MATRIX TRANSFORMATIONS

Reference:

- "3D Math Primer for Graphics and Game Development, 2nd ed."
  - Chapter 5
  - Chapter 6.x
- http://www.teamten.com/lawrence/graphics/homogeneous/

TRANSFORM MATRICES (4.2 + 5.1)

- Take this vector: $\vec{v} = [-0.5 \quad 3 \quad 9]$
- Recall, if we think of this as a point, it means:
  - Start at the origin.
  - Move 0.5 units in the opposite direction of the **world** x-axis.
  - Move 3 units in the direction of the **world** y-axis
  - Move 9 units in the direction of the **world** z-axis.
- Said mathematically:
  - $\vec{v} = -0.5 * [1 \quad 0 \quad 0] +$
    $3 * [0 \quad 1 \quad 0] +$
    $9 * [0 \quad 0 \quad 1]$
  - The unit vectors are the **basis vectors** of our standard coordinate system.
  - Expressing this as a matrix equation:
  - $\vec{v} = [-0.5 \quad 3 \quad 9] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
  - Said another way: "we're **transforming** the vector [-0.5 3 9] by the **identity matrix**.

TRANSFORM MATRICES, CONT.

- Big deal, right?
  - Yes…but…what if we express that point relative to a "non-standard" coordinate system?
  - In other words, use *non-standard* basis vectors.
- Example:
  - Let's assume +y and +z are [0,1,0] and [0,0,1], respectively (as before)
  - But…let's make +x be [-1,0,0]
  - Now the matrix is:

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

TRANSFORM MATRICES, CONT.

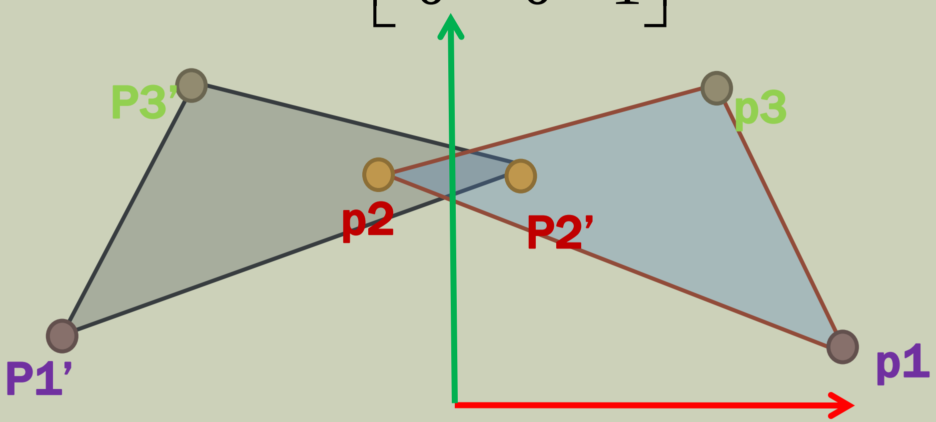
- Let's watch what happens as we "transform" a few points with this matrix…

$$\vec{p}_1 = \begin{bmatrix} 5 & 1 & 2 \end{bmatrix} \qquad \vec{p}_1^{'} = \vec{p}_1 * MysteryMat = \begin{bmatrix} -5 & 1 & 2 \end{bmatrix}$$

$$\vec{p}_2 = \begin{bmatrix} -1 & 4 & 3 \end{bmatrix} \qquad \vec{p}_2^{'} = \vec{p}_2 * MysteryMat = \begin{bmatrix} 1 & 4 & 3 \end{bmatrix}$$

$$\vec{p}_3 = \begin{bmatrix} 3 & 6 & 0 \end{bmatrix} \qquad \vec{p}_3^{'} = \vec{p}_3 * MysteryMat = \begin{bmatrix} -3 & 6 & 0 \end{bmatrix}$$

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



MysteryMat does a x-axis "Mirror".

TRANSFORM MATRICES, CONT.

