CS-184: Computer Graphics

Lecture #4: 2D Transformations

Prof. James O'Brien
University of California, Berkeley

V2008-F-04-1.0

Today

- 2D Transformations
  - "Primitive" Operations
    - Scale, Rotate, Shear, Flip, Translate
  - Homogenous Coordinates
  - SVD
  - Start thinking about rotations...

Introduction

- Transformation:

  An operation that changes one configuration into another

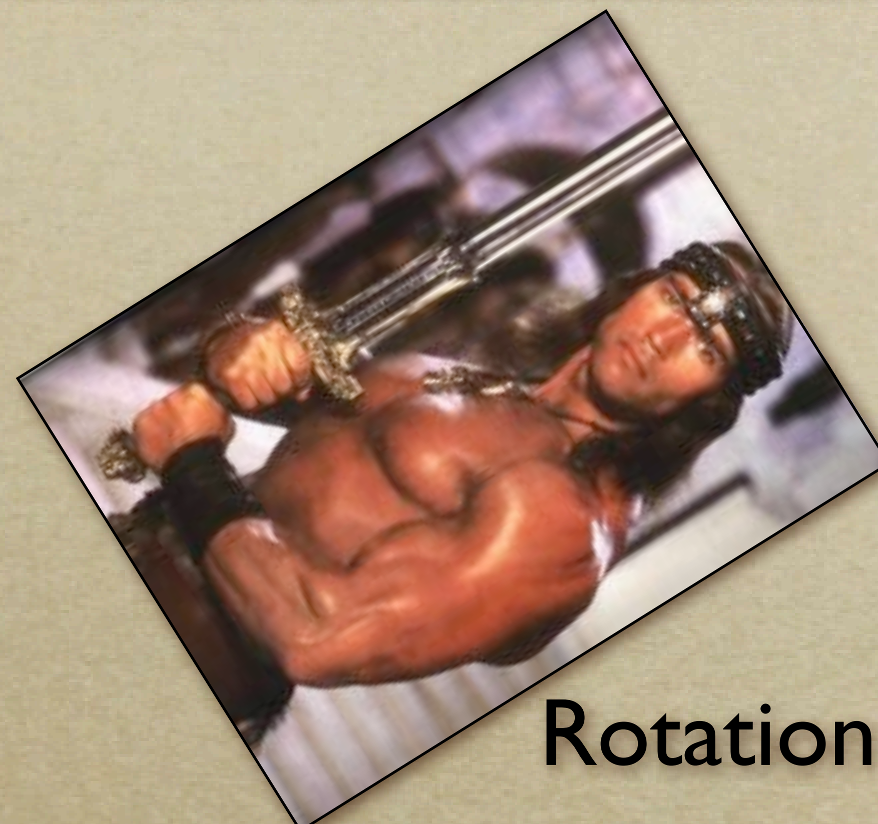
- For images, shapes, *etc*.

  A geometric transformation maps positions that define the object to other positions

  Linear transformation means the transformation is defined by a linear function... which is what matrices are good for.

Some Examples

Original



Rotation

Uniform Scale

Nonuniform Scale

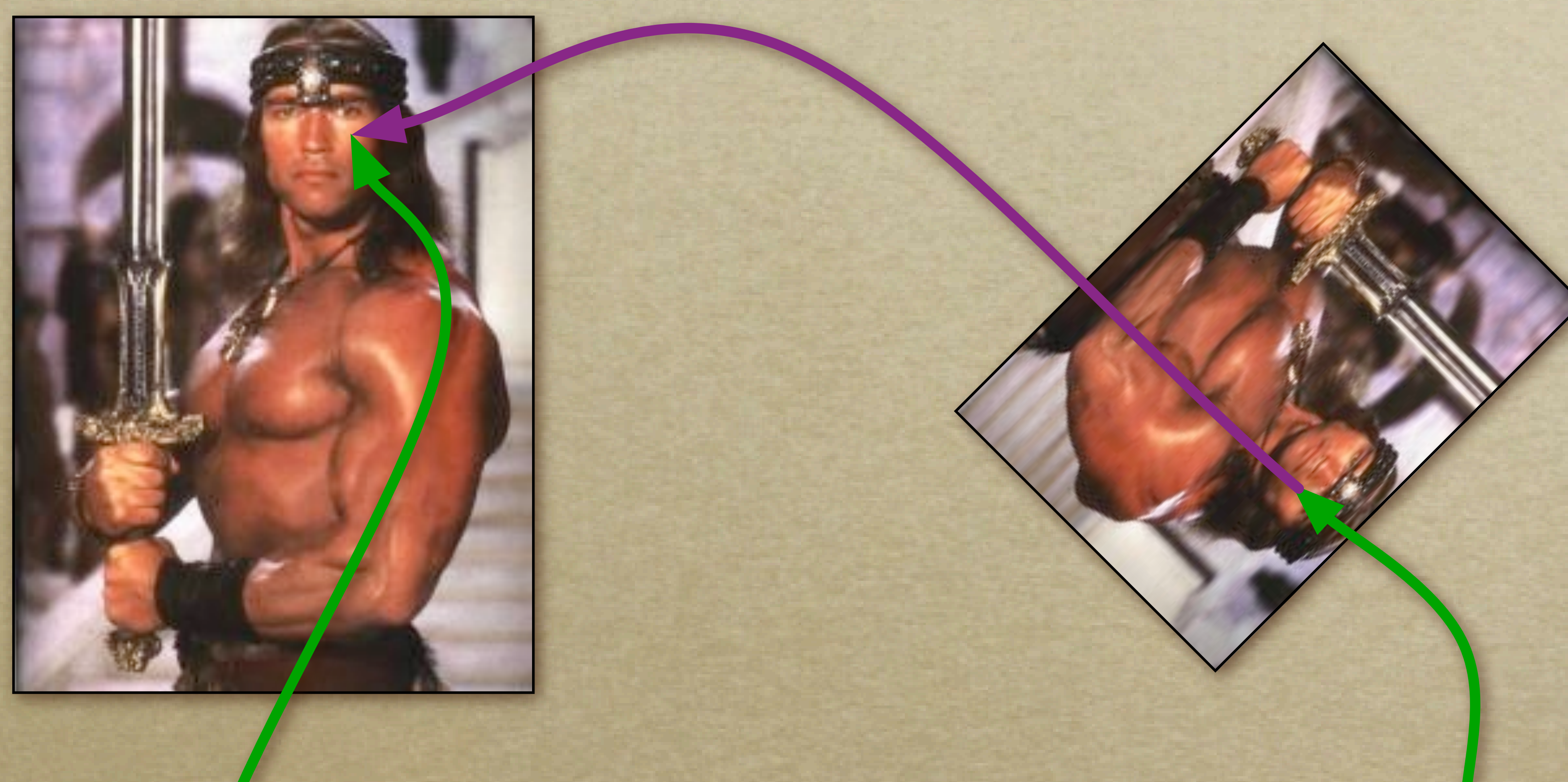
Shear

Images from *Conan The Destroyer*, 1984

4

Mapping Function

$$f(x) = x \text{ in old image}$$



$$c(x) = [195, 120, 58]$$

$$c'x = c(f(x))$$

Linear -vs- Nonlinear

Nonlinear (swirl)

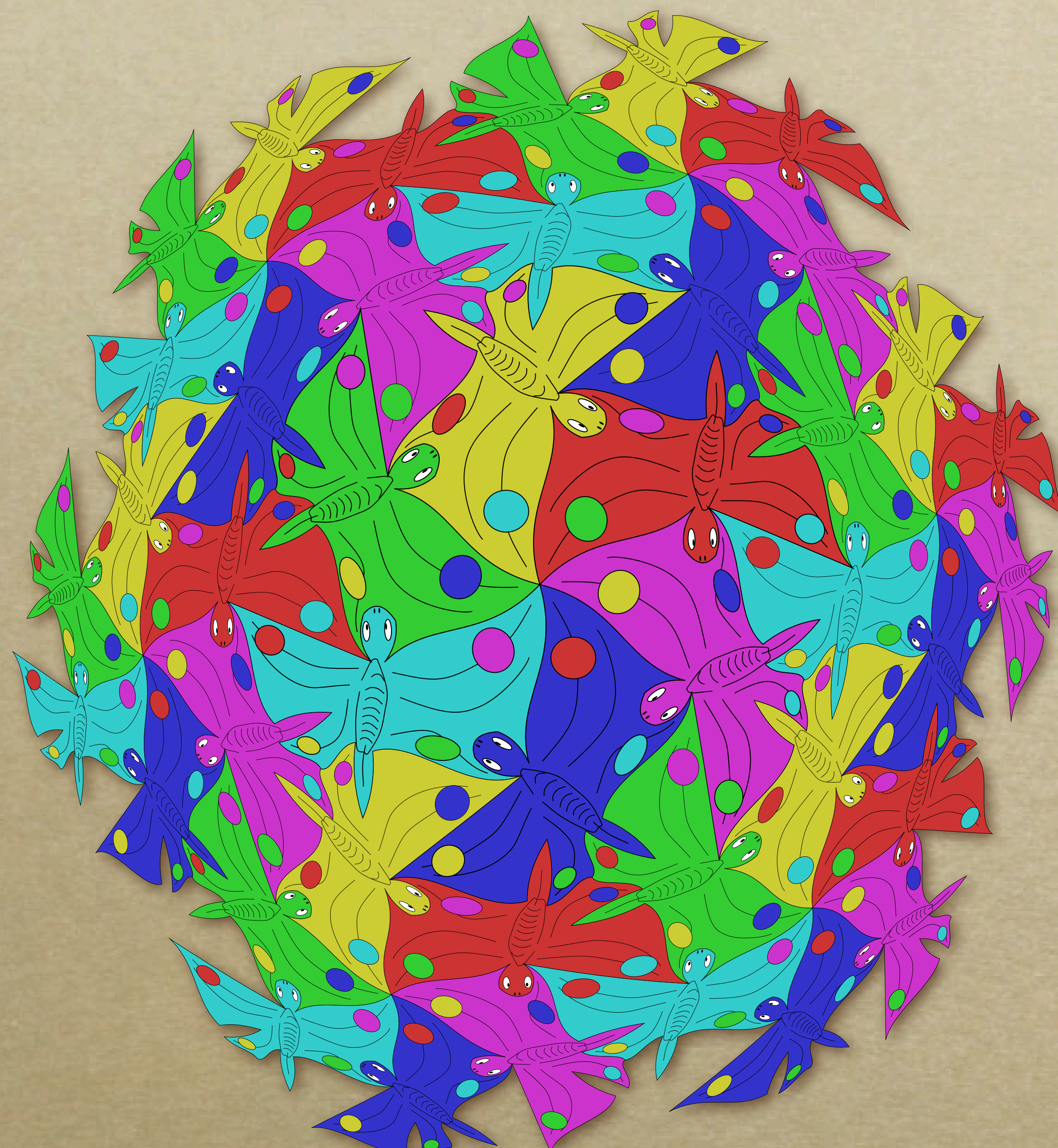
Linear (shear)

Geometric -vs- Color Space

Color Space Transform
(edge finding)

Linear Geometric
(flip)

