cs155 – z sweedyk

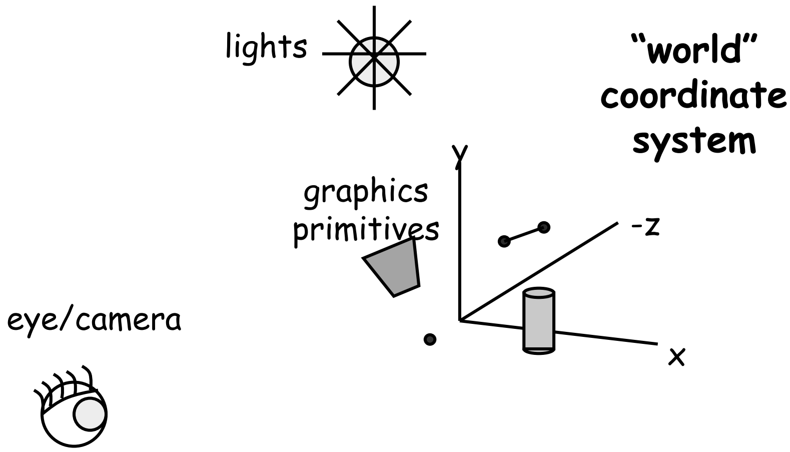
3D graphics

scene graphs

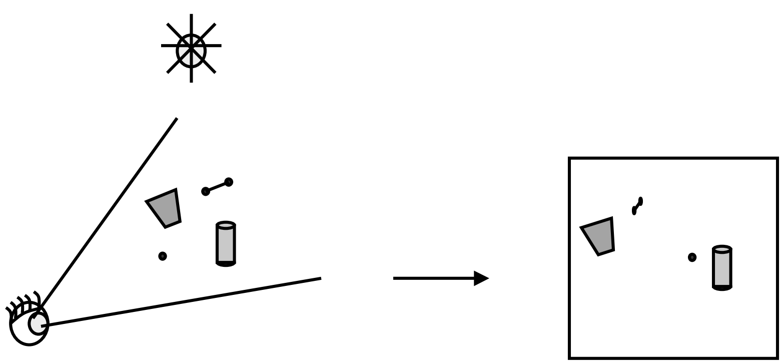
overview

- brief overview of 3D
- modeling transforms & homogenous coordinates
- hierarchical coordinates

3d scene



rendering

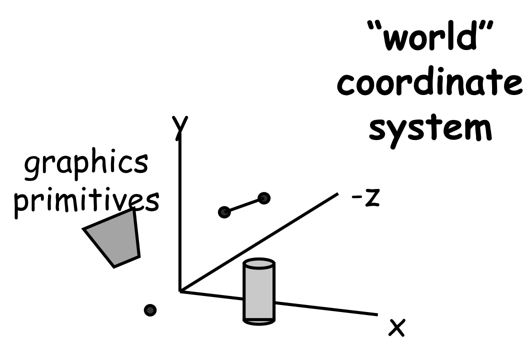


overview

DONE

- brief overview of 3D
- modeling transforms & homogenous coordinates
- hierarchical coordinates

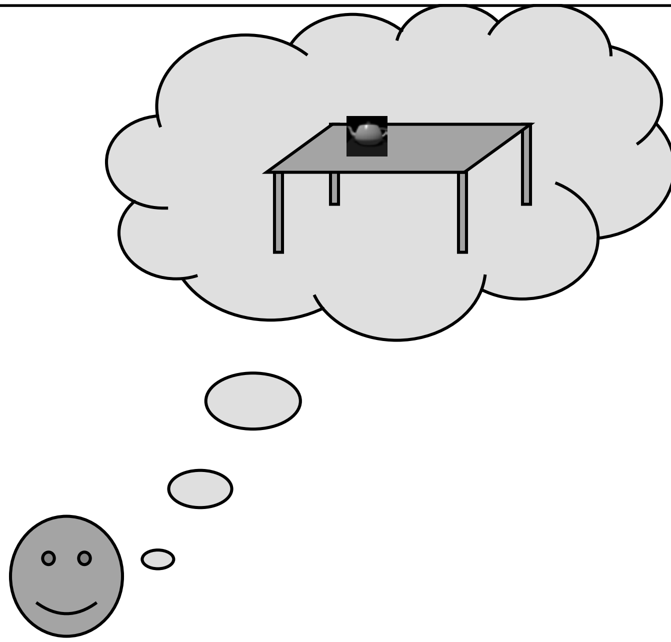
today: composing the scene



geometric primitives

- points
- lines
- triangles
- spheres

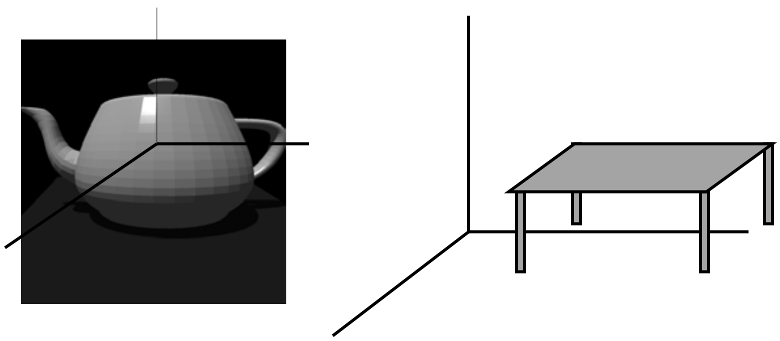
triangle mesh



3d model store

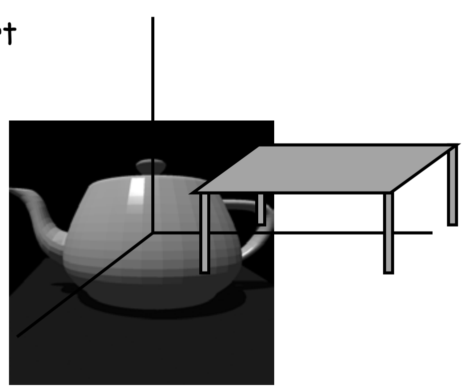
receipt
one triangle mesh teapot
one triangle mesh table

putting it together



putting it together

call Martha Stewart



making it fit

transform teapot

scale by .1

translate by (20,10,2)

making it fit

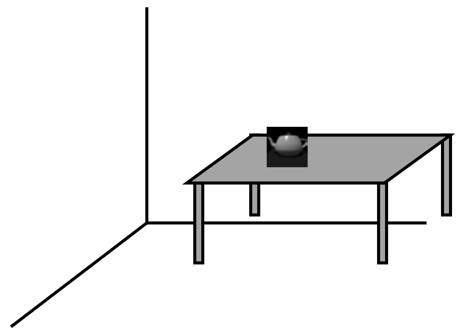
for each triangle in the teapot

scale by .1

translate by (20,10,2)

will this work?

scale and translate teapot triangles ...



