phép biến đổi khác trên hình ảnh.
- Mô phỏng và thiết kế 3D: Mô phỏng chuyển động, làm xoay các đối tượng và thực hiện các phép biến đổi để tạo ra các mô hình 3D chân thực.
- Công nghiệp và tự động hóa: Điều khiển hoạt động của các máy và thiết bị tự động hóa khác. Giúp cho việc điều khiển và điều chỉnh chính xác các máy móc, robot và hệ thống tự động hóa.

53



## 3.3 Phép đối xứng (reflection)

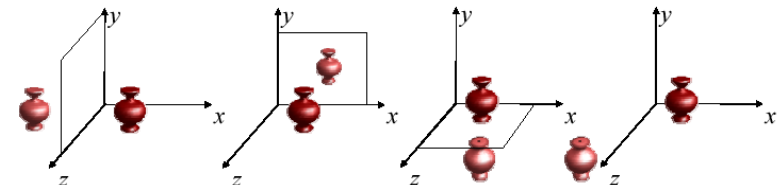
### 3.3.1 Đối xứng qua mặt phẳng Oxy, Oyz, Ozx

- Nếu xem trục tọa độ là trục đối xứng thì phép đối xứng tương đương với phép quay 180° quanh trục đó.
- Xét phép đối xứng qua mặt Oxy tương ứng với phép quay quanh trục Ox góc 180°:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(180) & -\sin(180) & 0 \\ 0 & \sin(180) & \cos(180) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [Re_{Ox}] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [Rf_{Ox}] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

54



Đối xứng qua Oyz

Đối xứng qua Oxy

Đối xứng qua Oxz

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P_{\text{world}} = [M_{\text{ref}}] * P_{\text{obj}}$$

55



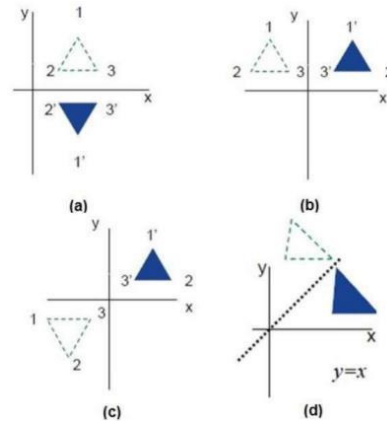
### 3.3.2 Đối xứng qua một điểm (qua tâm)

- Điểm  $P(x_p, y_p, z_p, I)$  sau khi đối xứng qua tâm  $Q(a, b, c) \rightarrow P'(x', y', z', I)$

$$x' = x_p + 2(a - x_p)$$

$$y' = y_p + 2(b - y_p)$$

$$z' = z_p + 2(c - z_p)$$



56



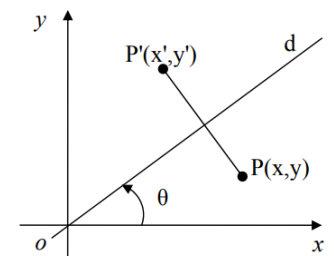
### 3.3.3 Đối xứng qua một đường thẳng đi gốc tọa độ

- Xét phép đối xứng qua một đường thẳng  $d$  qua gốc tọa độ và tạo với  $Ox$  một góc  $\theta$
- Áp dụng liên tiếp các phép biến đổi theo thứ tự sau đây:

(1) Quay  $R(-\theta)$  đưa đường thẳng  $d$  về trùng với trục  $Ox$

(2) Lấy đối xứng  $S_{Ox}$  qua trục  $Ox$

(3) Quay  $R(\theta)$  đưa đường thẳng  $d$  về vị trí ban đầu

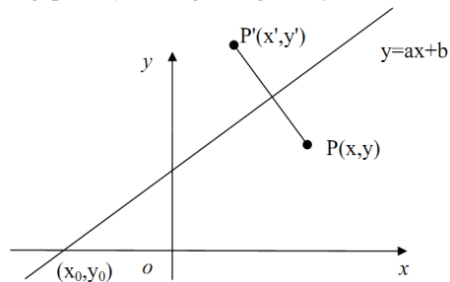


Hình 3.15. Phép đối xứng qua một đường thẳng qua gốc tọa độ

57



### 3.3.4 Đối xứng qua một đường thẳng bất kỳ



$$M_L = \begin{bmatrix} \frac{1-m^2}{1+m^2} & \frac{2m}{1+m^2} & \frac{-2bm}{1+m^2} \\ \frac{2m}{1+m^2} & \frac{m^2-1}{1+m^2} & \frac{2b}{1+m^2} \\ 0 & 0 & 1 \end{bmatrix}$$

