Adam Piel
Andrea Compton

apiel02
acompt01

COMP175: Introduction to Computer Graphics, Spring 2015

Assignment 2: Camera Algorithm Worksheet

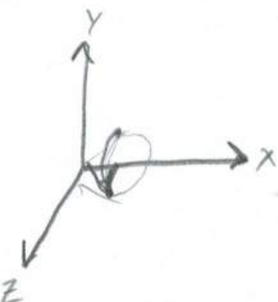
Due Monday March 9, 2015

2)

i) Matrix rotX_mat$\left(\frac{\pi}{4}\right)$ =

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(r) & -\sin(r) & 0 \\ 0 & \sin(r) & \cos(r) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = M$$

$$M \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \sqrt{2} \\ 1 \end{bmatrix}$$

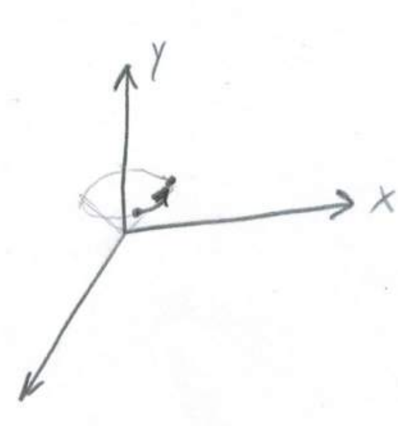


Point (1,1,1) rotates 45°, <u>into</u> the xz plane.
Becomes point $(1, 0, \sqrt{2})$.

ii) Matrix rotY_mat$\left(\frac{\pi}{4}\right)$ =

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = M$$

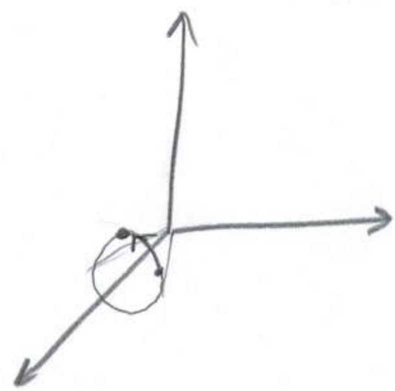
$$M \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} \sqrt{2} \\ 1 \\ 0 \\ 1 \end{bmatrix}$$



Point (1,1,1) rotates 45° into the xy plane. Becomes $(\sqrt{2}, 1, 0)$.

iii) Matrix rotZ_mat$\left(\frac{\pi}{4}\right)$ =

$$\begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = M$$



$$M \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \sqrt{2} \\ 1 \\ 1 \end{bmatrix}$$

Point $(1,1,1)$ rotates 45° around the z axis to $(0, \sqrt{2}, 1)$.

3) $$p' = M_1^{-1} \cdot M_2^{-1} \cdot M_3 \cdot M_2 \cdot M_1 \cdot p$$

M_1: rotates arbitrary axis around Y axis

M_2: rotates axis <u>to lie</u> on x axis

M_3: rotates around x axis

rotate

$p = (p_x, p_y, p_z, 1)$

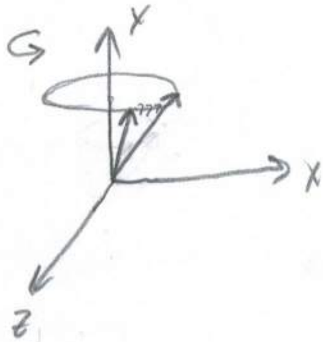
about

$a = \langle a_x, a_y, a_z, 0 \rangle$

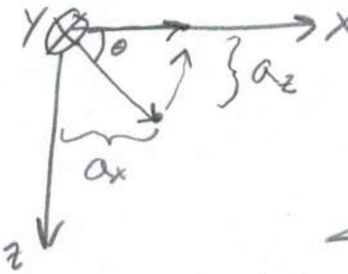
by λ radians

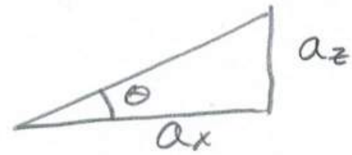
How do we find M_1, M_2, M_3 in terms of rotX_mat, rotY_mat, and rotZ_mat?

i) First, rotate around Y axis, into XY plane



or from the top:

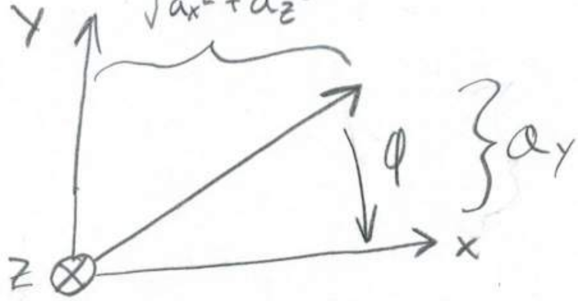




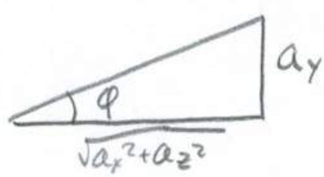
Angle to rotate around y $\Theta = \arctan\left(\frac{a_z}{a_x}\right)$

$$\boxed{M_1 = \text{rotY\_mat}\left(\arctan\left(\frac{a_z}{a_x}\right)\right)}$$

ii) Now, M_2 rotates vector <u>onto</u> x axis



$\varphi = -\arctan\left(\frac{a_y}{\sqrt{a_x^2+a_z^2}}\right)$



$$\boxed{M_2 = \text{rotZ\_mat}\left(-\arctan\left(\frac{a_y}{\sqrt{a_x^2+a_z^2}}\right)\right)}$$