now rotate it a little

transforms

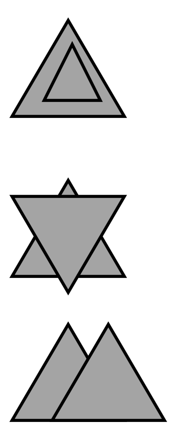
- scale
- rotate
- translate

triangle

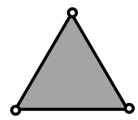
scale

rotate

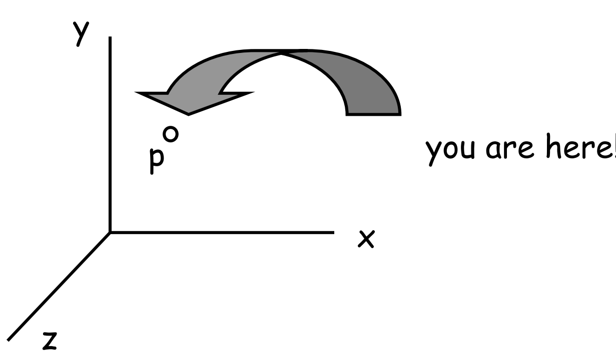
translate



triangle



points



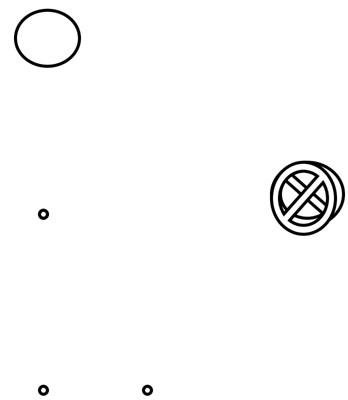
point

call prof gu

scale

rotate

translate



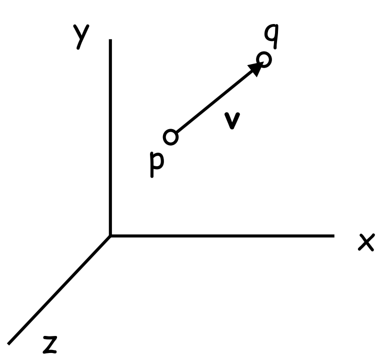
linear algebra

- scalars
- vectors

scalars: real numbers

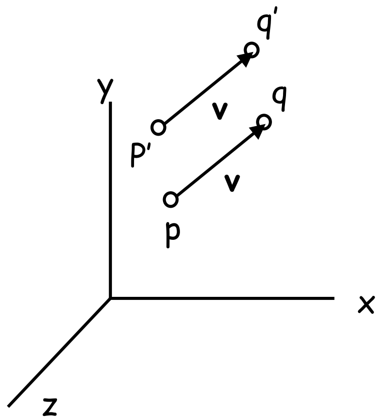
3.8
2.7
4.1
-1000.2
5

vector: magnitude & direction in (3d) space



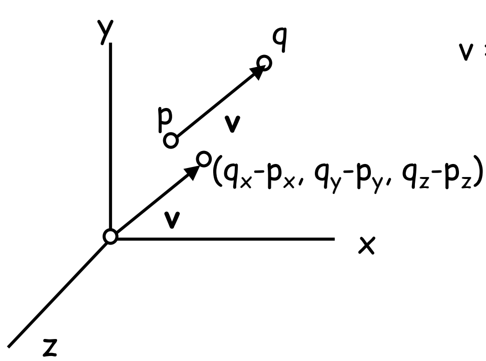
v: the way you get from p to q

vector: magnitude & direction in (3d) space



A vector does not have a position in space !

naming vectors

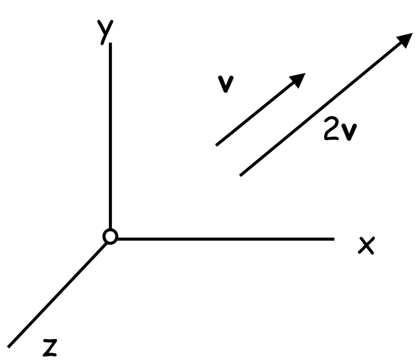


$v = \langle q_x-p_x, q_y-p_y, q_z-p_z \rangle$

linear spaces

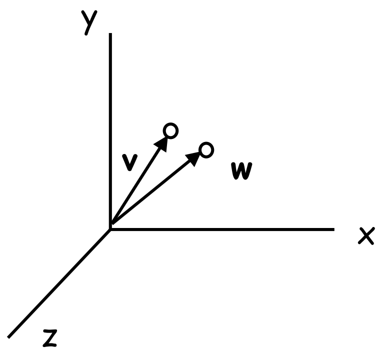
- scalars
- vectors
- scalar multiplication
- vector addition

scalar multiplication



$2v = \langle 2v_x, 2v_y, 2v_z \rangle$

vector addition

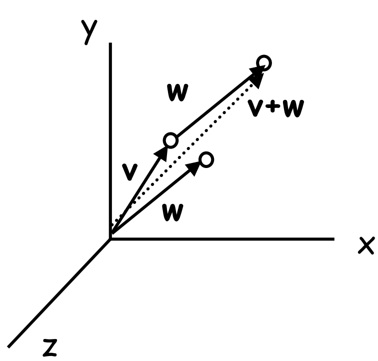


$v = \langle v_x, v_y, v_z \rangle$

$w = \langle w_x, w_y, w_z \rangle$

9/11/2005 29

vector addition

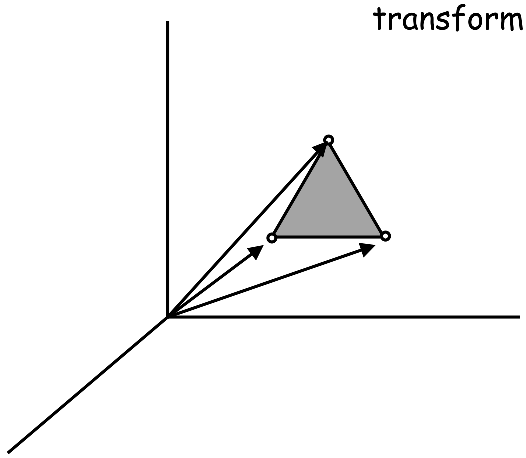


$v+w = \langle v_x+w_x, v_y+w_y, v_z+w_z \rangle$

9/11/2005 30

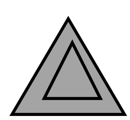
triangle

transform vectors!