Instancing



RHW

M.C. Escher, from Ghostscript 8.0 Distribution

8

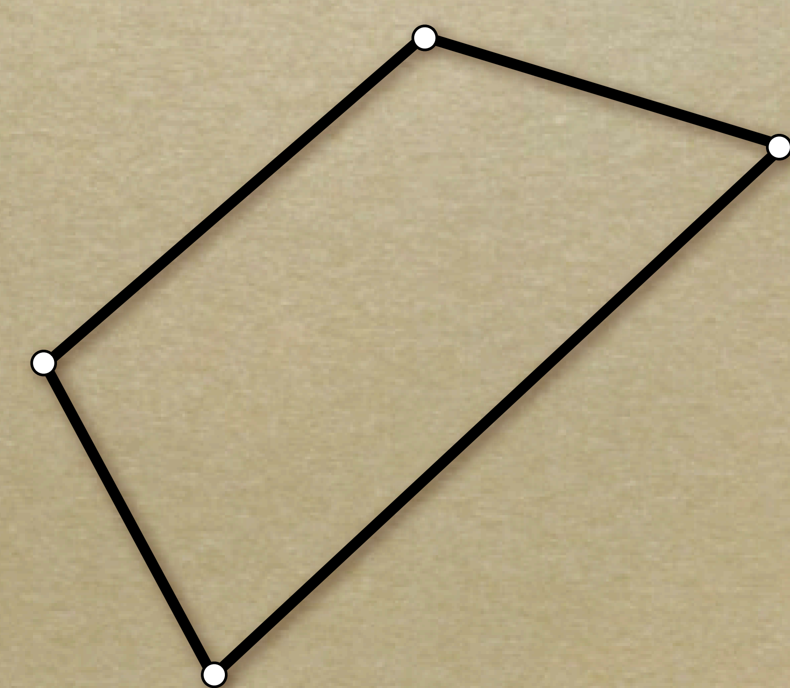
Instancing

- Reuse geometric descriptions
- Saves memory


RHW

Linear is Linear

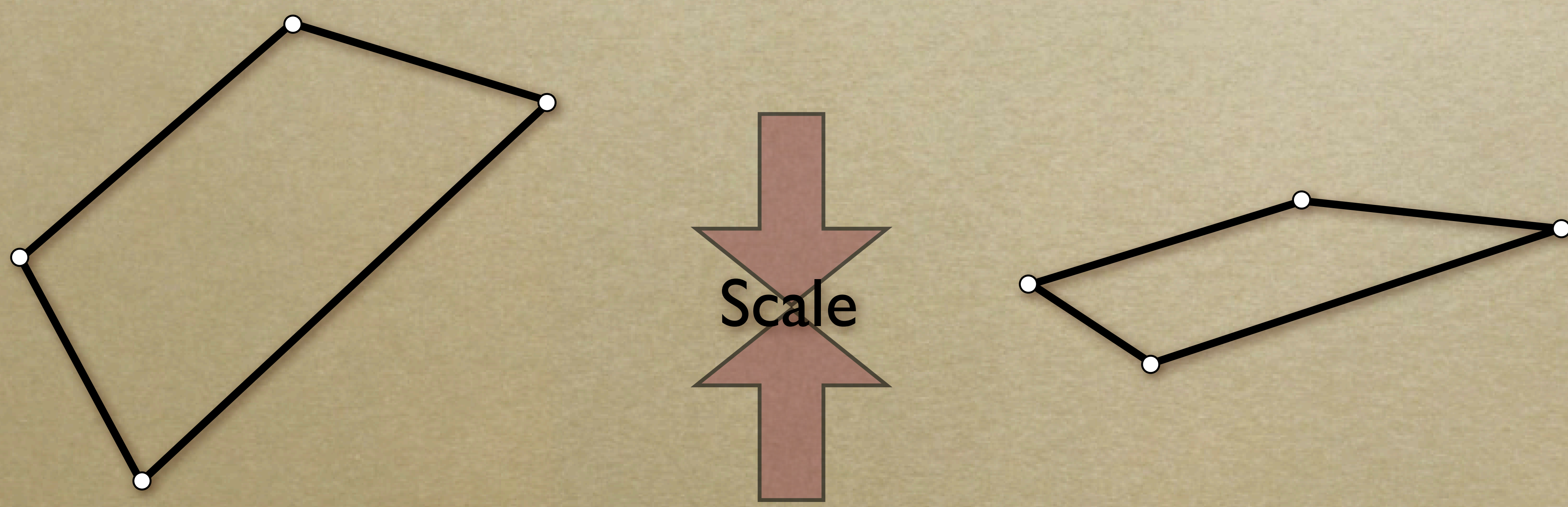
- Polygons defined by points
- Edges defined by interpolation between two points
- Interior defined by interpolation between all points
- *Linear* interpolation



10

Linear is Linear

- Composing two linear function is still linear
- Transform polygon by transforming vertices



Linear is Linear

- Composing two linear function is still linear
- Transform polygon by transforming vertices

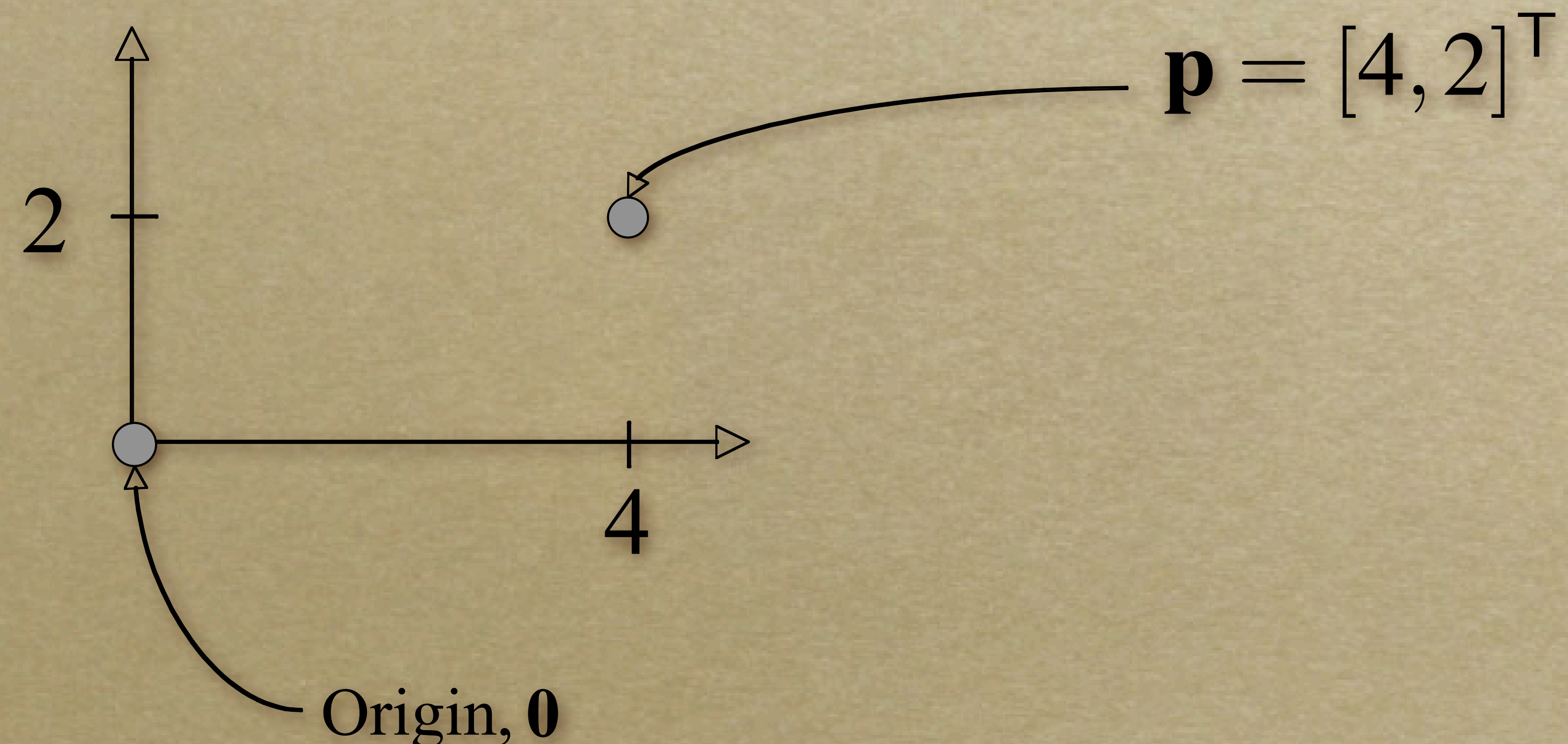
$$f(x) = a + bx \qquad g(f) = c + df$$

$$g(x) = c + df(x) = c + ad + bdx$$

$$g(x) = a' + b'x$$

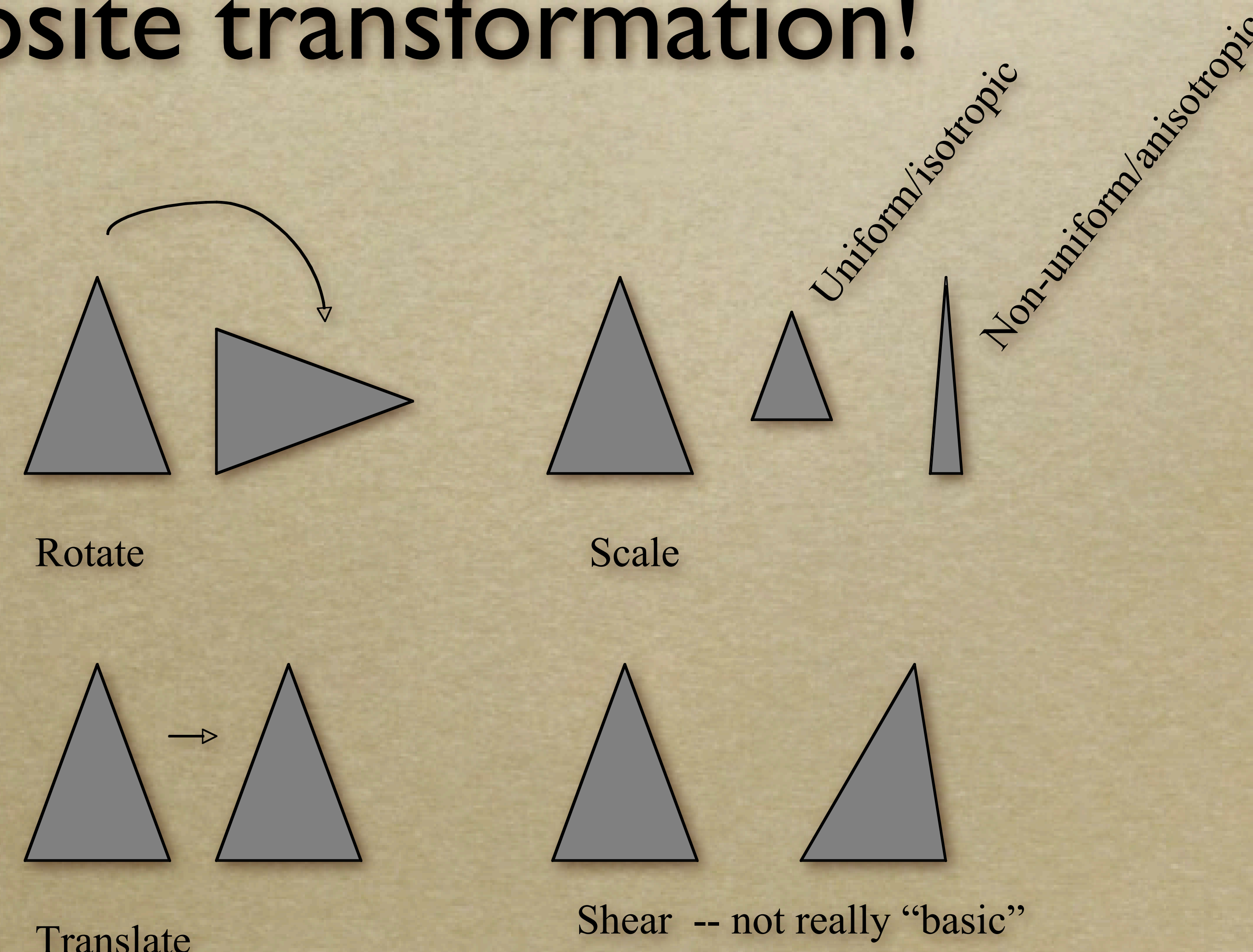
Points in Space

- Represent point in space by vector in R^n
  - Relative to some origin!
  - Relative to some coordinate axes!
- Later we'll add something extra...



Basic Transformations

- Basic transforms are: rotate, scale, and translate
- Shear is a composite transformation!



14

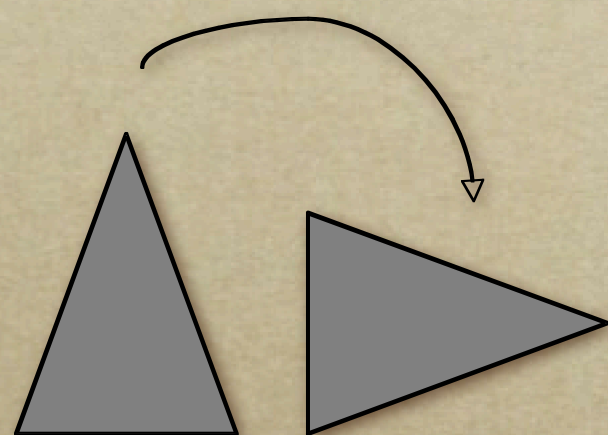
Linear Functions in 2D

$$x' = f(x,y) = c_1 + c_2 x + c_3 y$$
$$y' = f(x,y) = d_1 + d_2 x + d_3 y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} t_x \\ t_y \end{bmatrix} + \begin{bmatrix} M_{xx} & M_{xy} \\ M_{yx} & M_{yy} \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