iii) M_3 rotates about the vector by λ:

$$M_3 = \text{rotX\_mat}(\lambda)$$

3.2 To rotate around a point h instead of the origin, we need to add a translation step.

$$M_T = \begin{bmatrix} 1 & 0 & 0 & -h_x \\ 0 & 1 & 0 & -h_y \\ 0 & 0 & 1 & -h_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P' = M_T^{-1} \cdot M_1^{-1} \cdot M_2^{-1} \cdot M_3 \cdot M_2 \cdot M_1 \cdot M_T \cdot P$$

translate so h is at origin

translate back

4)

$$p' = M_1 \cdot \underbrace{M_2}_{\text{Acale}} \cdot \underbrace{M_3}_{\text{Rotation}} \cdot \underbrace{M_4}_{\text{Translation}} \cdot p$$

Translation

Rotation

Acale

Unhinge

M_1: Unhinge: this matrix, applied last, transforms the space from a box, to a trapezoidal volume.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{1+c} & \frac{-c}{1+c} \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad \text{wher} \quad \boxed{c = -\frac{near}{far}}$$

(Clipping planes)

M_2: Acale: this matrix transforms the space from a 2x2x2 box to the actual size needed.

$$M_2 = \begin{bmatrix} \frac{1}{far \cdot \tan(\theta_w/2)} & 0 & 0 & 0 \\ 0 & \frac{1}{far \cdot \tan(\theta_h/2)} & 0 & 0 \\ 0 & 0 & \frac{1}{far} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Where far is the distance to the far clipping plane, θ_w is the width angle, and θ_h is the height angle.

M_3: The rotation matrix aligns the camera axes u, v, w with the x, y, z axes.

$$M_3 = \begin{bmatrix} u_x & u_y & u_z & 0 \\ v_x & v_y & v_z & 0 \\ w_x & w_y & w_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

M_4: The translation matrix moves the view point (camera location) to the origin:

$$M_4 = \begin{bmatrix} 1 & 0 & 0 & -P_{0x} \\ 0 & 1 & 0 & -P_{0y} \\ 0 & 0 & 1 & -P_{0z} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4) part ii)

To move a point one unit right, use the translation matrix:

$$M_R = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

moves one unit in positive x

To move the eye one unit right, we move it to the origin, move it one unit right, and move it back.

$$\mathcal{E}' = \underbrace{M_1 \cdot M_2 \cdot M_3 \cdot M_4}_{\text{Move } \mathcal{E} \text{ back from screen space to world space.}} \cdot \underbrace{M_R}_{\text{Translate one unit right}} \cdot \underbrace{M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M_1^{-1}}_{\text{Move } \mathcal{E} \text{ from world space to screen space}} \cdot \mathcal{E}$$

"Down" and "forward" are the same, but replace M_R above with:

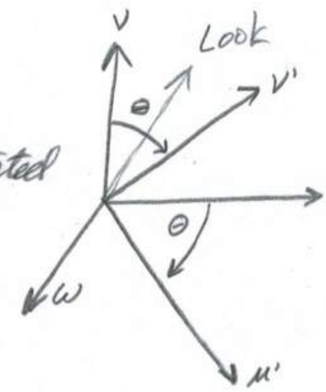
$$M_D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad M_F = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

respectively.

How will you use u, v, w vectors in conjunction with a rotation angle θ to get new u, v, w vectors when:

1) Adjusting the "spin" in a clockwise direction by θ radians?

First, we know w stays the same; spin does not affect the look vector.



$w' = w$

The other two vectors are rotated θ degrees about the w axis.

$$M_\theta = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

($-\theta$ so moving clockwise)

$$V' = M_1 \cdot M_2 \cdot M_3 \cdot M_4 \cdot M_\theta \cdot M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M_1^{-1} \cdot V$$

$$u' = \underbrace{M_1 \cdot M_2 \cdot M_3 \cdot M_4}_{\text{Move back to world space}} \cdot \underbrace{M_\theta}_{\text{Rotate vector}} \cdot \underbrace{M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M}_{\text{Move to screen space}} \cdot u$$

2) "Pitch" to face upword θ radians.
A positive rotation about u axis.

$$M_\theta = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$u' = u$

$V' = M_1 \cdot M_2 \cdot M_3 \cdot M_4 \cdot M_\theta \cdot M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M_1^{-1} \cdot V$

$w' = M_1 \cdot M_2 \cdot M_3 \cdot M_4 \cdot M_\theta \cdot M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M_1^{-1} \cdot w$

3) "Yaw" θ to right.

A positive rotation about v access.

$$M_\theta = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$v' = v$$

$$u' = M_1 \cdot M_2 \cdot M_3 \cdot M_4 \cdot M_\theta \cdot M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M_1^{-1} \cdot u$$

$$w' = M_1 \cdot M_2 \cdot M_3 \cdot M_4 \cdot M_\theta \cdot M_4^{-1} \cdot M_3^{-1} \cdot M_2^{-1} \cdot M_1^{-1} \cdot w$$

5/ Computing a full matrix inverse isn't always the most efficent way of inverting a transformation if you know all the components.

$$T_v = \begin{bmatrix} 1 & 0 & 0 & v_x \\ 0 & 1 & 0 & v_y \\ 0 & 0 & 1 & v_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_v^{-1} = \begin{bmatrix} 1 & 0 & 0 & -v_x \\ 0 & 1 & 0 & -v_y \\ 0 & 0 & 1 & -v_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

So $T_v^{-1} = T_{-v}$, or the inverse of a matrix which translates by a vector v, is the same as a matrix which translates by $-v$.