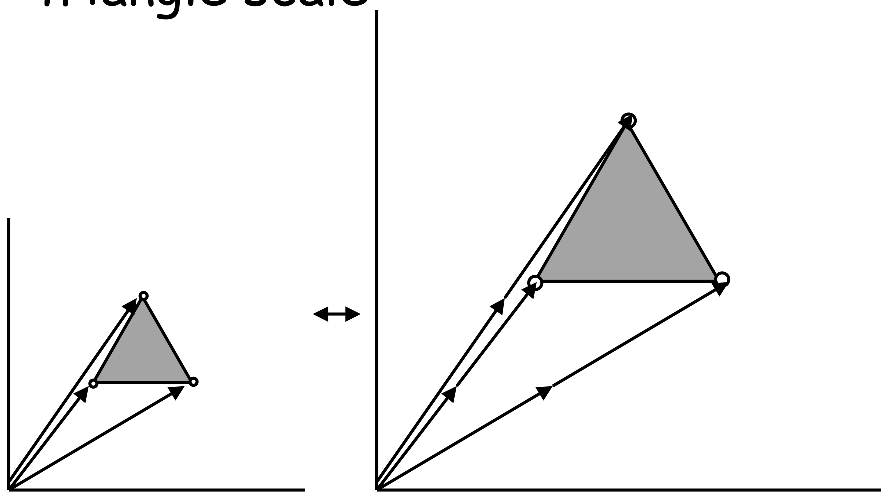


triangle

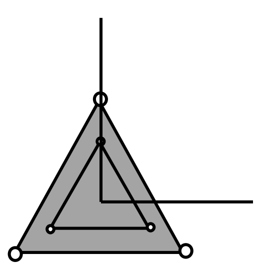
scale



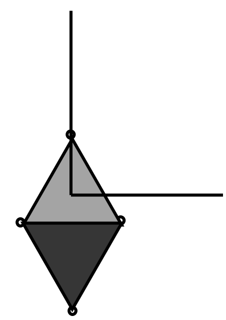
triangle scale



triangle scale

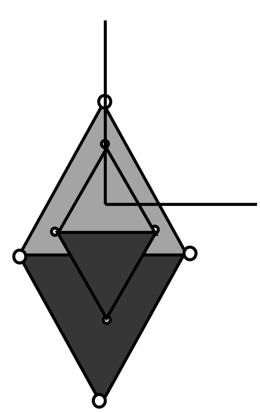


triangle scale



9/11/2005 35

triangle scale



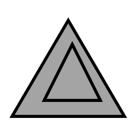
9/11/2005 36

scale

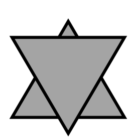
$$\begin{bmatrix} s & 0 & 0 \\ 0 & t & 0 \\ 0 & 0 & u \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} sx \\ ty \\ uz \end{bmatrix}$$

triangle

scale

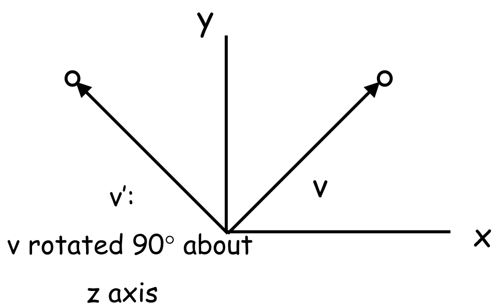


rotate



translate

rotating a vector



v'= ________________

rotation

$$\begin{bmatrix} \cos 90 & -\sin 90 & 0 \\ \sin 90 & \cos 90 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} v_x \cos 90 - v_y \sin 90 \\ v_x \sin 90 + v_y \cos 90 \\ v_z \end{bmatrix}$$

rotate about z axis

$$\begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

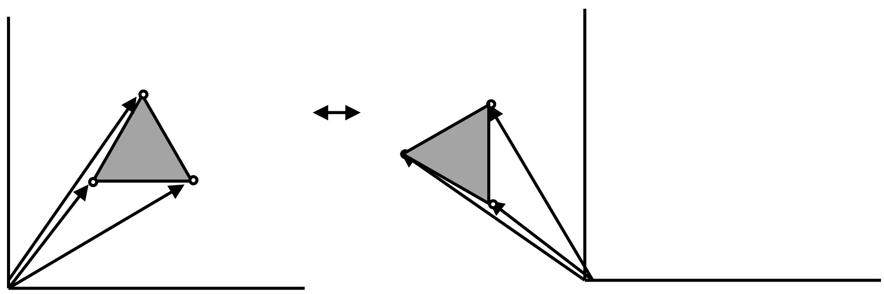
- the first column specifies what happens to (1,0,0)
- the second column specifies what happens to (0,1,0)
- the third column specifies what happens to (0,0,1)

rotation in 3D about x axis

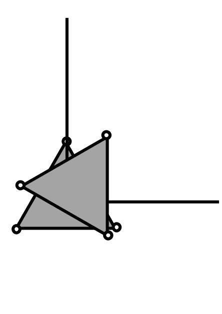
$$\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}$$

fill in the blanks

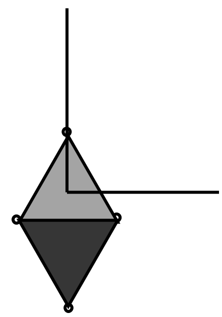
triangle rotate



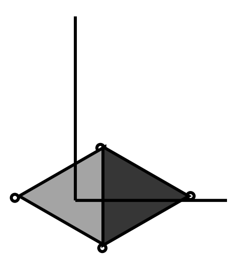
triangle rotate



triangle rotate



triangle rotate

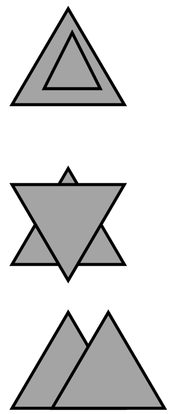


triangle

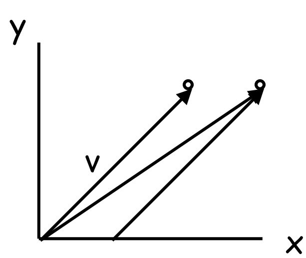
scale

rotate

translate



translating a point



p translated 2 units
to the right

f(<x,y,z>)=<x+2,y,z>

operators

- scale
- rotate

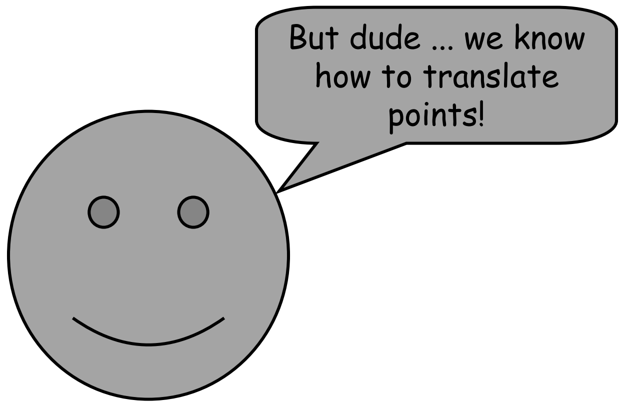
} linear

- translate — non-linear

linear transformation

- f(**v**) is linear if
  $f(\alpha \mathbf{u}+\beta \mathbf{v}) = \alpha f(\mathbf{u})+\beta f(\mathbf{v})$

- translation is not a linear transform: define $T(v) = v+w_0$ where w_0 is a non-zero vector
  $T(u+v) = u + v + w_0$
  $T(u)+T(v) = u+v+2w_0$