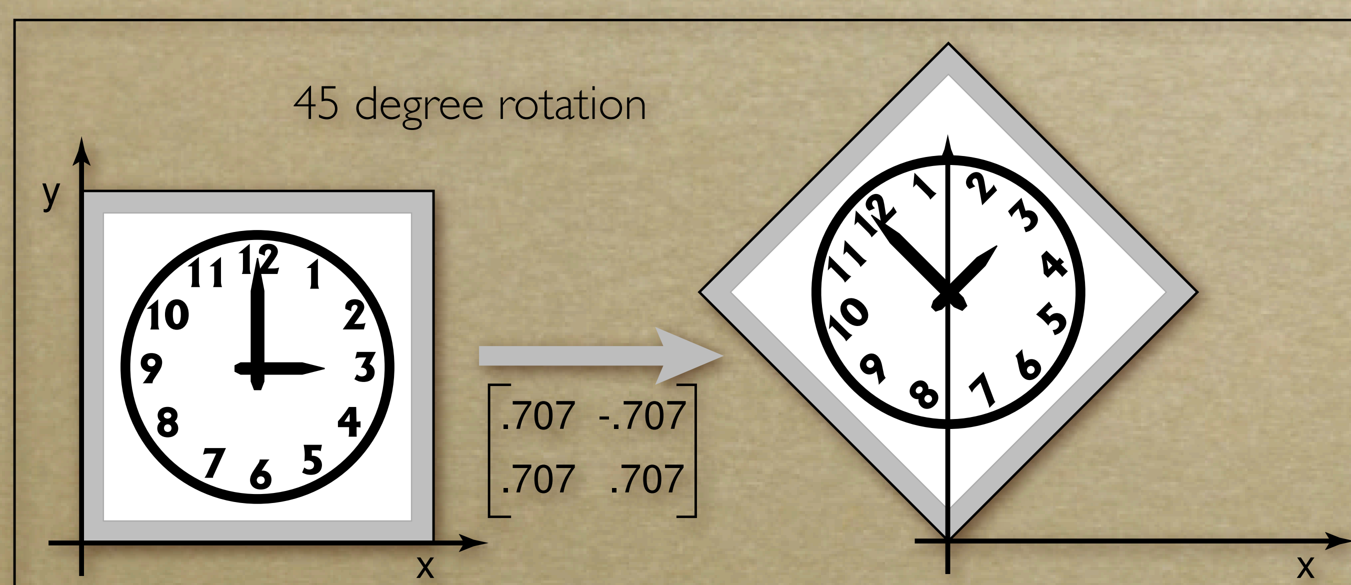
$$\mathbf{x}' = \mathbf{t} + \mathbf{M} \cdot \mathbf{x}$$

Rotations



Rotate

$$\mathbf{p}' = \begin{bmatrix} \mathrm{Cos}(\theta) & -Sin(\theta) \\ Sin(\theta) & \mathrm{Cos}(\theta) \end{bmatrix} \mathbf{p}$$



Rotations

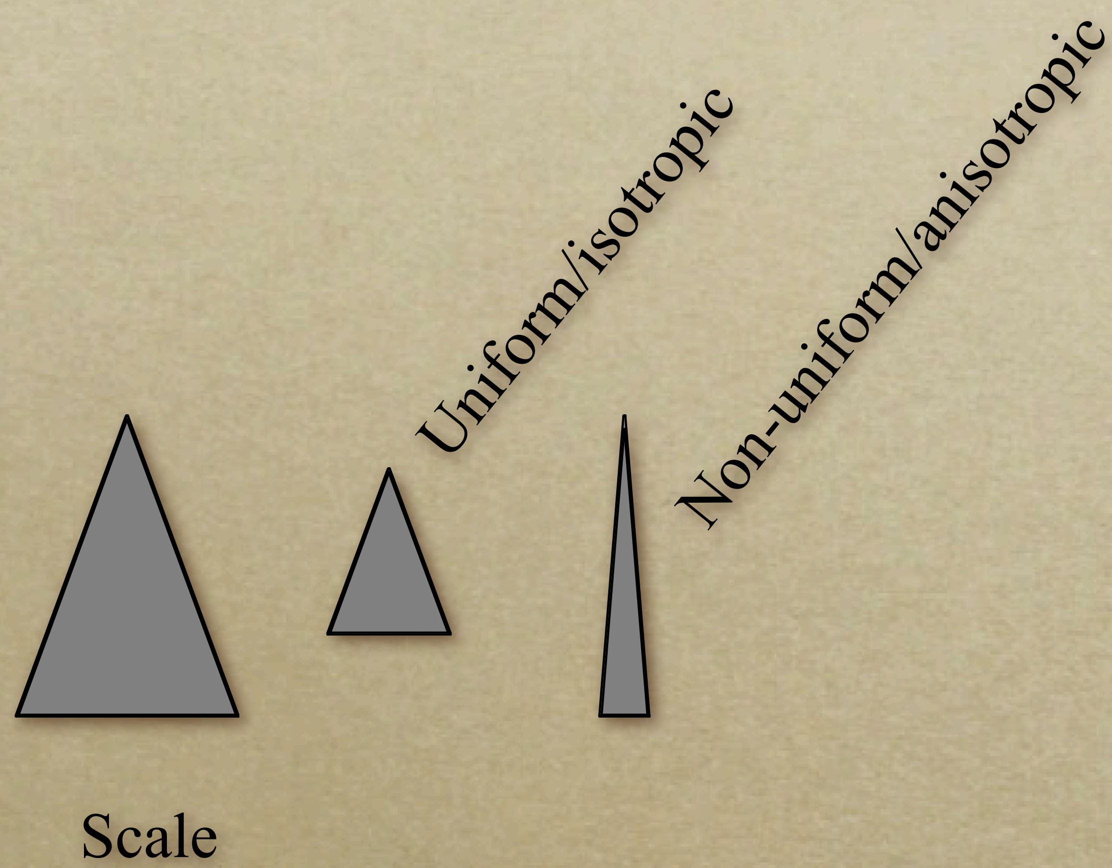
- Rotations are positive counter-clockwise
- Consistent w/ right-hand rule
- Don't be different...
- Note:
  - rotate by zero degrees give identity
  - rotations are modulo 360 (or 2π)

17

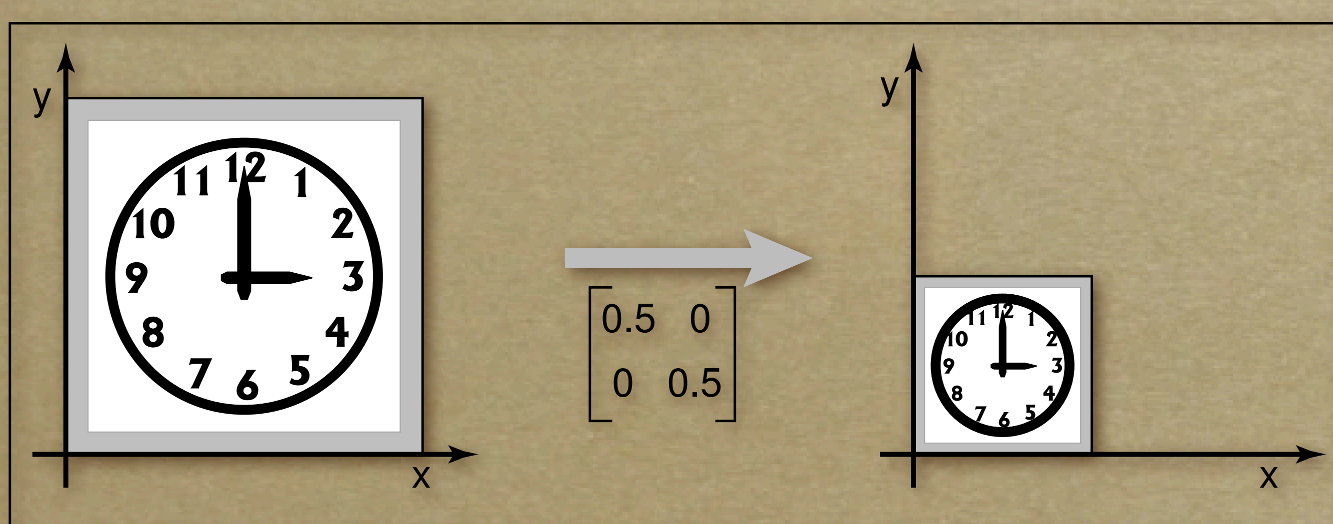
Rotations

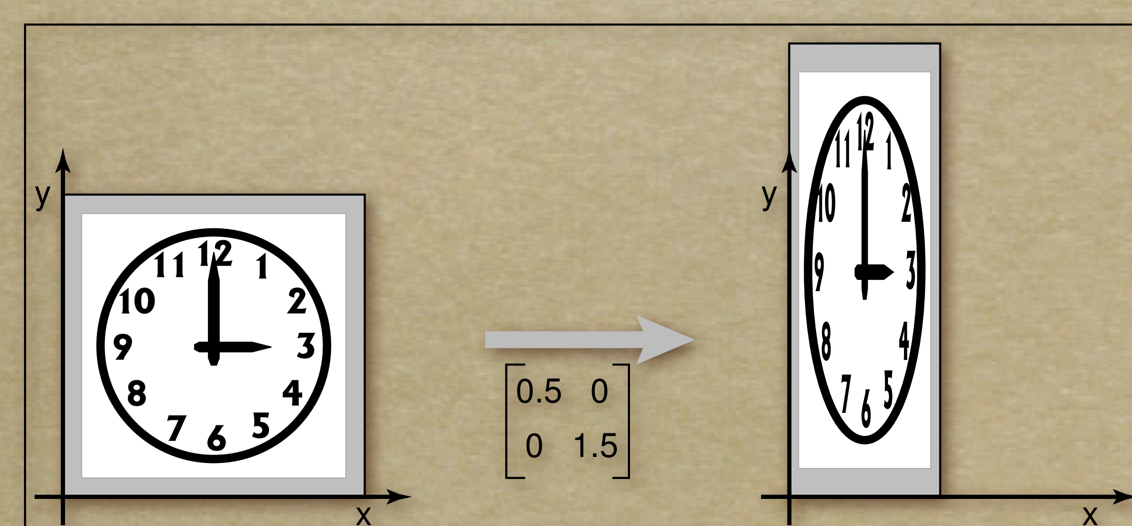
- Preserve lengths and distance to origin
- Rotation matrices are orthonormal
- $\mathrm{Det}(\mathbf{R}) = 1 \neq -1$
- In 2D rotations commute...
  - But in 3D they won't!

Scales



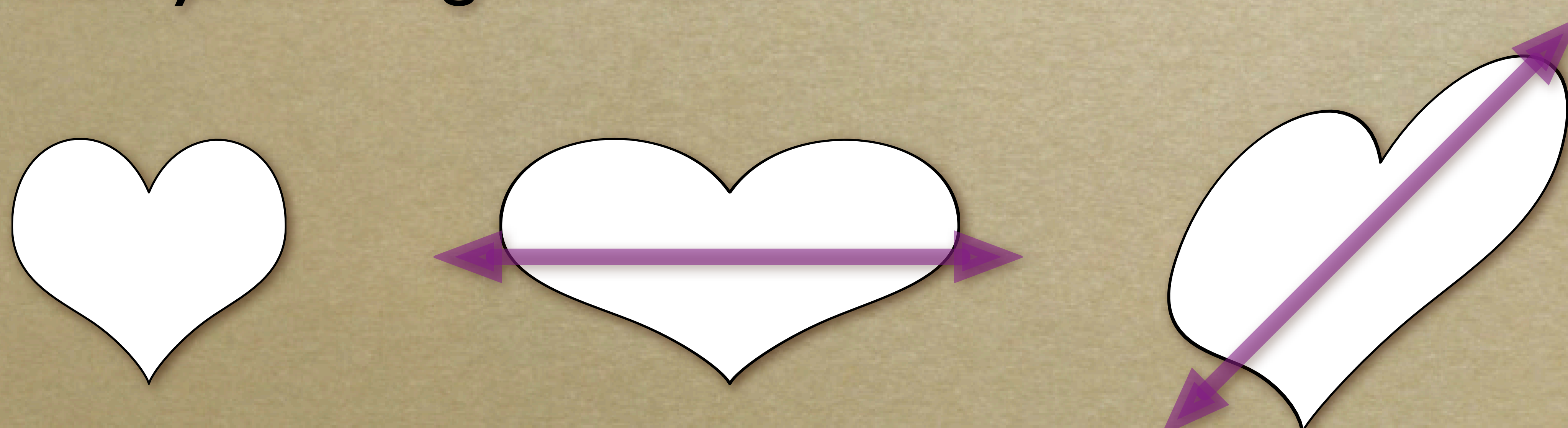
$$\mathbf{p}' = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \mathbf{p}$$





Scales

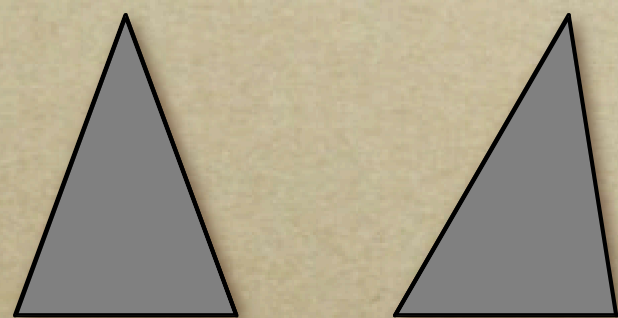
- Diagonal matrices
  - Diagonal parts are scale in X and scale in Y directions
  - Negative values flip
  - Two negatives make a positive (180 deg. rotation)
  - Really, axis-aligned scales



Not axis-aligned...