Hình 3.16. Phép đối xứng qua đường thẳng  $y = mx + b$

### 3.3.5 Đối xứng qua một mặt phẳng bất kỳ

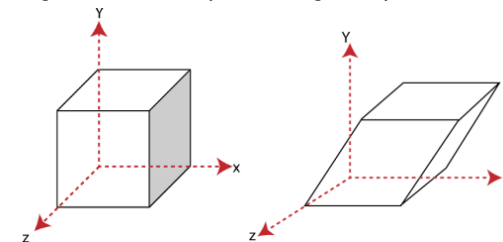
...

58



### 3.4 Phép biến dạng (shear)

- Phép Shear làm thay đổi (hoặc biến dạng) hình dạng của vật thể.
- Biến dạng theo bất kỳ trục tọa độ nào cũng bị ảnh hưởng bởi tọa độ tương ứng với hai trục còn lại. Tâm là gốc tọa độ.
- Phép Shear theo hướng các trục:  $Ox, Oy, Oz$  một giá trị tỷ lệ  $h$



- Phép Shear theo trục  $Ox$ : Ở đây tọa độ của  $x$  của đối tượng không đổi trong khi tọa độ của  $y$  và  $z$  thay đổi.

59





$$[S_x] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ h_y & 1 & 0 & 0 \\ h_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Phép Shear theo trục Oy: Tọa độ của y của đối tượng không đổi trong khi tọa độ của x, z thay đổi.

$$[S_y] = \begin{bmatrix} 1 & h_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & h_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Phép Shear theo trục Oz: Tọa độ của y của đối tượng không đổi trong khi tọa độ của x, y thay đổi.

60



$$[S_y] = \begin{bmatrix} 1 & 0 & h_x & 0 \\ 0 & 1 & h_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Phép Shear tổng quát:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & h_{yx} & h_{zx} & 0 \\ h_{xy} & 1 & h_{zy} & 0 \\ h_{xz} & h_{yz} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = [S] \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Tất cả các phần tử nằm trên đường chéo chính = 1

61



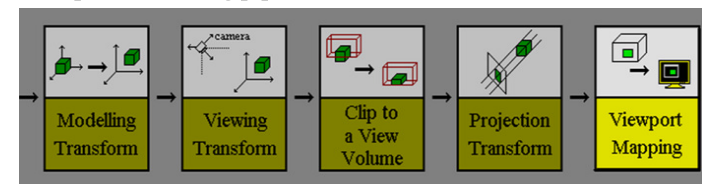
### 3.5 Phân rã phép biến đổi

- Một phép biến đổi bất kì có thể được phân rã thành tích các phép biến đổi cơ sở như tịnh tiến, quay, tỉ lệ.

62



### 3.6 Overview OpenGL viewing pipeline



- 1) Modelview matrix – Projection matrix using:

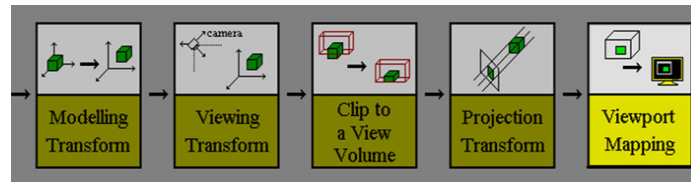
**glMatrixMode(GL\_MODELVIEW):** Model transform, View transform

- glTranslate()
- glRotate()
- glScale()
- glLoadMatrix()
- glMultMatrix()
- gluLookAt(x0, y0, z0, xref, yref, zref, Vx, Vy, Vz)