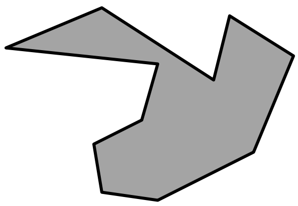
composite transform

$$M_1 v \longrightarrow M_2(M_1 v)$$

$$\Updownarrow$$

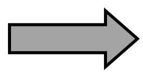
$$(M_2 M_1)v$$

transform polygon mesh



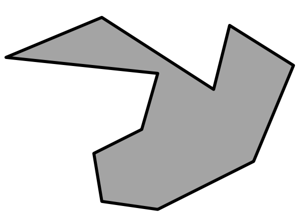
1,000,000
vertices

10 transforms



10,000,000
computations

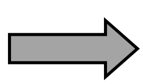
transform polygon mesh



1,000,000
vertices

~~10 transforms~~

1 composite transform



1,000,000

~~10,000,000~~
computations

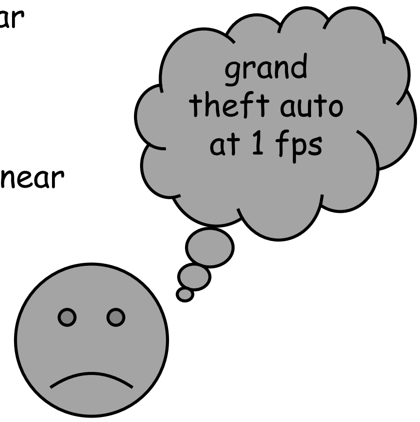
operators

- scale
- rotate

linear

- translate

non-linear



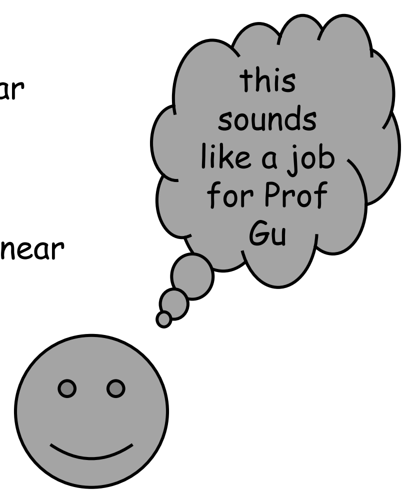
operators

- scale
- rotate

linear

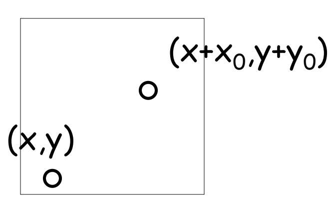
- translate

non-linear

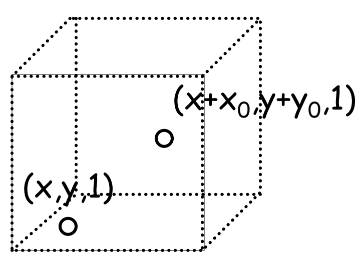


let’s step back into 2D for a moment

hold onto your hat ...



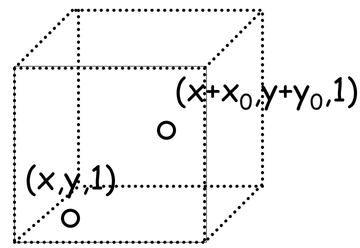
not a 2D linear xfm



3D linear xfm

linear transform

$$\begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+x_0 \\ y+y_0 \\ 1 \end{bmatrix}$$



NOTE: this only works as a
translate for points in the plane z=1!

2D homogenous coordinates

$$(x,y) \Longleftrightarrow (x,y,1)$$

Can we do scale and rotate in homogenous coordinates?

scale

$$\begin{bmatrix} s & 0 & 0 \\ 0 & t & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} sx \\ ty \\ 1 \end{bmatrix}$$

rotate

$$\begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x\cos\phi - y\sin\phi \\ x\sin\phi + y\cos\phi \\ 1 \end{bmatrix}$$

translate

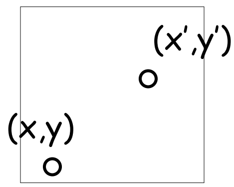
$$\begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+x_0 \\ y+y_0 \\ 1 \end{bmatrix}$$

transform form

$$\begin{bmatrix} ? & ? & ? \\ ? & ? & ? \\ \hline 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

we are not alone…

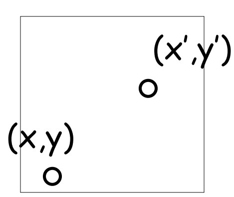
the parallel universe view of homogenous coordinates



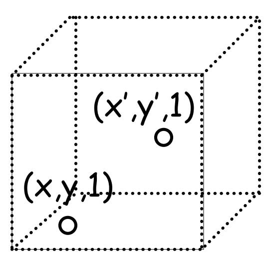
we live in this universe

we are not alone…

the parallel universe view of homogenous coordinates



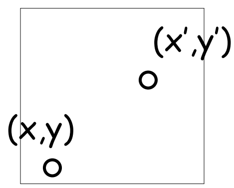
we live in this universe



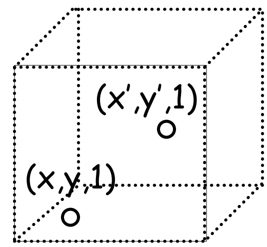
it's not the only one, but it is the only one we can experience!

and its better not to think about it …

the parallel universe view of homogenous coordinates

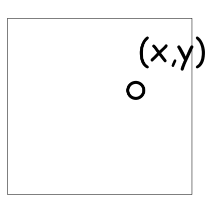


our universe has center (0,0)

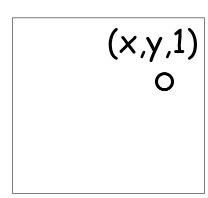


center?

2d and 2d homogenous

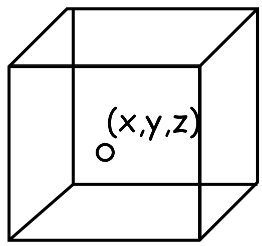


our universe

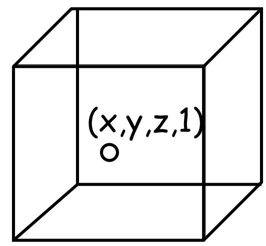


our universe when it comes to computing modeling transforms

3d and 3d homogenous



our universe



our universe when it comes to computing modeling transforms

scale

$$\begin{pmatrix} s & 0 & 0 & 0 \\ 0 & t & 0 & 0 \\ 0 & 0 & u & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} sx \\ ty \\ uz \\ 1 \end{pmatrix}$$

rotate about z axis

$$\begin{pmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x\cos\phi - y\sin\phi \\ x\sin\phi + y\cos\phi \\ z \\ 1 \end{pmatrix}$$

rotate about x & y axes are similar

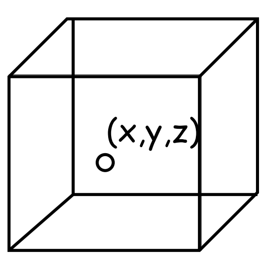
translate

$$\begin{pmatrix} 1 & 0 & 0 & x_0 \\ 0 & 1 & 0 & y_0 \\ 0 & 0 & 1 & z_0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x+x_0 \\ y+y_0 \\ z+z_0 \\ 1 \end{pmatrix}$$

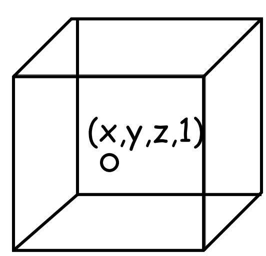
transform form

$$\begin{pmatrix} ? & ? & ? & ? \\ ? & ? & ? & ? \\ ? & ? & ? & ? \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix}$$

3d and 3d homogenous



our universe



our universe when it comes to computing modeling transforms

Composing transforms

$$M_1 M_2 v \neq M_2 M_1 v$$

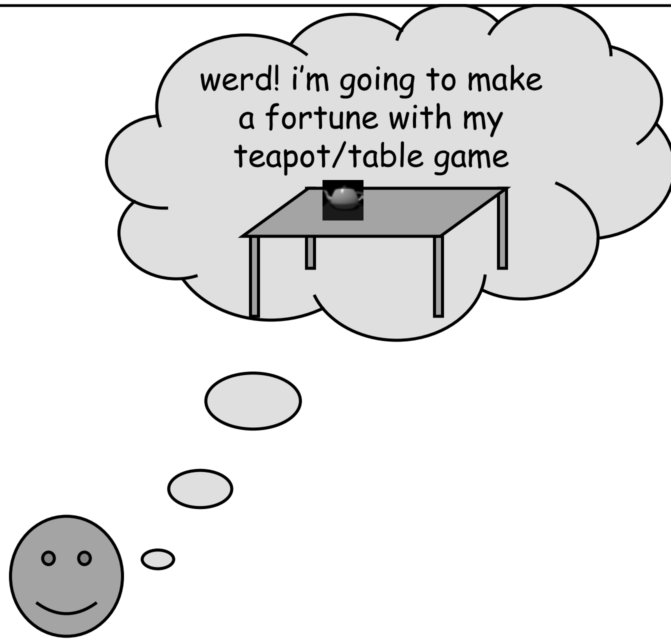
v is transformed by M_2 and then M_1

v is transformed by M_1 and then M_2

last matrix is first applied!