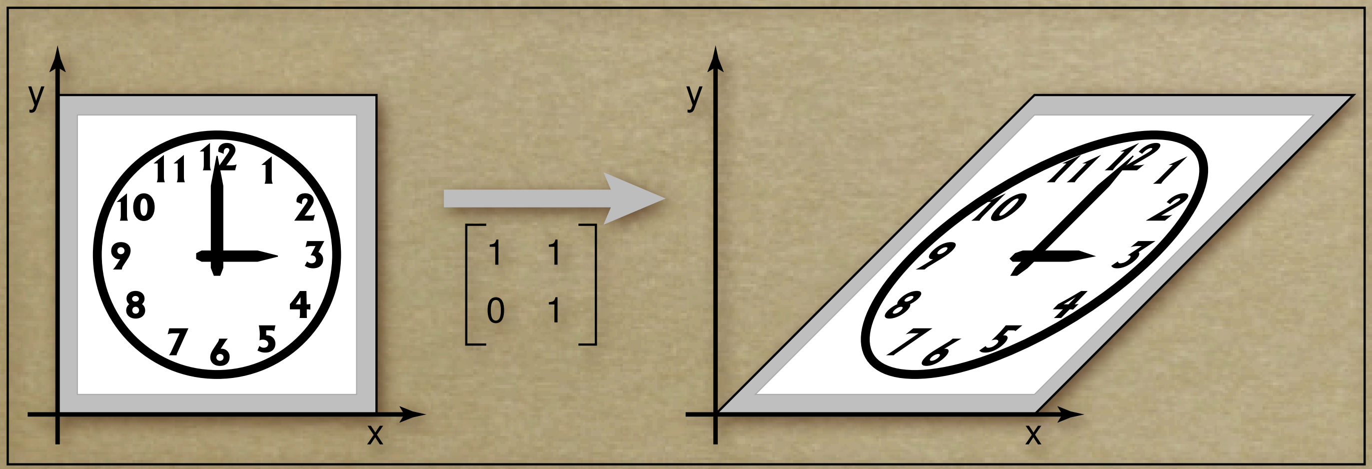
20

Shears



Shear

$$\mathbf{p}' = \begin{bmatrix} 1 & H_{yx} \\ H_{xy} & 1 \end{bmatrix} \mathbf{p}$$

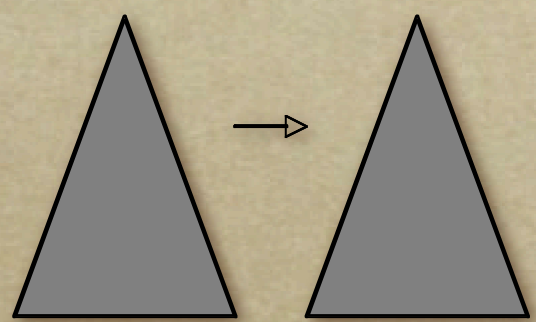


Shears

- Shears are not really primitive transforms
- Related to non-axis-aligned scales
- More shortly.....

Translation

- This is the not-so-useful way:



Translate

$$\mathbf{p}' = \mathbf{p} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Note that its not like the others.

Arbitrary Matrices

- For everything but translations we have:

$$\mathbf{x}' = \mathbf{A} \cdot \mathbf{x}$$

- Soon, translations will be assimilated as well

- What does an arbitrary matrix mean?

24

Singular Value Decomposition

- For any matrix, $\mathbf{A}$, we can write SVD:

$$\mathbf{A} = \mathbf{QSR}^{\mathsf{T}}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are orthonormal and $\mathbf{S}$ is diagonal

- Can also write Polar Decomposition

$$\mathbf{A} = \mathbf{QRSR}^{\mathsf{T}}$$

not the same $\mathbf{Q}$

where $\mathbf{Q}$ is still orthonormal

25

Decomposing Matrices

- We can force **Q** and **R** to have Det=1 so they are rotations
- Any matrix is now:
  - Rotation:Rotation:Scale:Rotation
  - See, shear is just a mix of rotations and scales

26

Composition

- Matrix multiplication composites matrices

$$\mathbf{p}' = \mathbf{BAp}$$

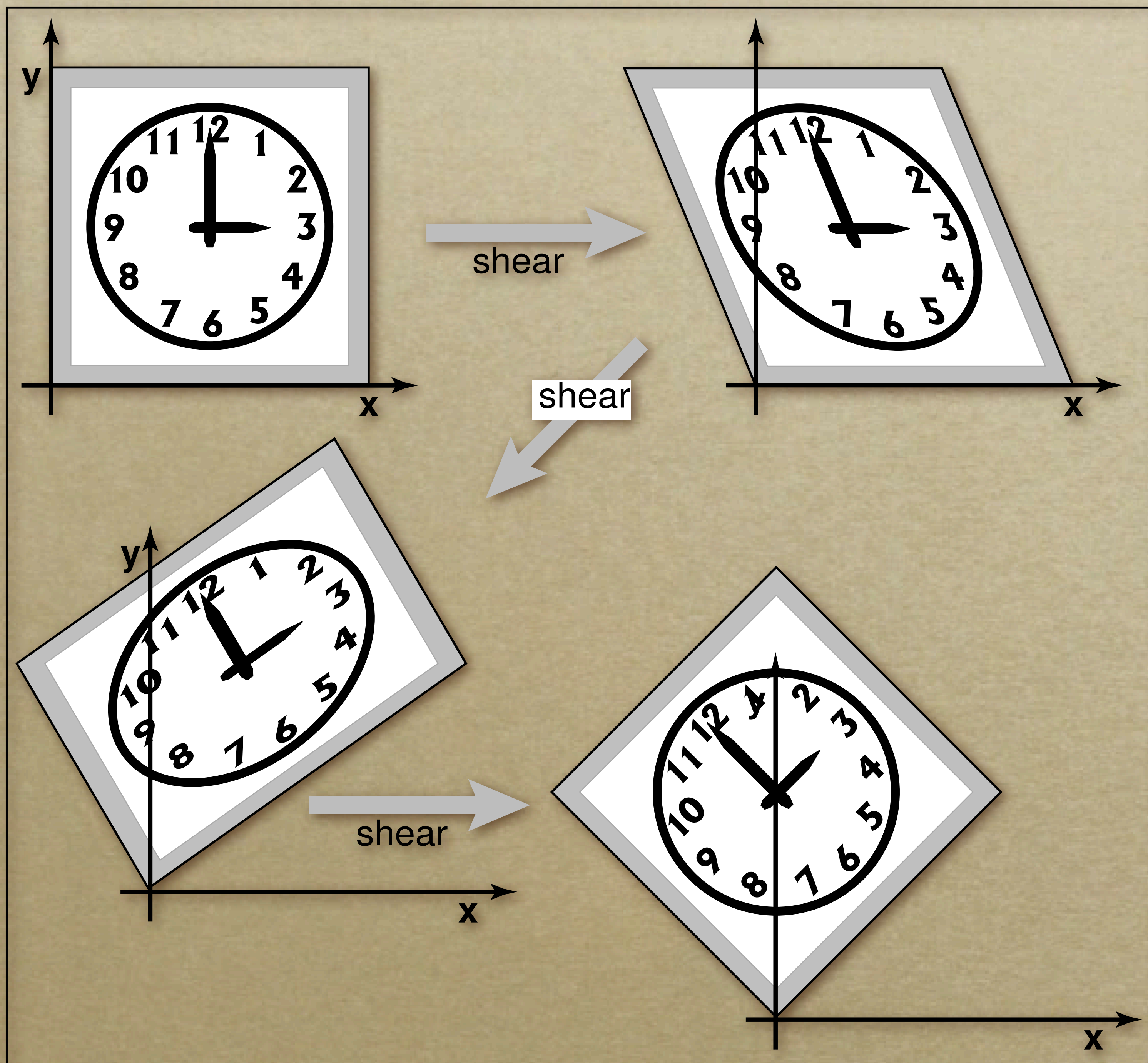
"Apply $\mathbf{A}$ to $\mathbf{p}$ and then apply $\mathbf{B}$ to that result."

$$\mathbf{p}' = \mathbf{B}(\mathbf{Ap}) = (\mathbf{BA})\mathbf{p} = \mathbf{Cp}$$

- Several translations composted to one
- Translations still left out...

$$\mathbf{p}' = \mathbf{B}(\mathbf{Ap} + \mathbf{t}) = \mathbf{BAp} + \mathbf{Bt} = \mathbf{Cp} + \mathbf{u}$$

Composition



Transformations built up from others

SVD builds from scale and rotations

Also build other ways

i.e. 45 deg rotation built from shears

Homogeneous Coordiantes

- Move to one higher dimensional space
  - Append a 1 at the end of the vectors

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \end{bmatrix} \qquad \tilde{\mathbf{p}} = \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}$$

  - For *directions* the extra coordinate is a zero

Homogeneous Translation

$$\tilde{\mathbf{p}}' = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}$$

$$\tilde{\mathbf{p}}' = \tilde{\mathbf{A}}\tilde{\mathbf{p}}$$

The tildes are for clarity to distinguish homogenized from non-homogenized vectors.

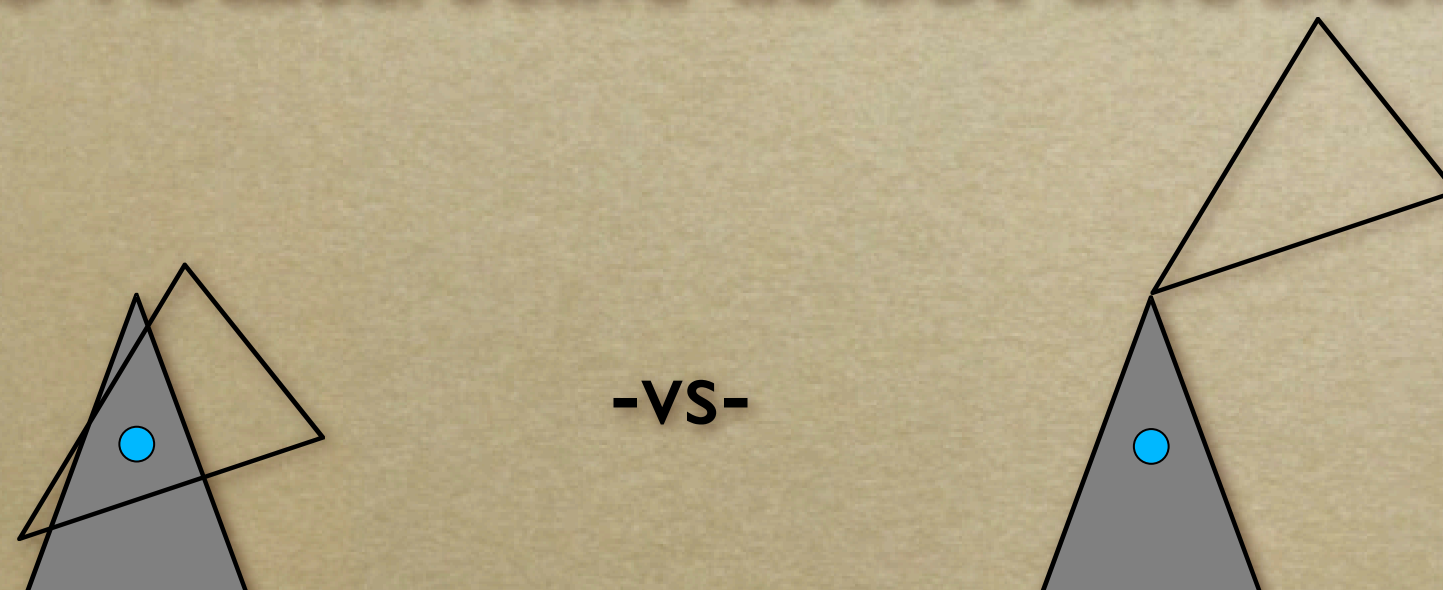
Homogeneous Others

$$\widetilde{\mathbf{A}} = \begin{bmatrix} & \mathbf{A} & \begin{matrix} 0 \\ 0 \end{matrix} \\ 0 & 0 & 1 \end{bmatrix}$$

Now everything looks the same...
Hence the term "homogenized!"

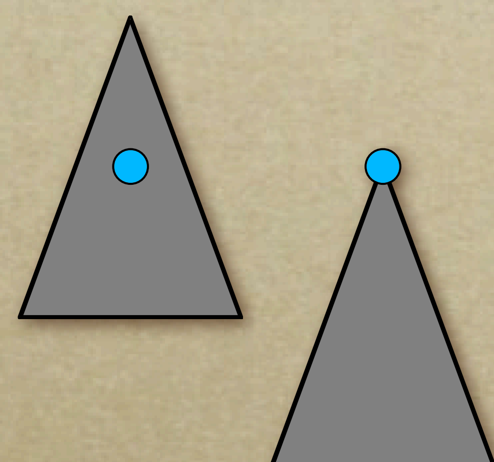
Compositing Matrices

- Rotations and scales always about the origin
- How to rotate/scale about another point?