- Let's try to apply this idea of basis vectors and local coordinate systems to a new matrix (4.2 in the book)

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

- What does this matrix do?
  - Method1: Try to plot a few points…OK, but not the best solution (we might not pick points that show the transform in action. Also, it takes a lot of time!)
  - Method2: Look at the local coordinate system defined by M…(see the next slide)

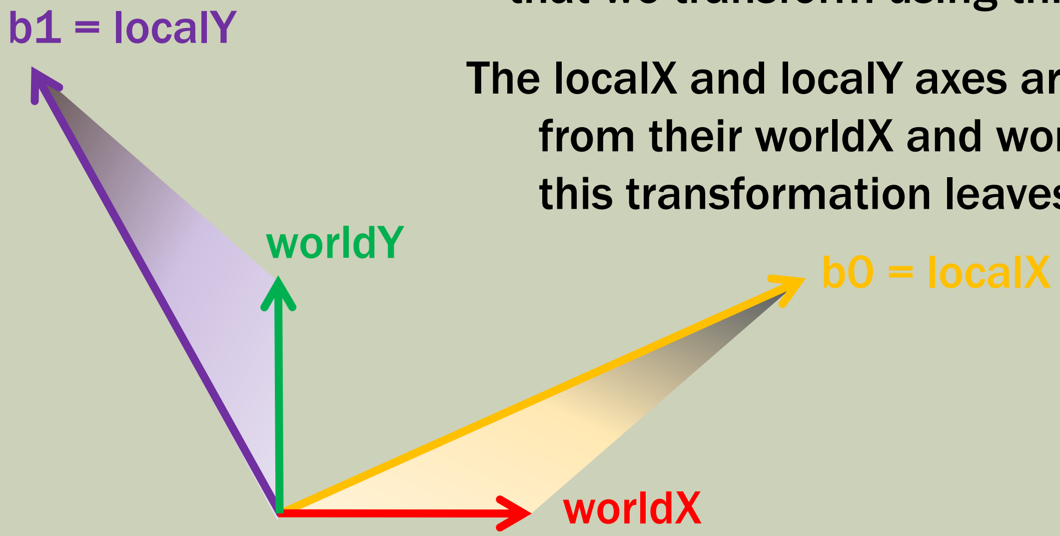
TRANSFORM MATRICES, CONT.

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

$$\vec{b}_0 = \begin{bmatrix} 2 & 1 & 0 \end{bmatrix}$$
$$\vec{b}_1 = \begin{bmatrix} -1 & 2 & 0 \end{bmatrix}$$
$$\vec{b}_2 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

Imagine "deforming" worldX into b0, worldY into b1, worldZ into b2. This is the same "deformation" the matrix will apply to all points that we transform using this matrix.

The localX and localY axes are rotated (~27 degrees) *and* scaled from their worldX and worldY counter-parts. localZ = worldZ, so this transformation leaves the z-values alone.



TRANSFORM MATRICES, CONT.

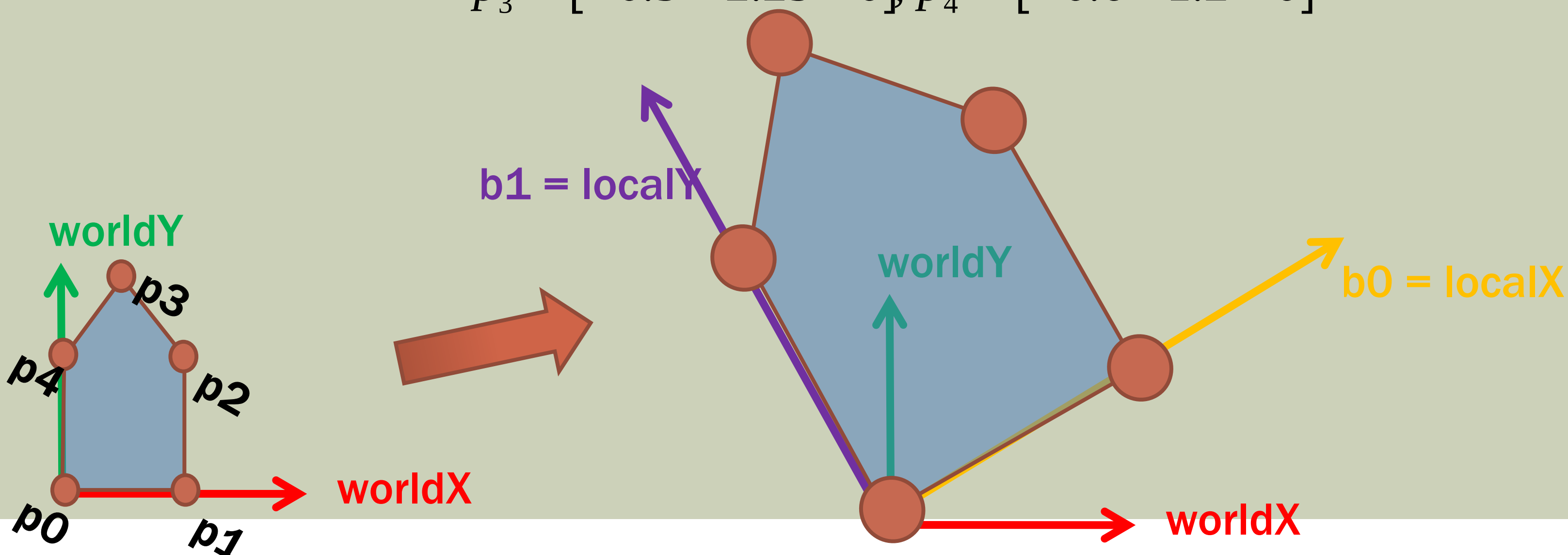
$$M = \begin{bmatrix} 2 & 1 & 0 \\ -1 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Let's say these are defined in **world space.**