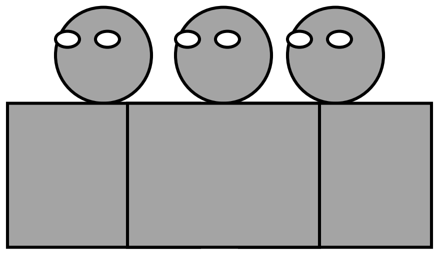
overview

- brief overview of 3D
- modeling transforms & homogenous coordinates DONE
- hierarchical coordinates

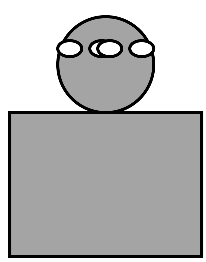


cool action figure

cool action figure



cool action figure

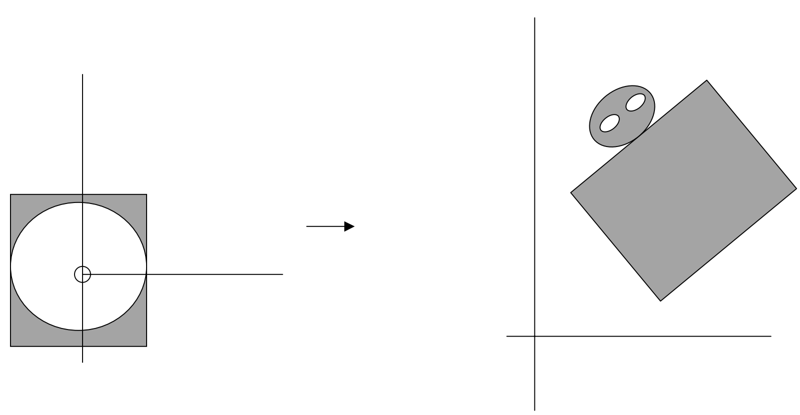


3D model store

receipt
one orange cube ← body
one orange sphere ← head
two white sphere ← eyes

cube has side length one and is centered at the origin
spheres have radius one and are centered at the origin

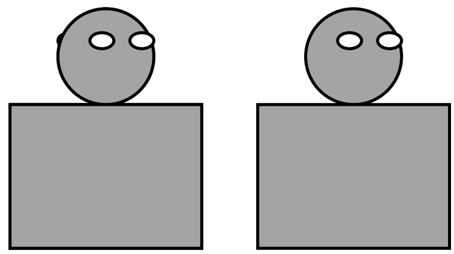
transformations



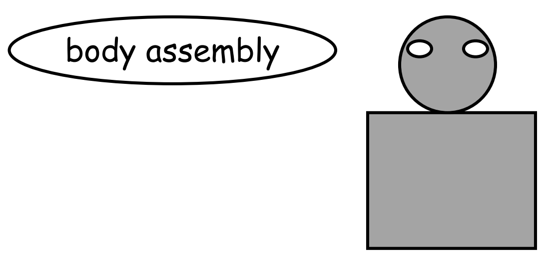
problem

transform components as independently and as one

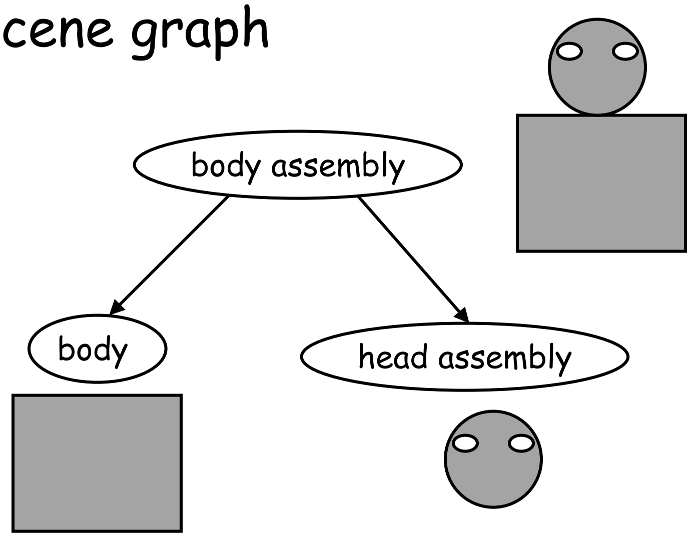
(in real time)