Rotate About Arb. Point

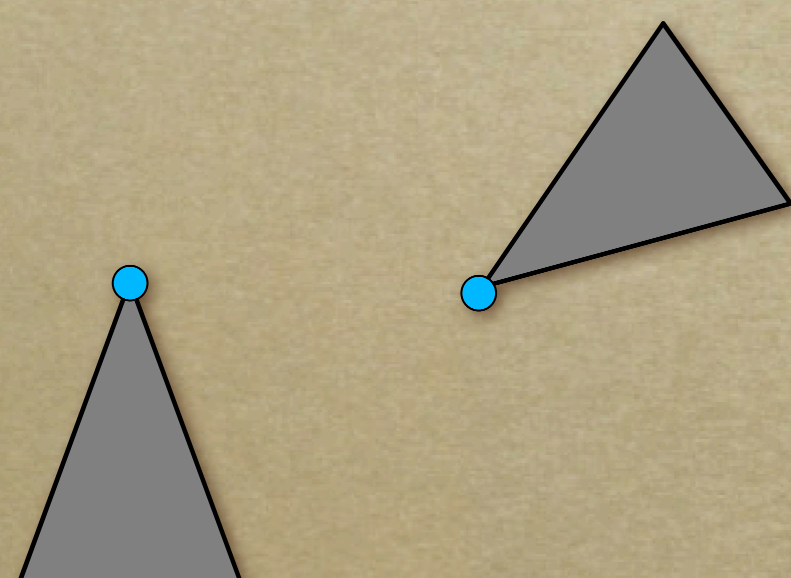
- Step 1: Translate point to origin



Translate (-**C**)

Rotate About Arb. Point

- Step 1: Translate point to origin
- Step 2: Rotate as desired



Translate (-**C**)

Rotate (θ)

34

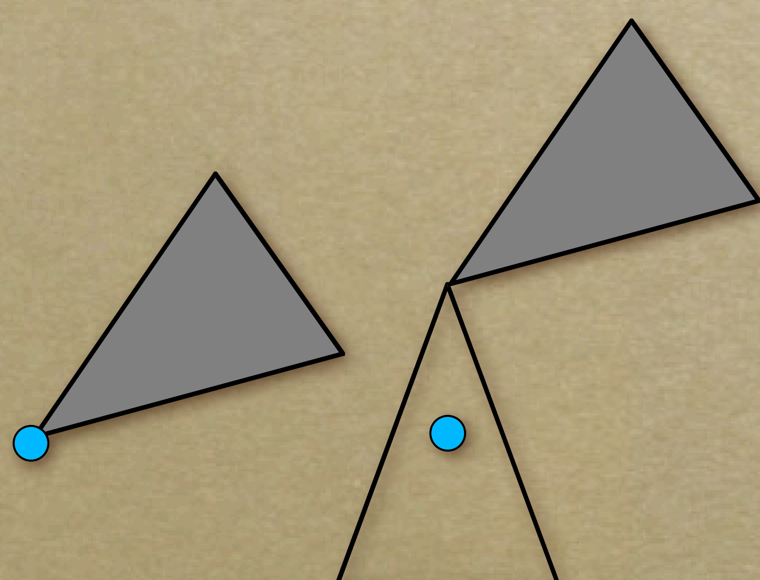
Rotate About Arb. Point

- Step 1: Translate point to origin
- Step 2: Rotate as desired
- Step 3: Put back where it was

Translate (-**C**)

Rotate (θ)

Translate (**C**)



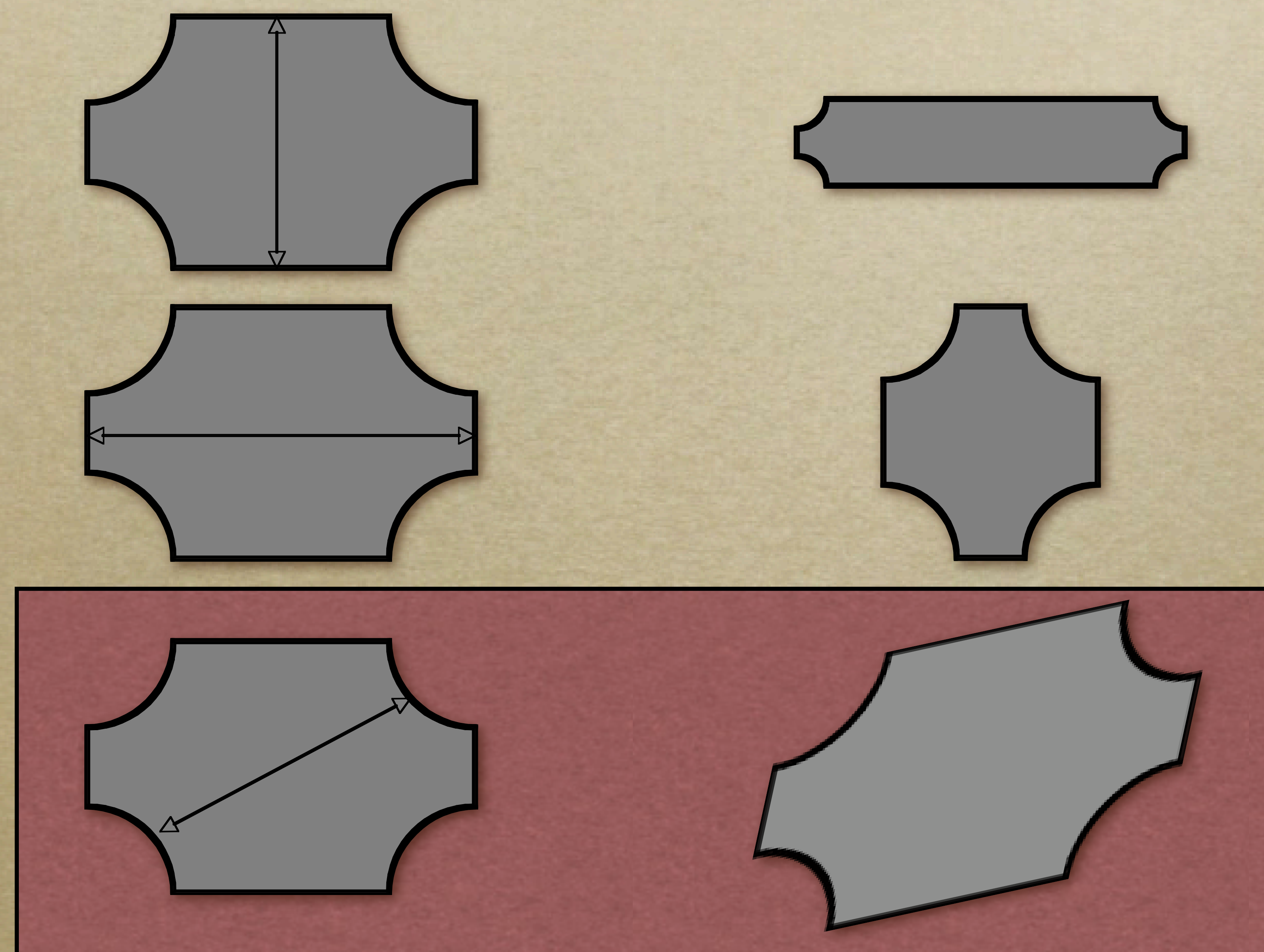
$$\tilde{\mathbf{p}}' = -\mathbf{T}\,\mathbf{R}\mathbf{T}\tilde{\mathbf{p}} = \mathbf{A}\tilde{\mathbf{p}}$$

Don’t negate the 1…

35

Scale About Arb. Axis

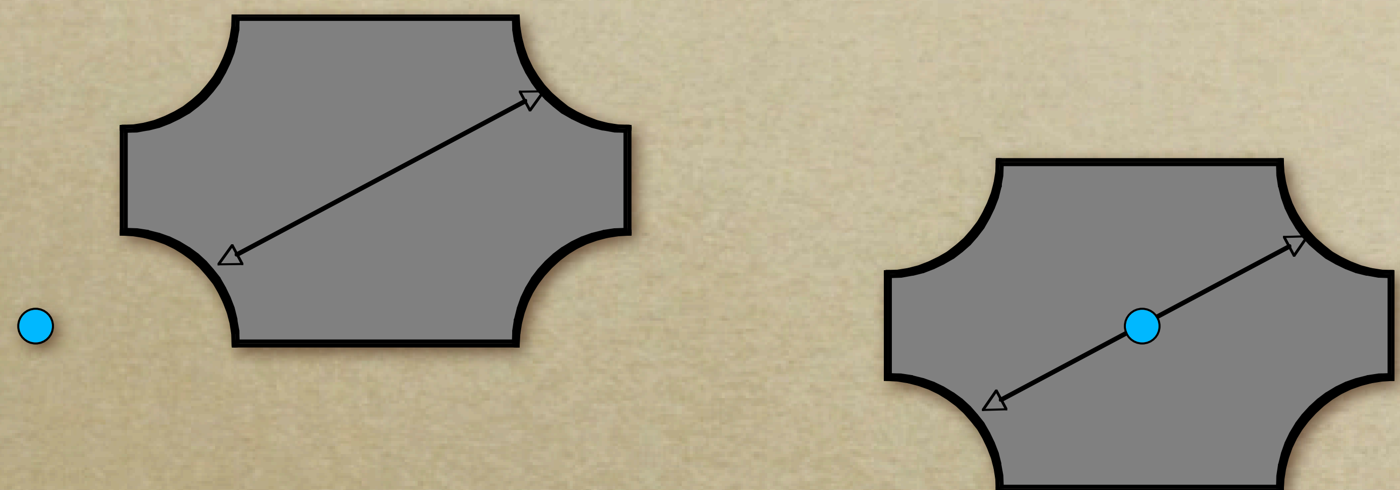
- Diagonal matrices scale about coordinate axes only:



Not axis-aligned

Scale About Arb. Axis

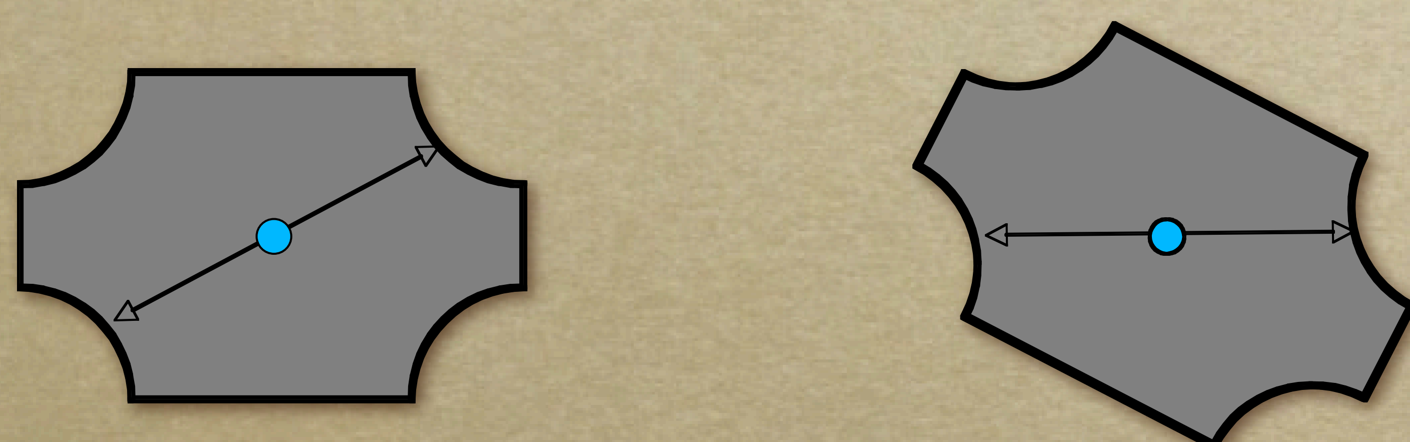
- Step 1: Translate axis to origin



37

Scale About Arb. Axis

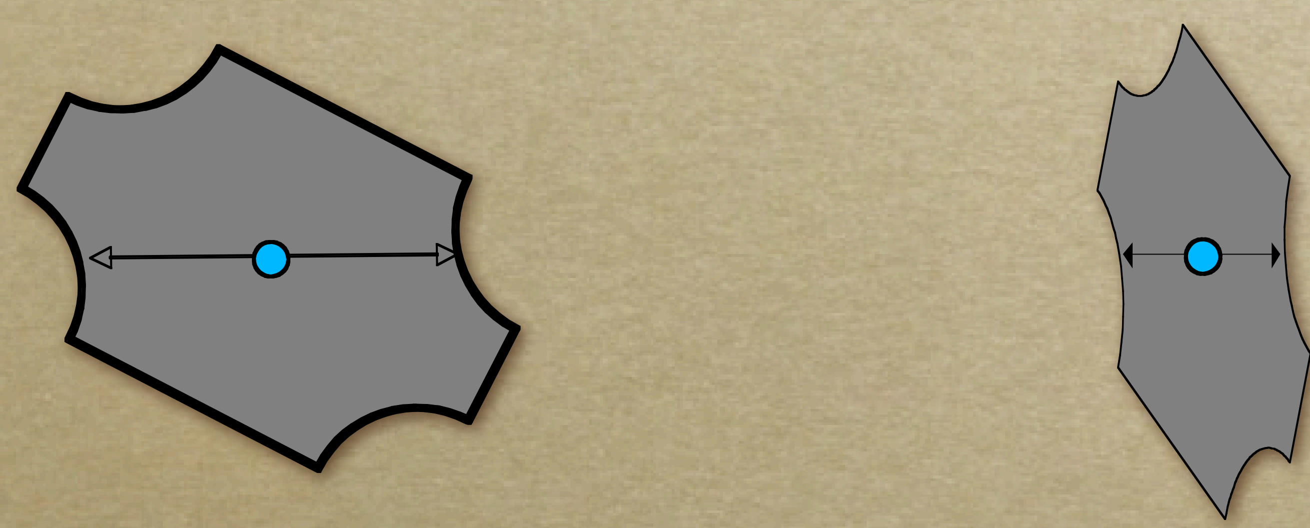
- Step 1: Translate axis to origin
- Step 2: Rotate axis to align with one of the coordinate axes



