

# 3-D Mathematical Rendering Polygonal Objects

---

## Rotation About An Arbitrary Axis

---

1. Translate one end of the axis to the origin

$$U = [P_2 - P_1] = [u_1, u_2, u_3]$$

Some useful values:

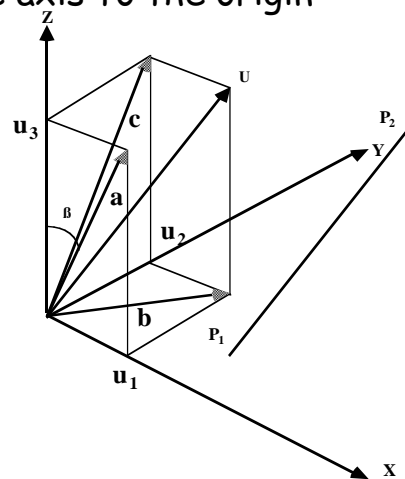
$$a = \sqrt{u_1^2 + u_3^2}$$

$$b = \sqrt{u_1^2 + u_2^2}$$

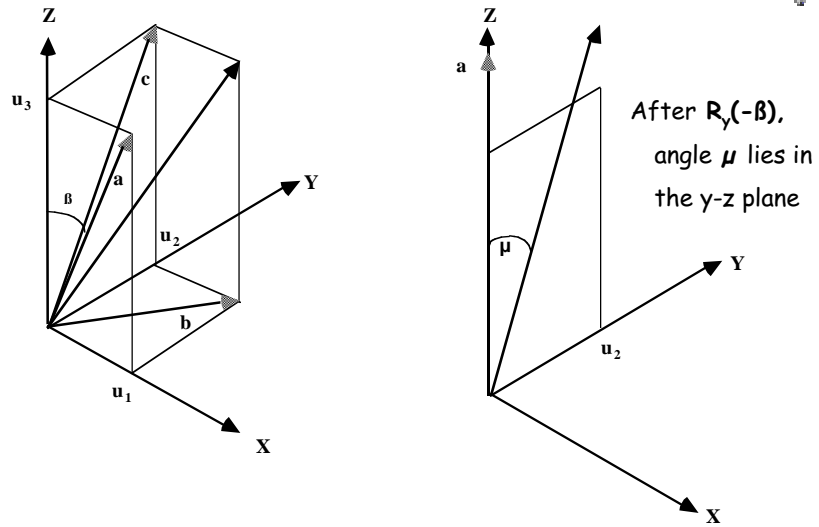
$$c = \sqrt{u_2^2 + u_3^2}$$

$$\cos \beta = u_3/a$$

$$\sin \beta = u_1/a$$



## 2. Rotate $-\beta$ degrees about the y-axis

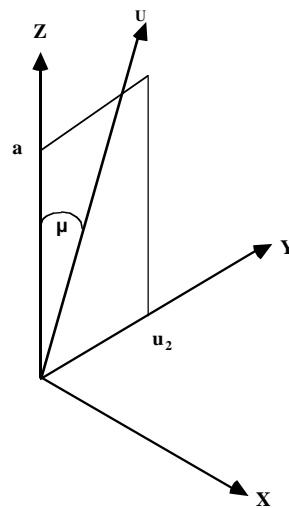


## 3. Rotate $-\mu$ degrees about the x-axis

$R_x(\mu)$

$$\cos \mu = a/||u||$$

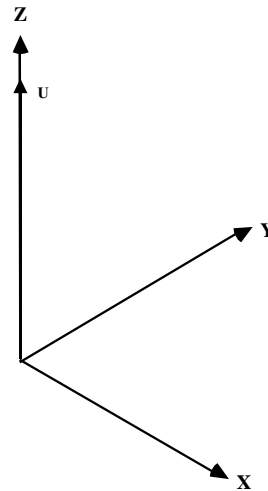
$$\sin \mu = u_2/||u||$$



#### 4. Rotate $R$ degrees about the z-axis

$U$  is aligned with the z-axis

Apply the original rotation,  $R$



5. Apply the inverses of the transformations in reverse order.

#### Rotation About an Arbitrary Axis

$$\mathbf{T}^{-1} R_y(\beta) R_x(-\mu) R R_x(\mu) R_y(-\beta)$$

$\mathbf{T}$

## Alternate view of the Rotation Matrix

---

Given  $P_1, P_2, P_3$

$P_1P_2$  is direction,  $P_1P_3$  is "up"

$$R_{cc} = \begin{pmatrix} r_{1x} & r_{2x} & r_{3x} & 0 \\ r_{1y} & r_{2y} & r_{3y} & 0 \\ r_{1z} & r_{2z} & r_{3z} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

## Alternate view of the Rotation Matrix

---

Z axis rotates to be aligned with  $P_1P_2$

$$R_z = [r_{3x}, r_{3y}, r_{3z}] = \text{normalized } P_1P_2$$

X axis rotates to be normal to  $P_1, P_2, P_3$  plane

$$R_x = [r_{1x}, r_{1y}, r_{1z}] = \text{normalized } P_1P_3 \times P_1P_2$$

Y axis rotates to be normal to  $R_x R_z$  plane

$$R_y = [r_{2x}, r_{2y}, r_{2z}] = \text{normalized } R_z \times R_x$$

## Rendering Polygonal Meshes

---

- Optimize for polygon

- Work:

- ┆ Per polygon

- ┆ Per pixel

## World- $\rightarrow$ ViewSpace

---

- Per vertex

- Transformations

- Clipping

## ViewSpace->Screen

---

- Lots per vertex, incremental per pixel
  - Shading
  - Hidden surface removal
- Polygon order independent

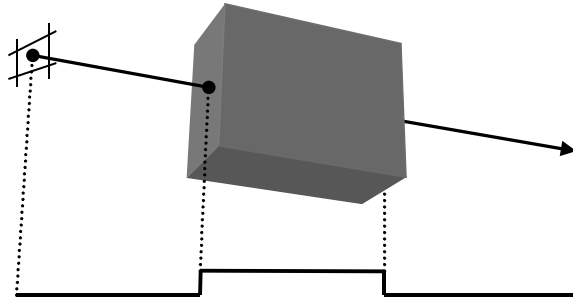
## Rendering CSG

---

- Three ways of rendering CSG reps
  - Ray Tracing
  - Volumetric (Voxel rep)
  - Variation of z-buffer

## Basic Idea: Ray/primitive Classification

---



## Combine Ray Classification

---

## Modify z-buffer for *CSG*