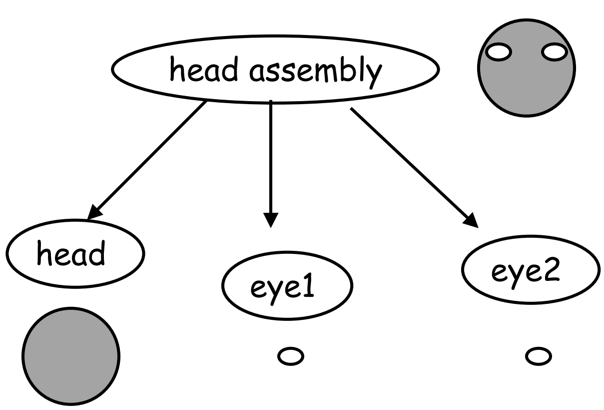
scene graph



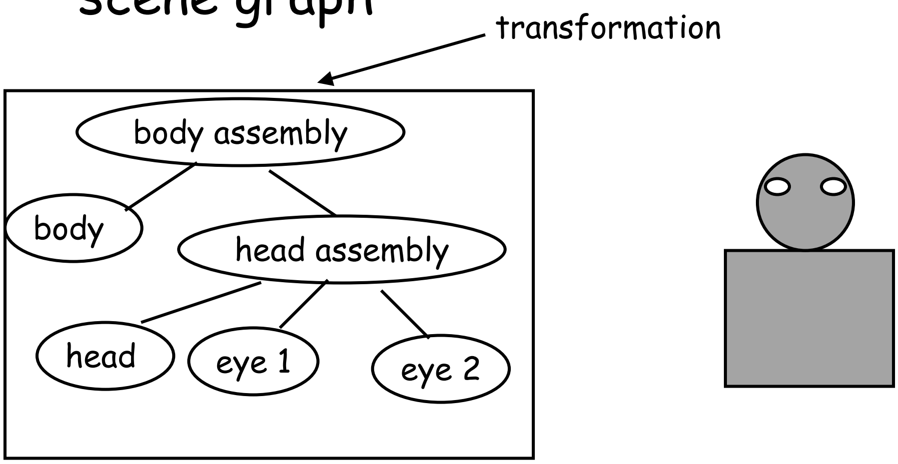
scene graph



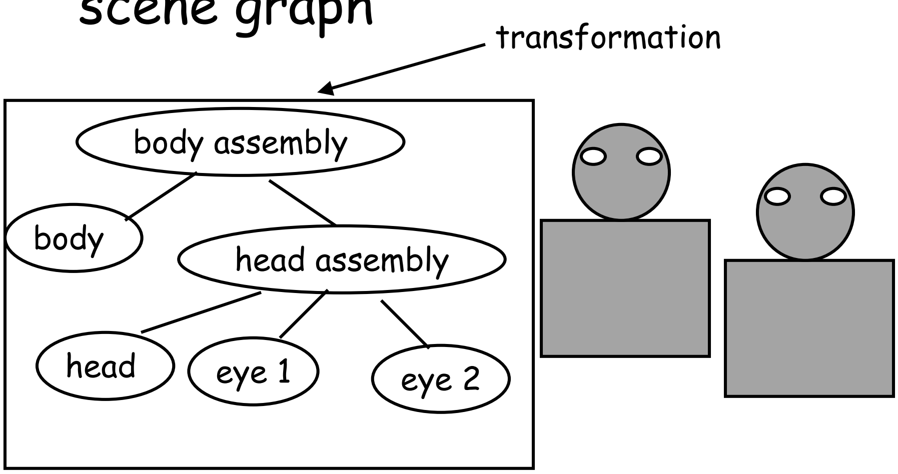
scene graph



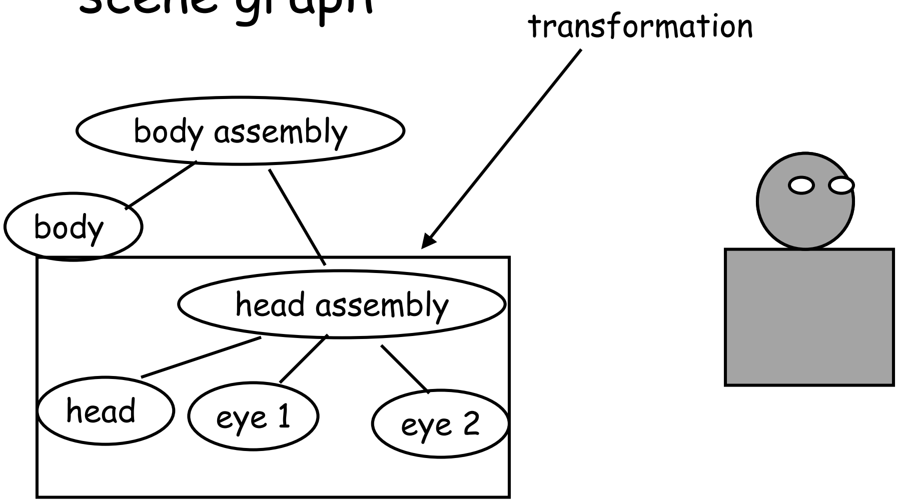
scene graph



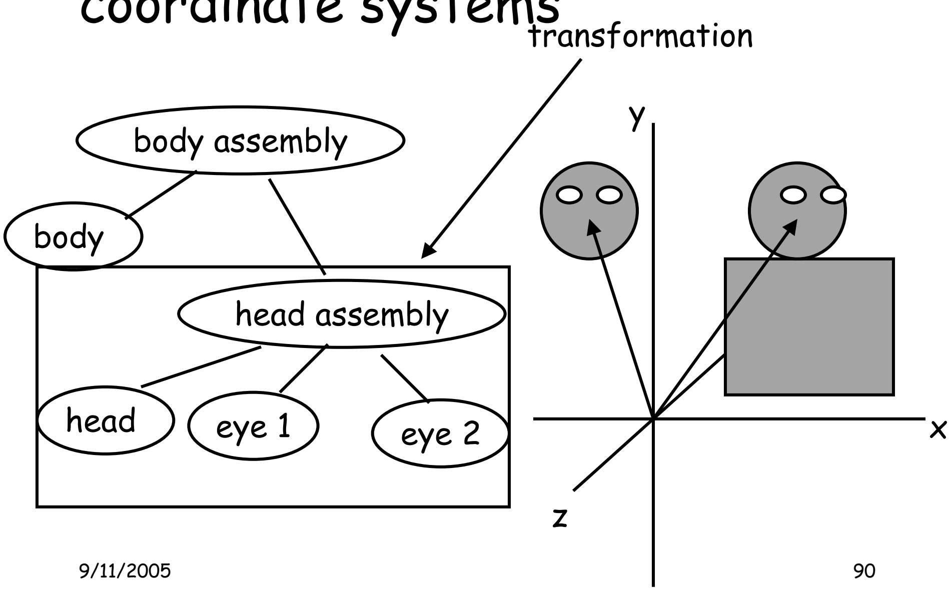
scene graph



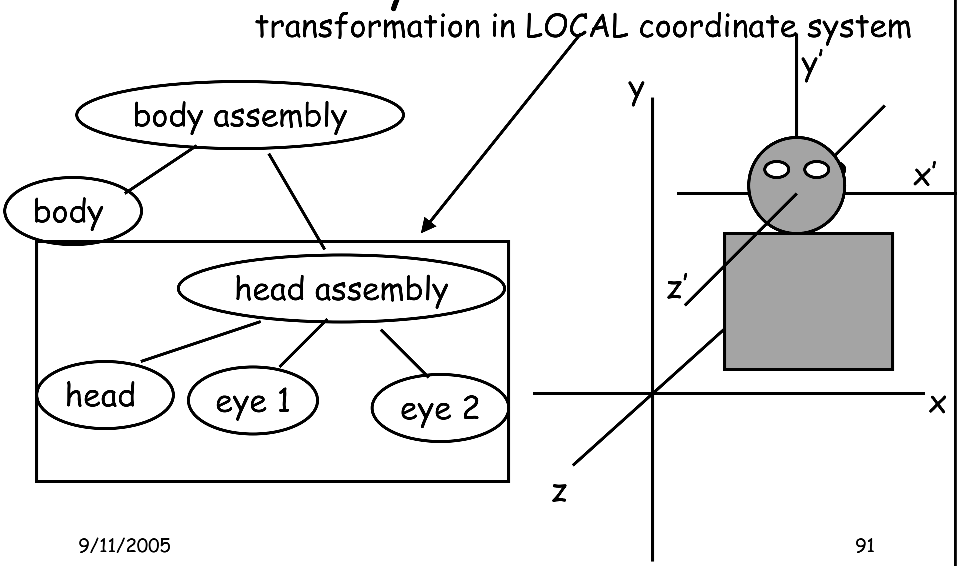
scene graph



coordinate systems

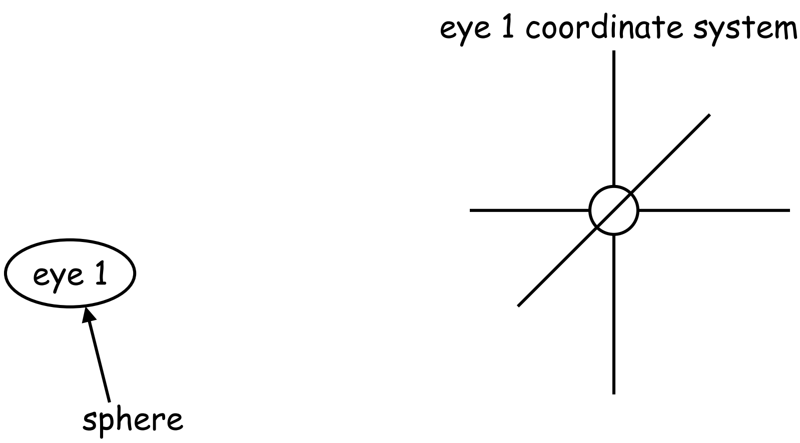


coordinate systems



9/11/2005 91

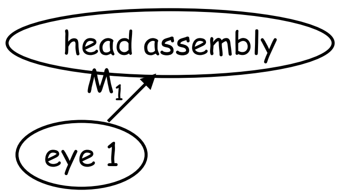
hierarchical coordinates

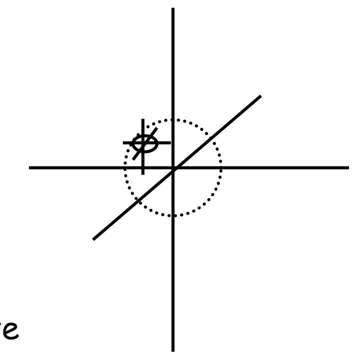


9/11/2005 92

hierarchical coordinates

head assembly coordinate system

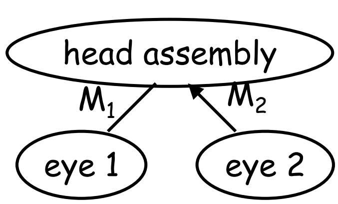


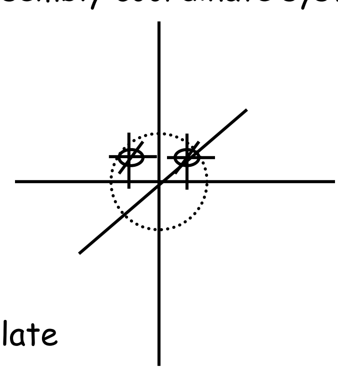


M_1: scale and translate

hierarchical coordinates

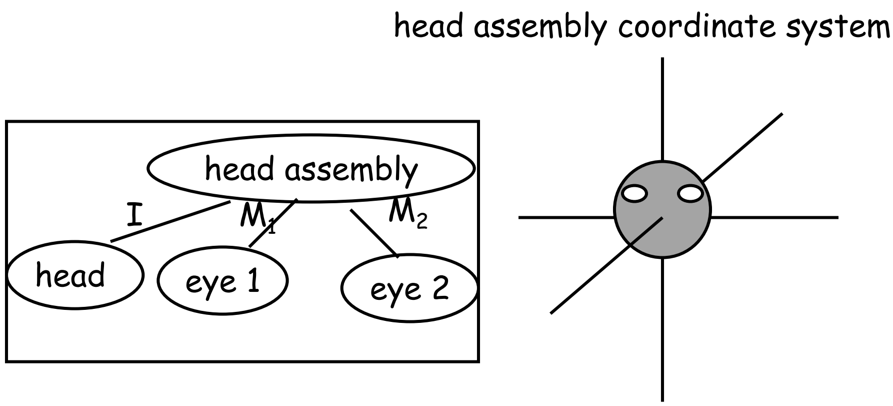
head assembly coordinate system



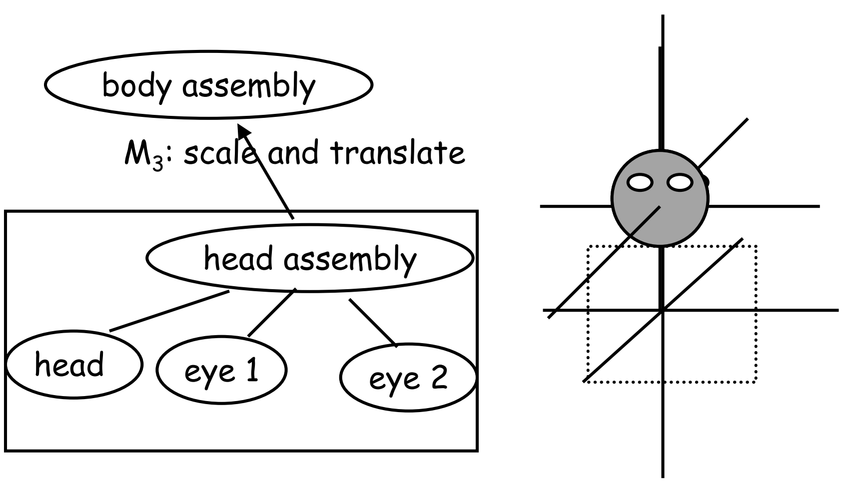


M_2: scale and translate

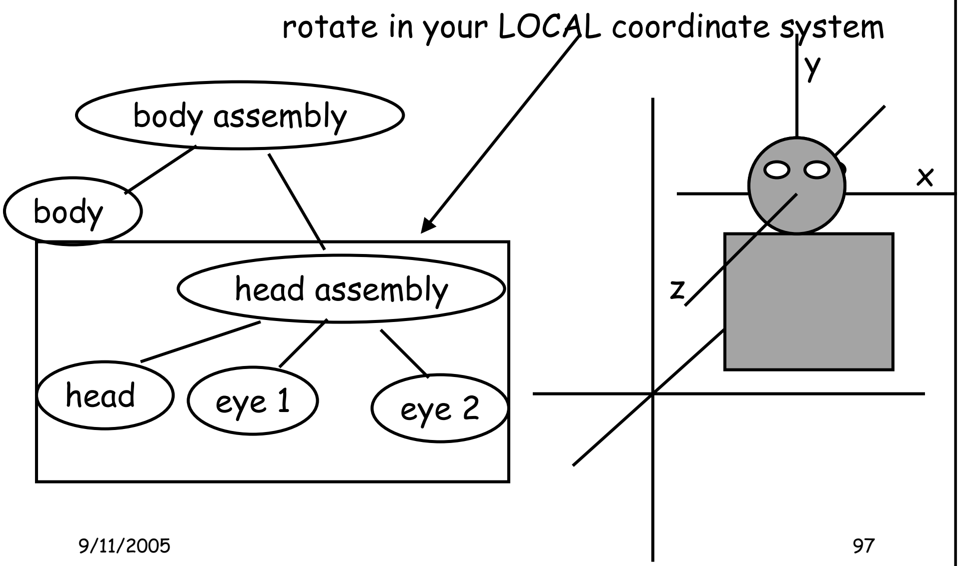
hierarchical coordinates



hierarchical coordinates



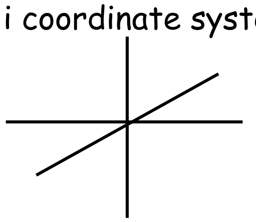
hierarchical coordinates