38

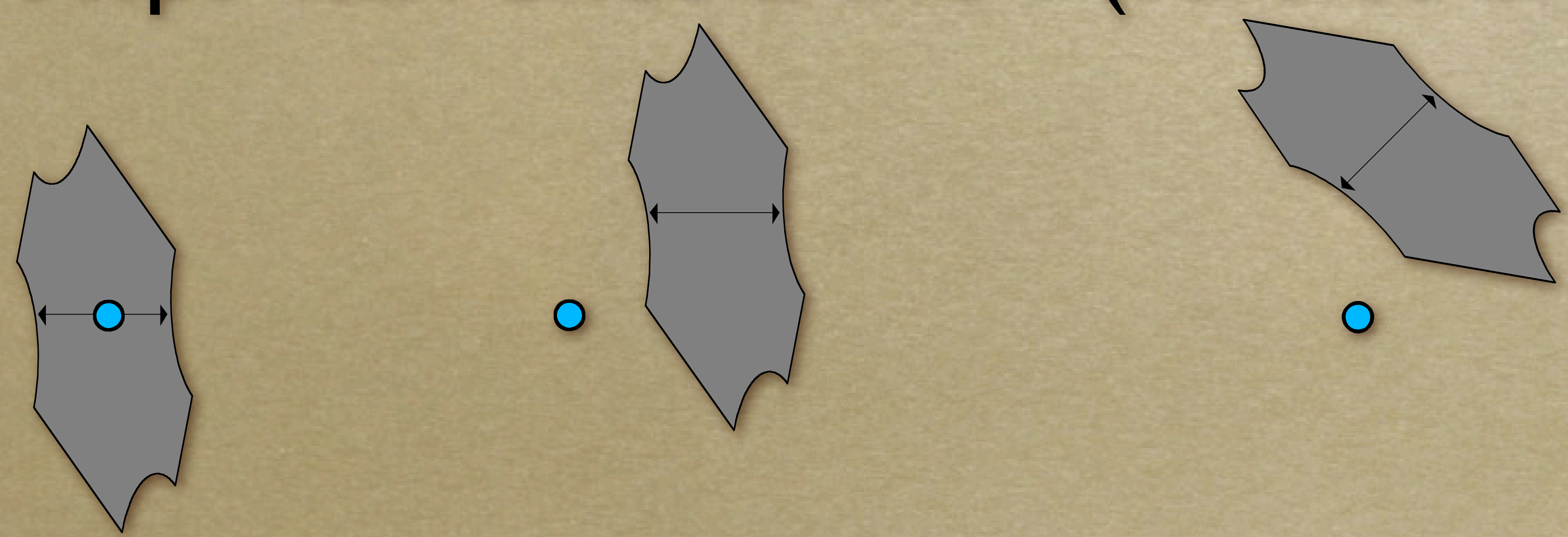
Scale About Arb. Axis

- Step 1: Translate axis to origin
- Step 2: Rotate axis to align with one of the coordinate axes
- Step 3: Scale as desired



Scale About Arb. Axis

- Step 1: Translate axis to origin
- Step 2: Rotate axis to align with one of the coordinate axes
- Step 3: Scale as desired
- Steps 4&5: Undo 2 and 1 (reverse order)



Order Matters!

- The order that matrices appear in matters

$$\mathbf{A} \cdot \mathbf{B} \neq \mathbf{BA}$$

- Some special cases work, but they are special
- But matrices are associative

$$(\mathbf{A} \cdot \mathbf{B}) \cdot \mathbf{C} = \mathbf{A} \cdot (\mathbf{B} \cdot \mathbf{C})$$

- Think about efficiency when you have many points to transform...

Matrix Inverses

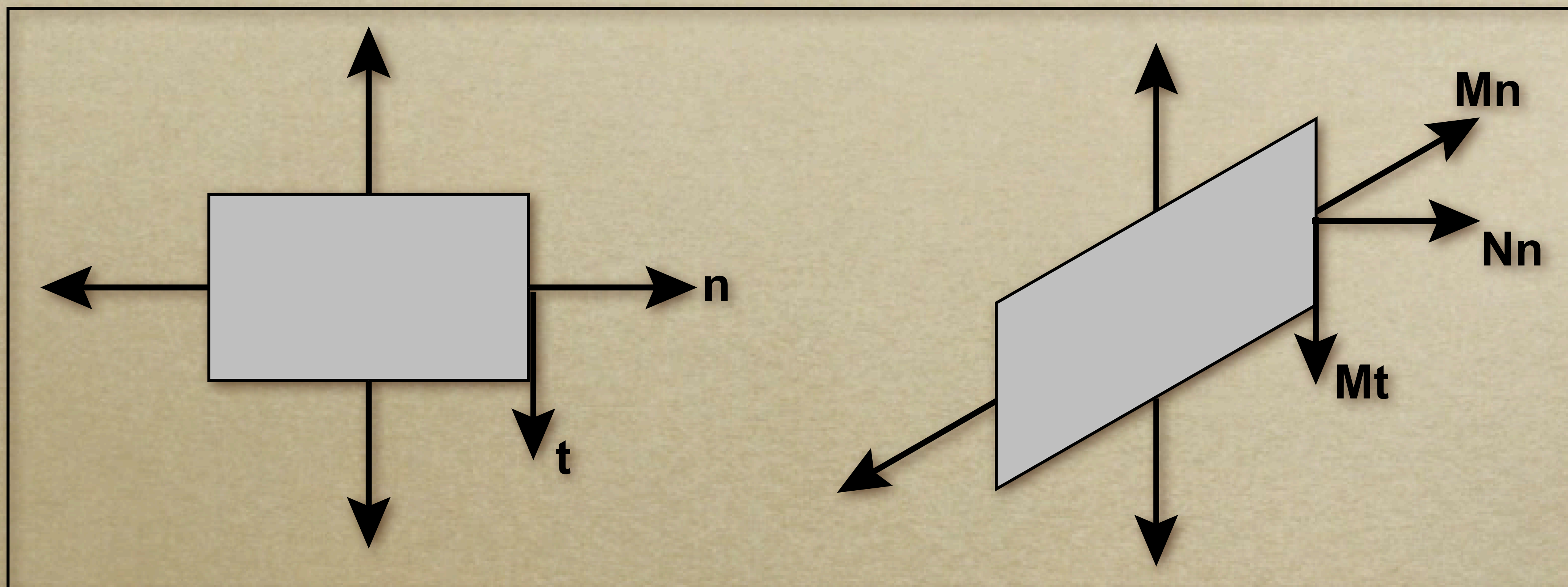
- In general: $\mathbf{A}^{-1}$ undoes effect of $\mathbf{A}$
- Special cases:
  - Translation: negate t_x and t_y
  - Rotation: transpose
  - Scale: invert diagonal (axis-aligned scales)
- Others:
  - Invert matrix
  - Invert SVD matrices

Point Vectors / Direction Vectors

- Points in space have a 1 for the "w" coordinate
- What should we have for $\mathbf{a} - \mathbf{b}$?
  - $w = 0$
  - Directions not the same as positions
  - Difference of positions is a direction
  - Position + direction is a position
  - Direction + direction is a direction
  - Position + position is nonsense

43

Somethings Require Care



For example normals do not transform normally

$$\mathbf{M}(\mathbf{a} \times \mathbf{b}) \neq (\mathbf{M}\mathbf{a}) \times (\mathbf{M}\mathbf{b})$$

Use inverse transpose of the matrix for normals.
See text book.

3D Transformations

- Generally, the extension from 2D to 3D is straightforward
  - Vectors get longer by one
  - Matrices get extra column and row
  - SVD still works the same way
  - Scale, Translation, and Shear all basically the same
- Rotations get interesting

45

Translations

$$\tilde{\mathbf{A}} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

For 2D

$$\tilde{\mathbf{A}} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For 3D

Scales

$$\tilde{\mathbf{A}} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For 2D

$$\tilde{\mathbf{A}} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For 3D

(Axis-aligned!)

Shears

$$\tilde{\mathbf{A}} = \begin{bmatrix} 1 & h_{xy} & 0 \\ h_{yx} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

For 2D

$$\tilde{\mathbf{A}} = \begin{bmatrix} 1 & h_{xy} & h_{xz} & 0 \\ h_{yx} & 1 & h_{yz} & 0 \\ h_{zx} & h_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For 3D

(Axis-aligned!)

48

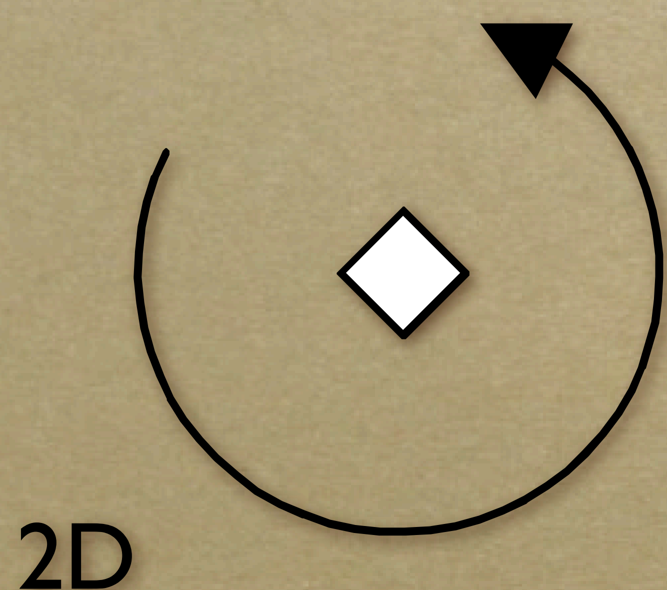
Shears

$$\tilde{\mathbf{A}} = \begin{bmatrix} 1 & h_{xy} & h_{xz} & 0 \\ h_{yx} & 1 & h_{yz} & 0 \\ h_{zx} & h_{zy} & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

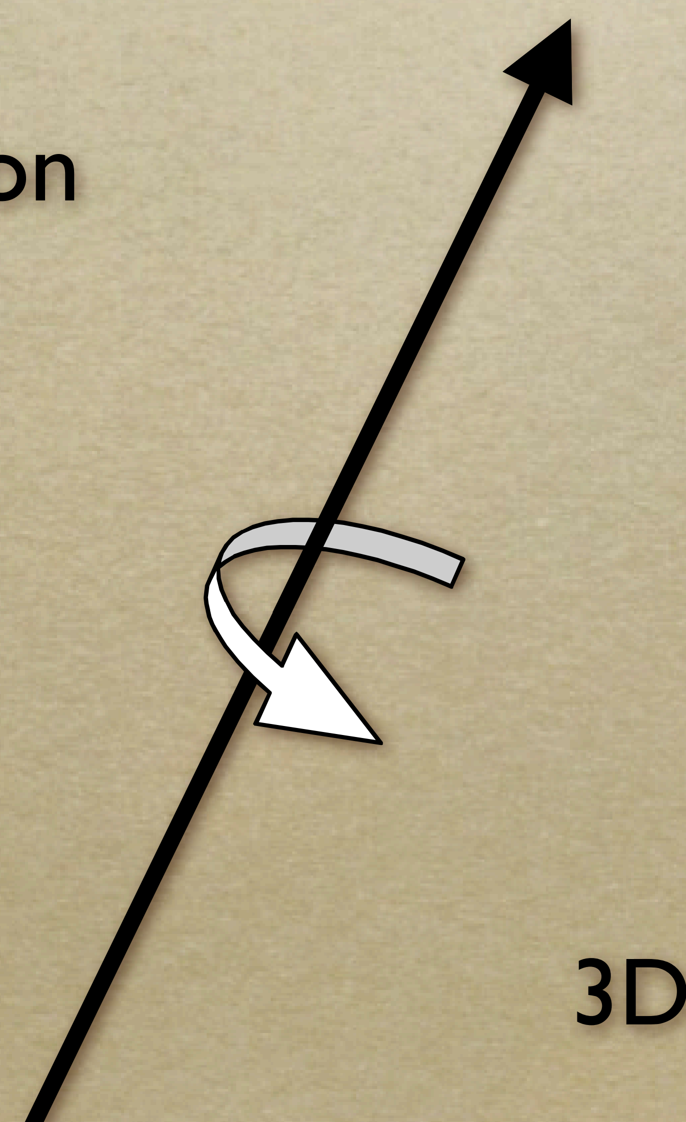
Shears y into x

Rotations

- 3D Rotations fundamentally more complex than in 2D
  - 2D: amount of rotation
  - 3D: amount and axis of rotation