$$\vec{p}_0 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \vec{p}_1 = \begin{bmatrix} 0.5 & 0 & 0 \end{bmatrix}, \vec{p}_2 = \begin{bmatrix} 0.5 & 0.6 & 0 \end{bmatrix}$$
$$\vec{p}_3 = \begin{bmatrix} 0.25 & 1 & 0 \end{bmatrix}, \vec{p}_4 = \begin{bmatrix} 0 & 0.6 & 0 \end{bmatrix}$$

When we multiply (transform) them by M (just as we did on the last slide), these points are transformed ***in the same way the world axes are transformed into the basis vectors of M***.

$$\vec{p}_0' = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, \vec{p}_1' = \begin{bmatrix} 1 & 0.5 & 0 \end{bmatrix}, \vec{p}_2' = \begin{bmatrix} 0.4 & 1.7 & 0 \end{bmatrix}$$
$$\vec{p}_3' = \begin{bmatrix} -0.5 & 2.25 & 0 \end{bmatrix}, \vec{p}_4' = \begin{bmatrix} -0.6 & 1.2 & 0 \end{bmatrix}$$



ROTATION TRANSFORM (5.1)

- Now the problem of defining new transforms just boils down to defining the (transformed) basis vectors.
- Let's apply this to the problem of creating a matrix which rotates θ degrees around the z axis.

ROTATION TRANSFORM (5.1), CONT.