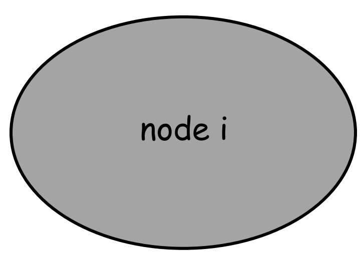


9/11/2005 97

scene graph node

node i coordinate system





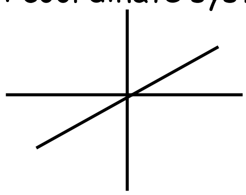
geometric primitives

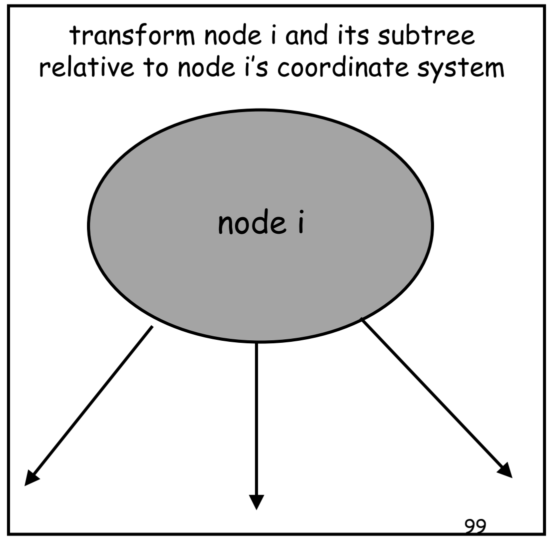
defined in node i's
coordinate system

9/11/2005 98

scene graph node

node i coordinate system





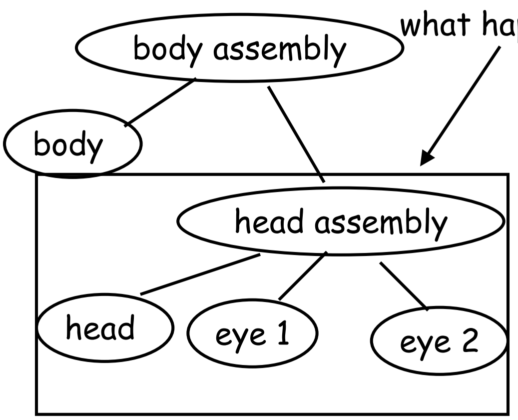
geometric primitives

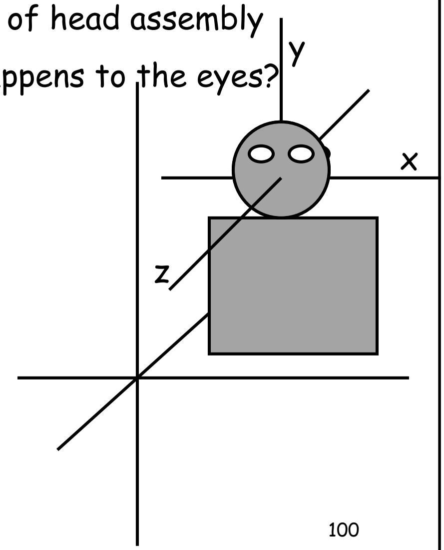
defined in node i's coordinate system

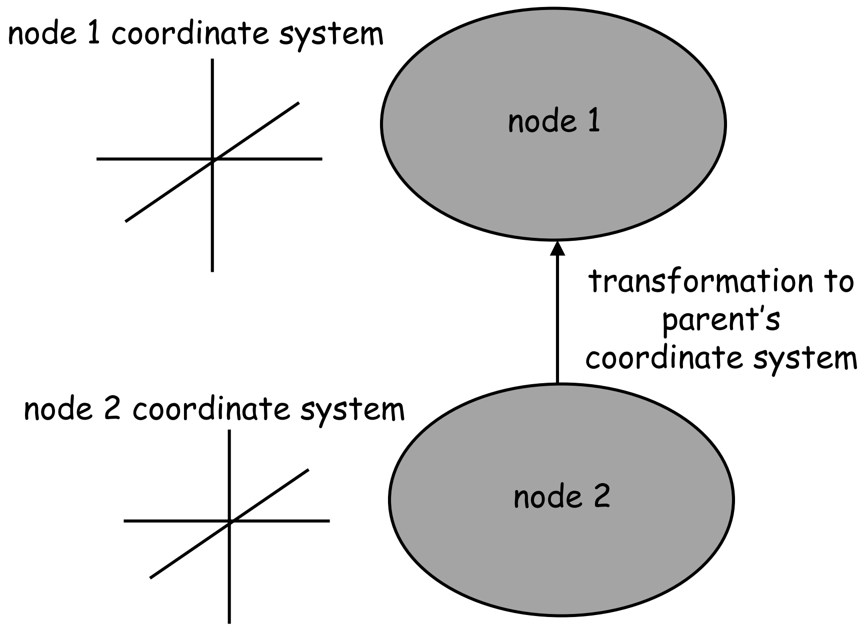
hierarchical coordinates

rotate of head assembly

what happens to the eyes?







hierarchical coordinates

// Left eye

- Transform to head assembly coordinate system:
  - translate(-1,1,0) M_1
  - scale(0.2,0.2,0.2) M_2
- Sphere (radius 1, centered at origin)

hierarchical coordinates

// Right eye

- Transform to head assembly coordinate system:
  - translate(1,1,0) M_4
  - scale(0.2,0.2,0.2) M_3
- Sphere (radius 1, centered at origin)

hierarchical coordinates

// Head

- Sphere (radius 1, centered at origin)

hierarchical coordinates

// Head assembly