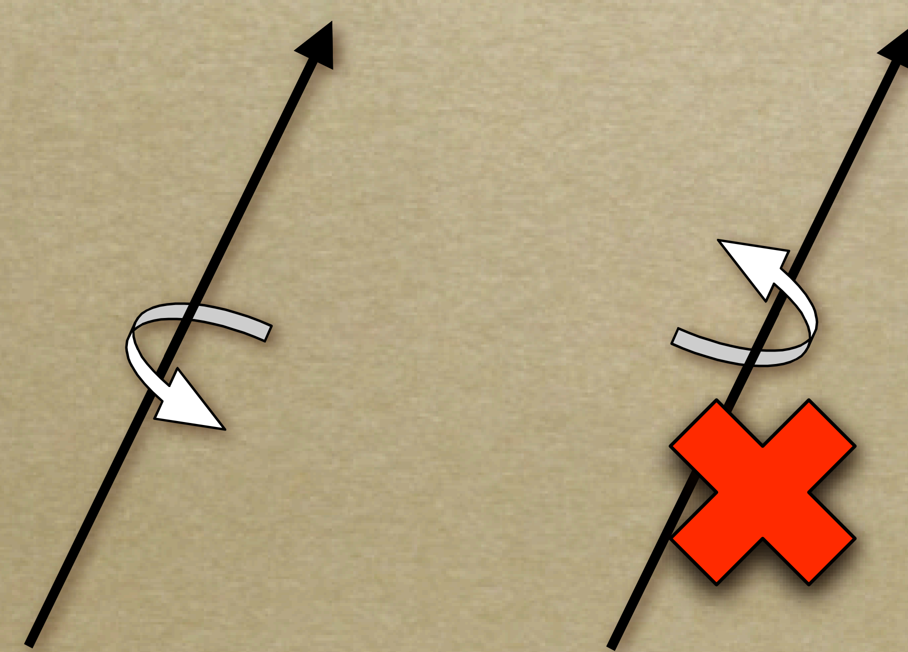


-vs-



Rotations

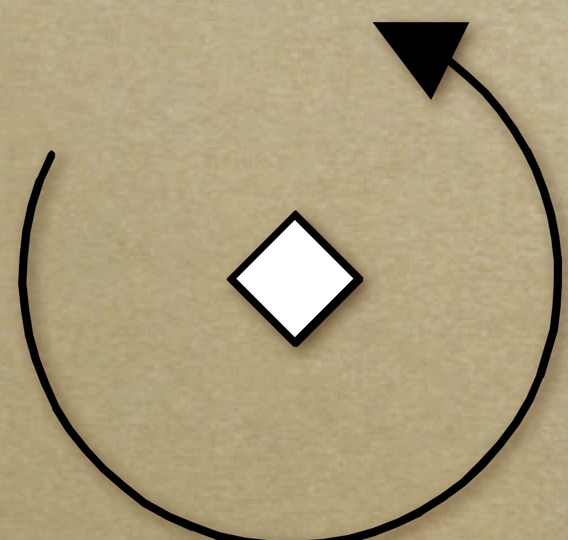
- Rotations still orthonormal
- $\mathrm{Det}(\mathbf{R}) = 1 \neq -1$
- Preserve lengths and distance to origin
- 3D rotations DO NOT COMMUTE!
- Right-hand rule
- Unique matrices

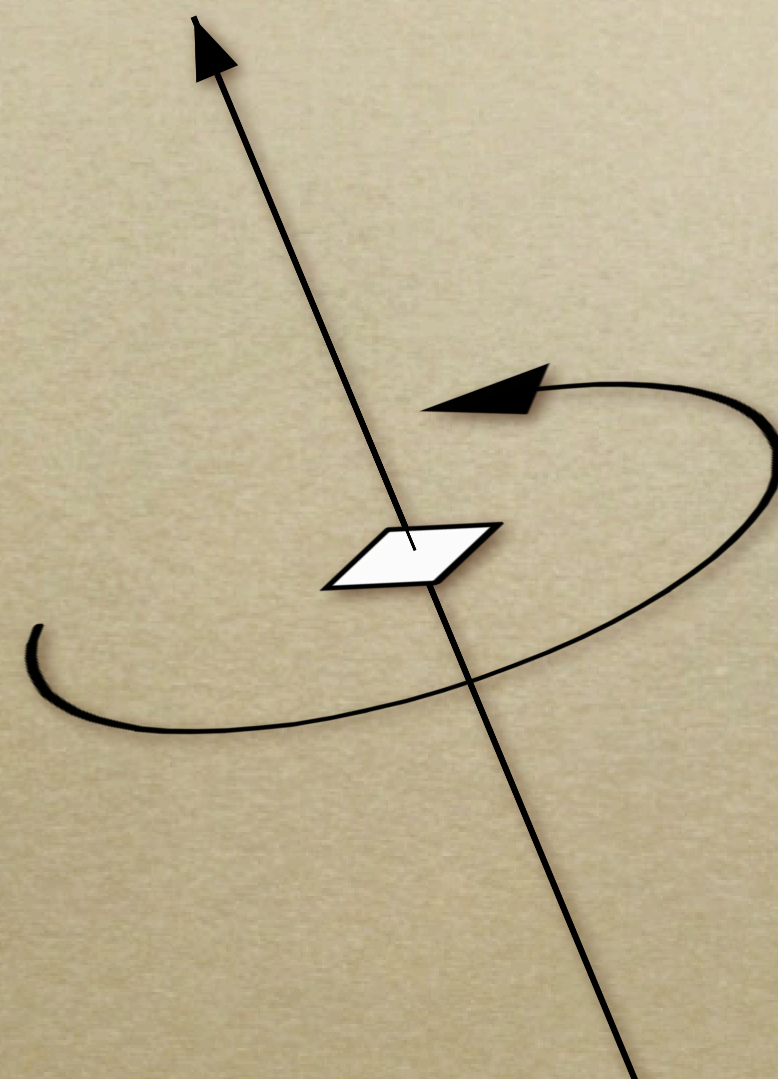


51

Axis-aligned 3D Rotations

- 2D rotations implicitly rotate about a third out of plane axis





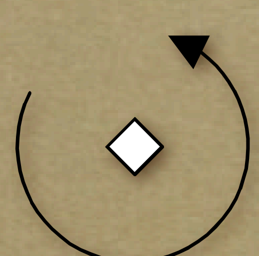
52

Axis-aligned 3D Rotations

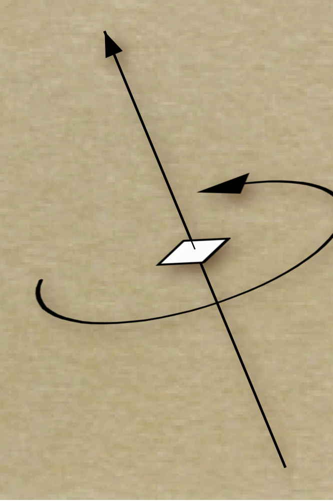
- 2D rotations implicitly rotate about a third out of plane axis

$$\mathbf{R} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

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



Note: looks same as $\tilde{\mathbf{R}}$



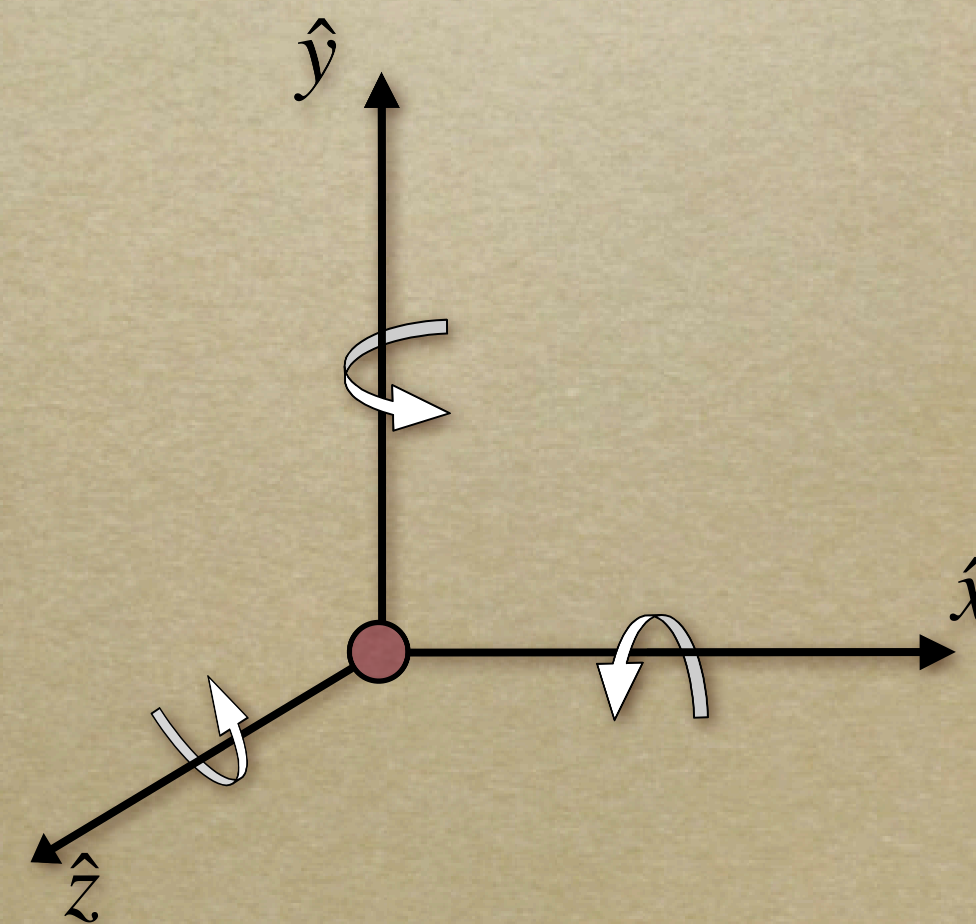
Axis-aligned 3D Rotations

$$\mathbf{R}_{\hat{x}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$\mathbf{R}_{\hat{y}} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

$$\mathbf{R}_{\hat{z}} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

"Z is in your face"



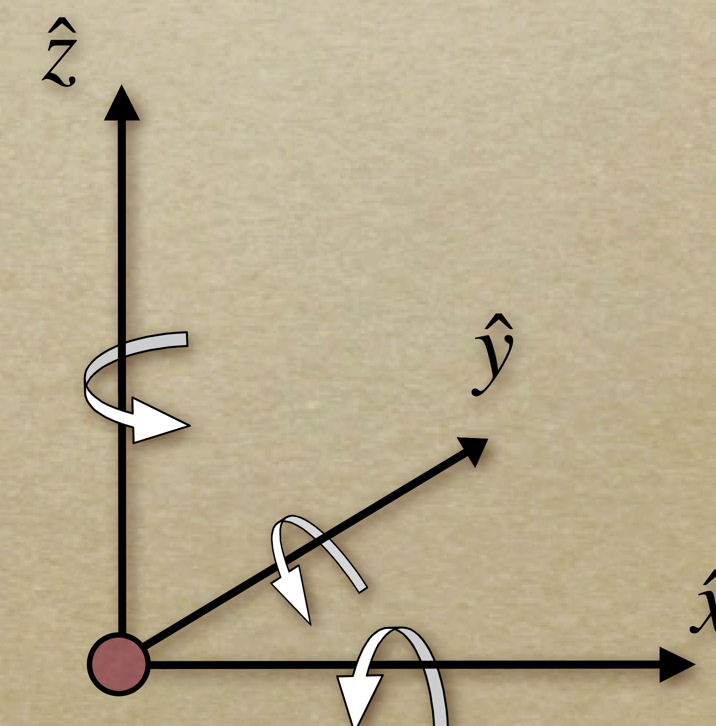
Axis-aligned 3D Rotations

$$\mathbf{R}_{\hat{x}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}$$

$$\mathbf{R}_{\hat{y}} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

$$\mathbf{R}_{\hat{z}} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Also right handed "Zup"



Axis-aligned 3D Rotations

- Also known as “direction-cosine” matrices

$$\mathbf{R}_{\hat{x}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \qquad \mathbf{R}_{\hat{y}} = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}$$

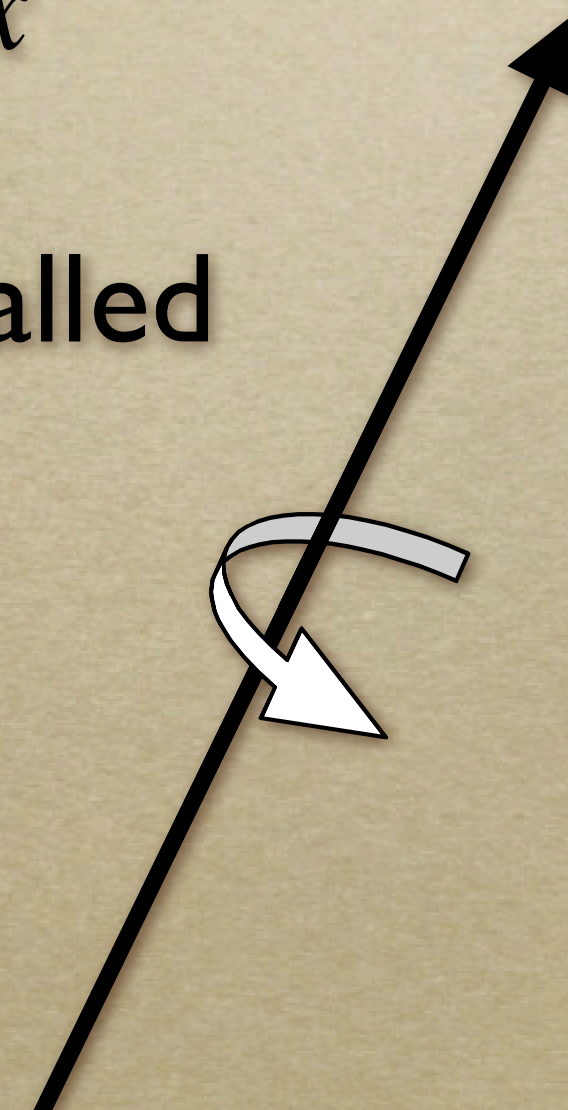
$$\mathbf{R}_{\hat{z}} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