63



## Overview OpenGL viewing pipeline



### 4) Projection matrix:

**glMatrixMode (GL\_PROJECTION):** *Projection transform, Clipping, Perspective divide*

- `gluOrtho2D (xwmin, xwmax, ywmin, ywmax)`
- `gluPerspective (theta, aspect, dnear, dfar)`
- `glFrustum (xwmin, xwmax, ywmin, ywmax, dnear, dfar)`

### 5) Viewport matrix:

- `glViewport (xPos, yPos, xSize, ySize)`
- `glutInitWindowSize (width, height)`
- `glutInitWindowPosition (x, y)`
- `glDepthRange()`

64



## 3.7 Phép biến đổi affine với OpenGL

### 3.7.1 Biểu diễn ma trận

- Trong OpenGL các điểm được biểu diễn dưới hệ tọa độ thuần nhất.
- Tọa độ của một điểm 3D được thể hiện bởi  $(x, y, z, w)^T$ , thông thường  $w = 1$   
 Chú ý: *OpenGL biểu diễn vector điểm ở dạng cột.*
- Một phép biến đổi trên một điểm  $P(x, y, z, w)$  tương ứng với việc nhân ma trận cột  $P$  với ma trận biến đổi  $M$  kích thước  $4 \times 4$ :  $P' = M.P$
- Trong mỗi bước ModelView và Projection, OpenGL đều lưu trữ ma trận biến đổi hiện hành.
- Hàm xử lý ma trận biến đổi cho quá trình ModelView: `glMatrixMode(GL_MODELVIEW)`
- Hàm xử lý ma trận biến đổi cho quá trình Projection: `glMatrixMode(GL_PROJECTION)`

65



Để thiết lập ma trận biến đổi hiện hành bằng ma trận M, dùng hàm:

**void glLoadMatrix{fd}(const TYPE \*m);**

Ma trận M có dạng :

$$[M] = \begin{bmatrix} m_1 & m_5 & m_9 & m_{13} \\ m_2 & m_6 & m_{10} & m_{14} \\ m_3 & m_7 & m_{11} & m_{15} \\ m_4 & m_8 & m_{12} & m_{16} \end{bmatrix}$$

Để hỗ trợ các thao tác thay đổi ma trận hiện hành, nhưng muốn khôi phục lại nó, OpenGL sử dụng stack cho mỗi loại ma trận hiện hành, với các hàm sau:

- Đẩy ma trận hiện hành vào trong stack: `void glPushMatrix(void)`
- Lấy ma trận hiện hành ở đỉnh stack: `void glPopMatrix(void)`

Ví dụ như dời tới một điểm nào đó để vẽ khối hộp, sau đó muốn trở lại vị trí ban đầu.

66



### 3.7.2 Các phép biến đổi Affine trong OpenGL

a) Translation: `glTranslate (dx, dy, dz);`

$$M_{\text{modelview}} = M_{\text{modelview}} * T(dx, dy, dz)$$

b) Rotation: `glRotate*(angle, x, y, z)`

$$M_{\text{modelview}} = M_{\text{modelview}} * R(a);$$

Where  $R(a)$  represents the rotation matrix around a vector whose coordinates are x, y and z.

c) Scaling: `glScale*(sx, sy, sz);`

d) Nhân ma trận M: **`void glMultMatrix();`**

Chú ý:

- Các thao tác biến đổi trên đều có nghĩa là lấy ma trận biến đổi hiện hành nhân với ma trận biến đổi affine cần thực hiện.
- Thứ tự thực hiện sẽ *ngược* với suy nghĩ thông thường, do là tọa độ được biểu diễn dạng vector cột – Chú ý  $(AB)^T = B^T A^T$

67



OpenGL functions	Model matrix	World space
glTranslatef( dx, dy, dz )	$T(d) = \begin{pmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$	$T_V = \begin{pmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x+d_x \\ y+d_y \\ z+d_z \\ 1 \end{pmatrix}$
glScalef( sx, sy, sz ).	scaling as 3x3 matrix $S(s_x, s_y, s_z) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix}$	$S_V = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} s_x x \\ s_y y \\ s_z z \end{pmatrix}$
glRotatef( degrees, ax, ay, az );	$R_x(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad R_z = \begin{pmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$	