- Transform to body assembly coordinate system
  - Translate(0,1/2,0) M_7
  - Scale(.4,.4,.4) M_6
- Local transform
  - Rotate(θ,y-axis) M_5
- Children
  - Left eye
  - Right eye
  - Head

hierarchical coordinates

// Body

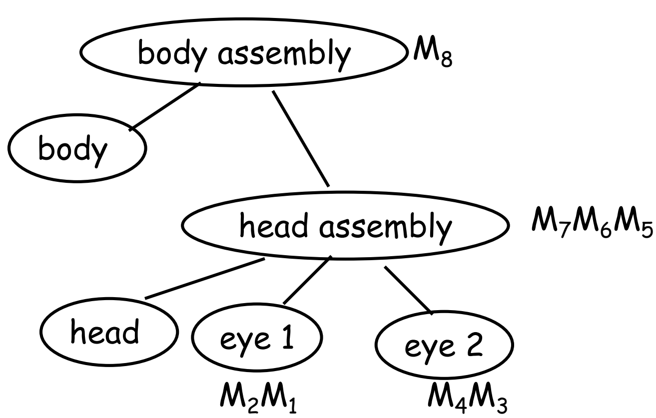
- Cube

hierarchical coordinates

// Body assembly

- Local transform
  - Translate(dx,dy,dz) M_8
- Children
  - Body
  - Head assembly

hierarchical coordinates



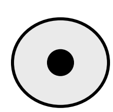
Exercise

What is the composite transform applied to left eye?

What is the composite transform applied to right eye?

Exercise

Suppose we used a texture mapped sphere
for the eye:



How would we modify the scene graph so
that we could roll the eyes?

Exercise: build the scene graph for a snowman that has a head, middle, bottom, two eyes, nose, and mouth.

snowman: what you can use

- spheres: radius 1, centered at origin (color options: white, black, red)
- scale (s,t,u)
- rotate (theta, vector)
- translate (dx,dy,dz)

download rt & snowman.ray

expand snowman.ray to include your scene graph!

render with rt