56

Arbitrary Rotations

- Can be built from axis-aligned matrices:

$$\mathbf{R} = \mathbf{R}_{\hat{z}} \cdot \mathbf{R}_{\hat{y}} \cdot \mathbf{R}_{\hat{x}}$$



- Result due to Euler... hence called Euler Angles
- Easy to store in vector

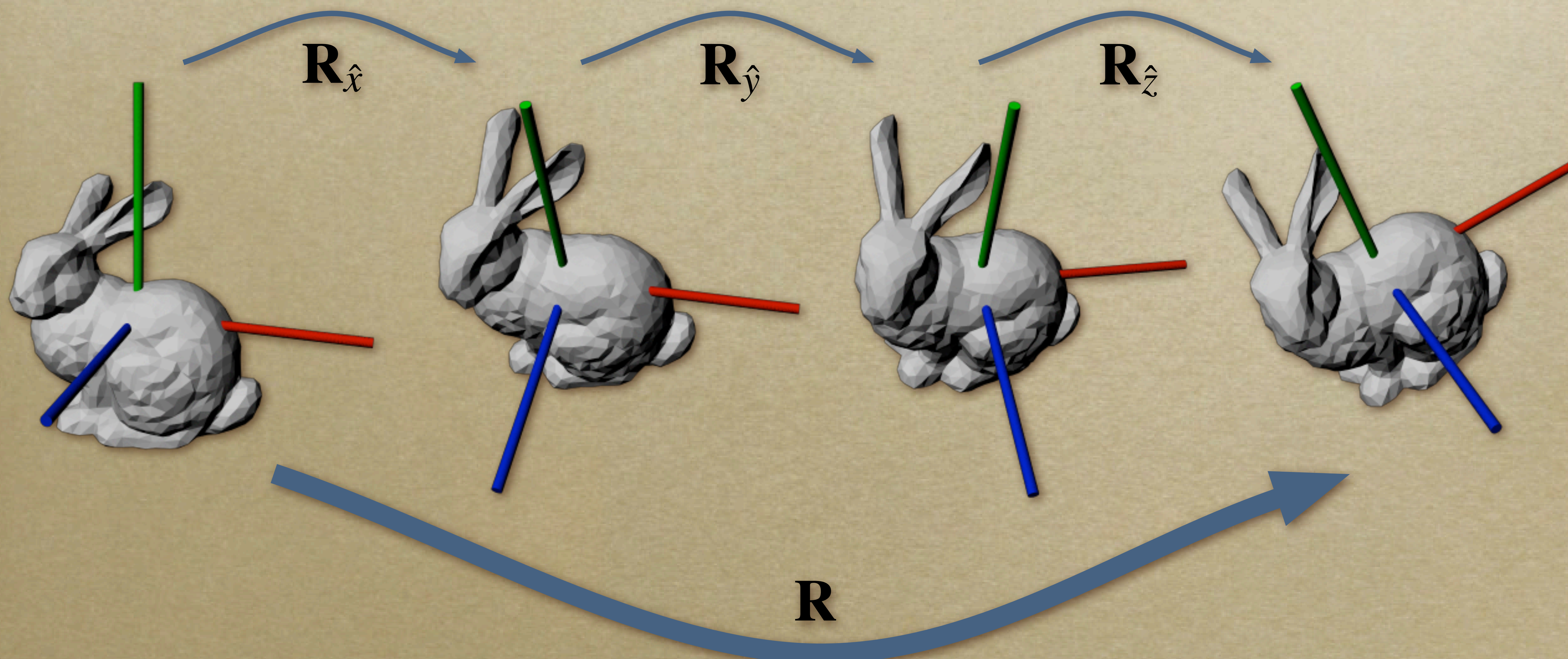
$$\mathbf{R} = \mathrm{rot}(x, y, z)$$

- But NOT a vector.

57

Arbitrary Rotations

$$\mathbf{R} = \mathbf{R}_{\hat{z}} \cdot \mathbf{R}_{\hat{y}} \cdot \mathbf{R}_{\hat{x}}$$



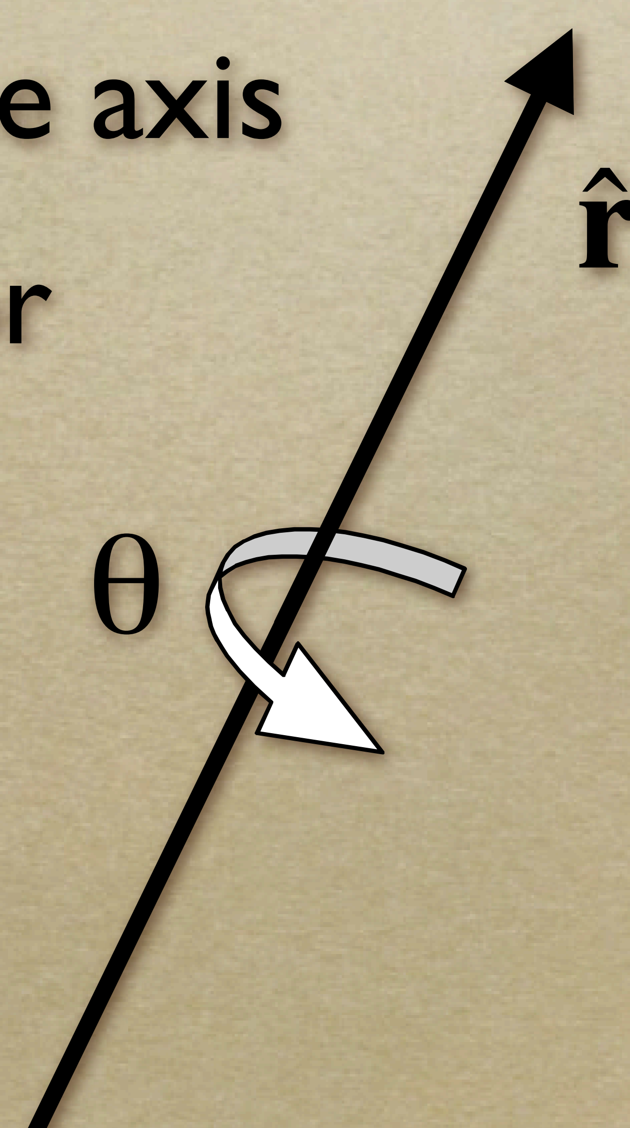
Arbitrary Rotations

- Allows tumbling
- Euler angles are non-unique
- Gimbal-lock
- Moving -vs- fixed axes
  - Reverse of each other

Exponential Maps

- Direct representation of arbitrary rotation
- AKA: axis-angle, angular displacement vector
- Rotate θ degrees about some axis
- Encode θ by length of vector

$$\theta = |\mathbf{r}|$$



Quaternions

- Due to Hamilton (1843)
  - Interesting history
  - Involves “hermaphroditic monsters”

61

Quaternions

- Uber-Complex Numbers

$$\mathrm{q} = (z_1, z_2, z_3, s) = (\mathbf{z}, s)$$

$$\mathrm{q} = iz_1 + jz_2 + kz_3 + s$$

$$i^2 = j^2 = k^2 = -1$$

$$\begin{array}{ll} ij = k & ji = -k \\ jk = i & kj = -i \\ ki = j & ik = -j \end{array}$$

Quaternions

- Multiplication natural consequence of defn.

$$\mathrm{q}\cdot\mathrm{p} = \left(\mathbf{z}_q s_p + \mathbf{z}_p s_q + \mathbf{z}_p \times \mathbf{z}_q \;\;,\;\; s_p s_q - \mathbf{z}_p \cdot \mathbf{z}_q\right)$$

- Conjugate

$$\mathrm{q}^* = (-\mathbf{z}, s)$$

- Magnitude

$$||\mathrm{q}||^2 = \mathbf{z}\cdot\mathbf{z} + s^2 = \mathrm{q}\cdot\mathrm{q}^*$$

Quaternions

- Vectors as quaternions

$$v = (\mathbf{v}, 0)$$

- Rotations as quaternions

$$r = (\hat{\mathbf{r}} \sin\frac{\theta}{2}, \cos\frac{\theta}{2})$$

- Rotating a vector

$$x' = r \cdot x \cdot r^*$$

- Composing rotations

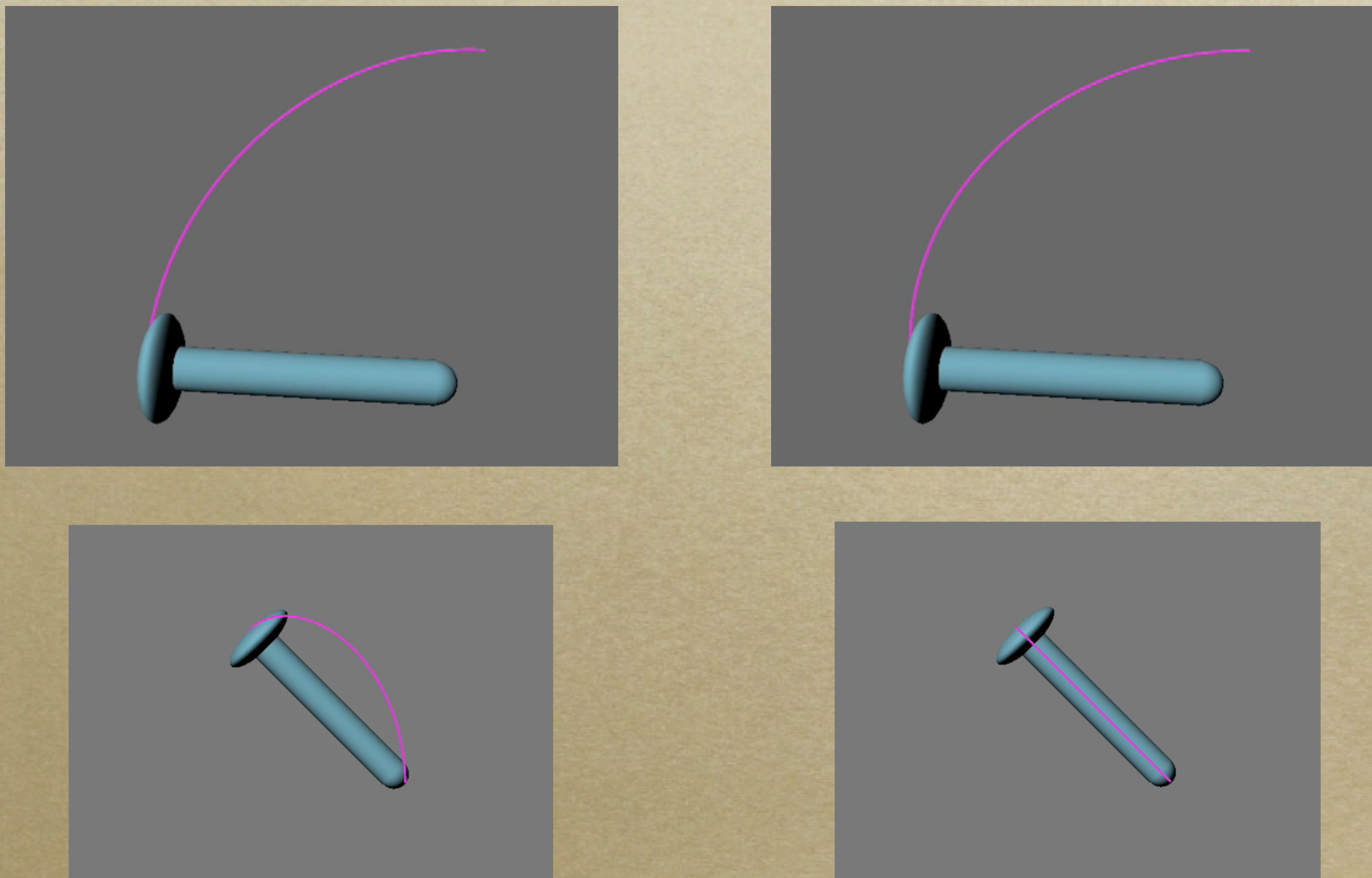
$$r = r_1 \cdot r_2$$

Quaternions

- No tumbling
- No gimbal-lock
- Orientations are "double unique"
- Surface of a 3-sphere in 4D $||\mathrm{r}|| = 1$
- Nice for interpolation

65

Interpolation



Rotation Matrices

- Eigen system
  - One real eigenvalue
  - Real axis is axis of rotation
  - Imaginary values are 2D rotation as complex number

Rotation Matrices

- Consider:

$$\mathbf{RI} = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{yz} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Columns are coordinate axes after transformation (true for general matrices)
- Rows are original axes in original system (not true for general matrices)