68



**Ví dụ:** Thực hiện phép quay quanh trục Oz một góc  $\alpha$  và tịnh tiến đi một đoạn theo vector  $(tr_x, tr_y, tr_z)$ , các bước thực hiện sẽ là

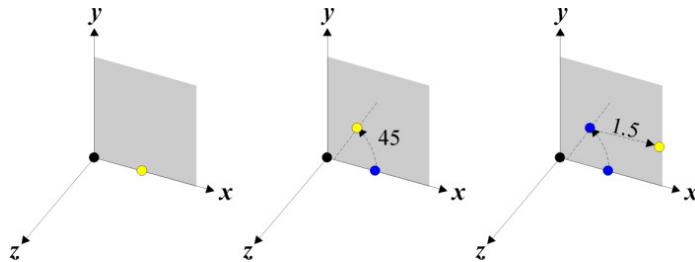
Thao tác	Ma trận hiện hành
Khởi tạo <b>glMatrixMode(GL_MODELVIEW)</b> <b>glLoadIdentity()</b>	$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$
Phép tịnh tiến: <b>glTranslatef(tr<sub>x</sub>, tr<sub>y</sub>, tr<sub>z</sub>)</b>	$\begin{bmatrix} 1 & & tr_x \\ & 1 & tr_y \\ & & 1 & tr_z \\ & & & 1 \end{bmatrix}$
Phép quay <b>glRotatef(<math>\alpha</math>, 0, 0, 1)</b>	$\begin{bmatrix} \cos \alpha & -\sin \alpha & tr_x \\ \sin \alpha & \cos \alpha & tr_y \\ & & 1 & tr_z \\ & & & 1 \end{bmatrix}$

69



Ví dụ:

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(1.5, 0.0, 0.0);
glRotatef(45.0, 0.0, 0.0, 1.0);
glVertex3f(1.0, 0.0, 0.0);
```

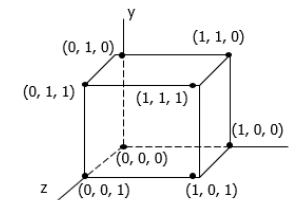


70



## BÀI TẬP

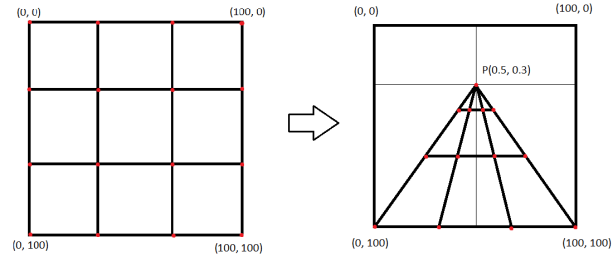
- Một hình chóp A(0, 0, 0), B(1, 0, 0), C(0, 1, 0) và D(0, 0, 1) được xoay một góc  $45^\circ$  quanh đoạn thẳng L được xác định theo hướng  $V = j + k$  và đi qua đỉnh C. Xác định tọa độ các đỉnh sau phép xoay.
- Tìm các tọa độ mới của khối vuông đơn vị như hình bên đây, sau khi xoay quanh một trục xác định bởi điểm A(2, 1, 0) và B(3, 3, 1). Góc xoay là  $90^\circ$  ngược chiều kim đồng hồ.



71



- 3) Cho điểm  $P_1(x_1, y_1)$ , điểm  $P_2(x_2, y_2)$ . Tìm phép biến đổi để  $P_1$  thành  $P_2$ .
- 4) Thực hiện biến đổi tọa độ các đỉnh của khung lưới như sau:



- 5) Lập trình C/C++ tự xây dựng các hàm sau để thay thế các hàm đã có trong OpenGL:

- ***myTranslate(tx, ty, tz);***
- ***myRotate( $\theta$ , ux, uy, uz);*** Quay quanh vector  $u$  ( $ux, uy, uz$ )
- ***myRotate( $\theta$ , P0, P1);*** Quay quanh 1 trục bất kỳ, Theo hướng  $P0 \rightarrow P1$
- ***myScale(sx, sy, sz);***
- ***myReflection(...);***
- ***myShear(sx, sy, sz);***