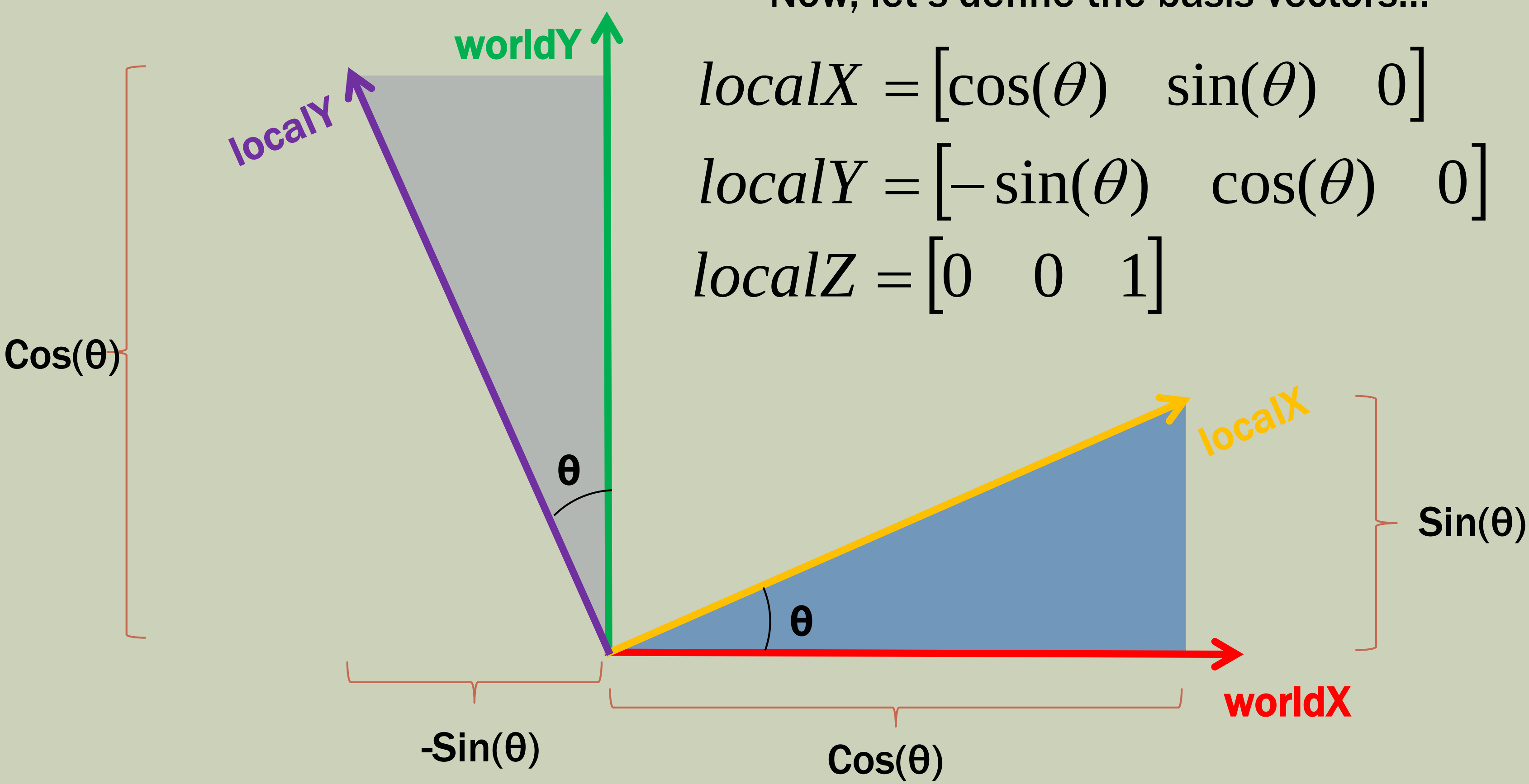
- We're looking in the +z direction:

Now, let's define the basis vectors…

$$localX = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \end{bmatrix}$$

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

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



ROTATION TRANSFORMATION (5.1), CONT.

- So, if we collect the 3 local vectors (basis vectors into a matrix), we have…

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

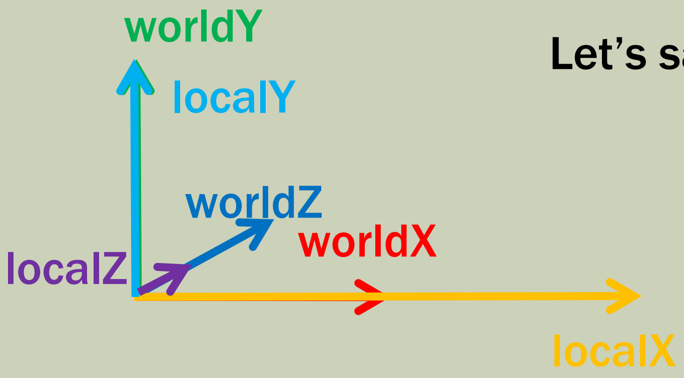
- We can similarly come up with matrices for x-axis and y-axis rotations…

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

Note: These matrices are for Left-Handed Systems. A Right-handed system would use the transpose.

SCALE TRANSFORM (5.2)

- The scale transform matrix is even simpler:



Let's say we want to scale:

2x in the x-direction
1x in the y-direction
½ in the z-direction

In general:

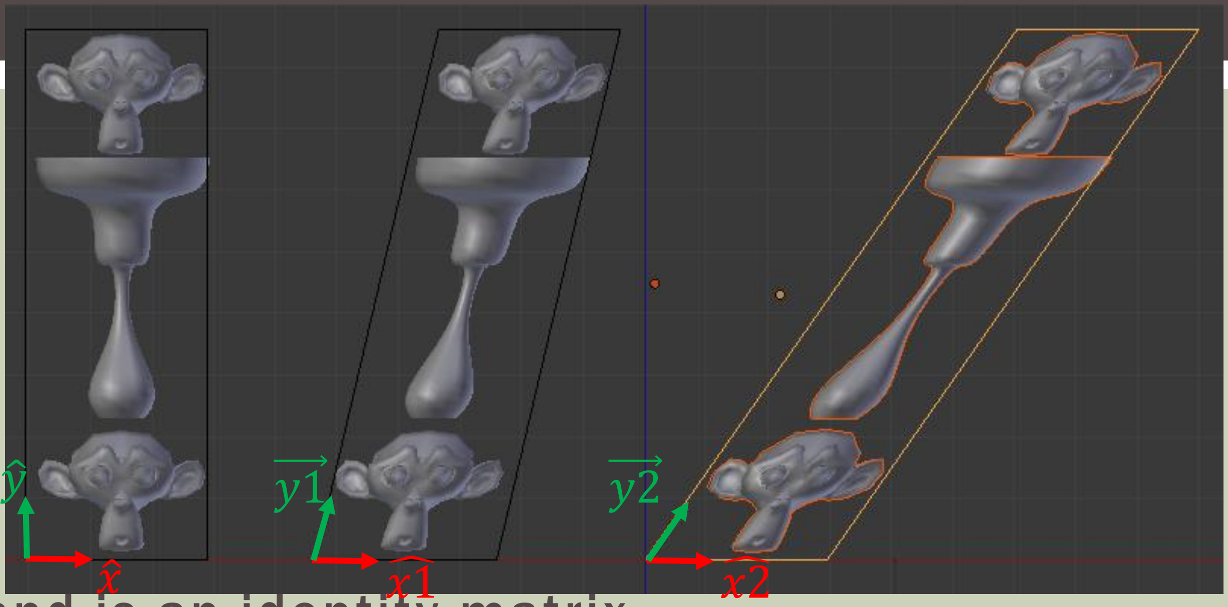
sx in the x-direction
sy in the y-direction
sz in the z-direction

$$S = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$$

$$\boxed{S = \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & sz \end{bmatrix}}$$

If sx = sy = sz, the operation is called a **universal scale.**

SHEAR (5.5)

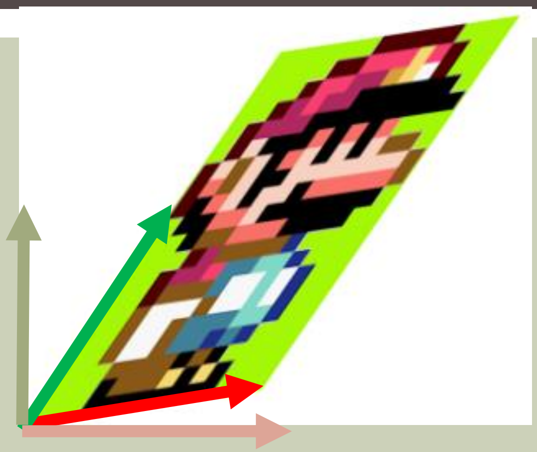


- Left-hand is an identity matrix
- Middle:
  - $\widehat{x_1} = \hat{x} = [1 \quad 0 \quad 0]$
  - $\widehat{z_1} = \hat{z} = [0 \quad 0 \quad 1]$
  - $\overrightarrow{y_1} \approx [0.3 \quad 1 \quad 0]$
- Right:
  - $\overrightarrow{y_1} \approx [0.8 \quad 1 \quad 0]$

$$SH_1 = \begin{bmatrix} 1 & 0 & 0 \\ \mathbf{0.3} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$SH_2 = \begin{bmatrix} 1 & 0 & 0 \\ \mathbf{0.8} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

SHEAR IN 2 DIRECTIONS



- $\overrightarrow{x'} \approx [0.9 \quad 0.1 \quad 0]$
- $\overrightarrow{y'} \approx [0.5 \quad 1 \quad 0]$
- $SH_{xy} = \begin{bmatrix} 0.9 & 0.1 & 0 \\ 0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

TRANSLATION (6.4)

- Let's say we want to develop a matrix for translation (let's say by amounts tx, ty, and tz).
  - Seems simple, right?
  - Wrong!
- Try it

TRANSLATION (6.4)

- Try it another way:

$$T = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

- Try to come up with values for a-i such that *all points* transformed by the matrix are moved by +4(x), +5(y), +6(z):

- You can't do it!

$$\begin{bmatrix} 5 & 7 & 3 \end{bmatrix} = \begin{bmatrix} 1 & 2 & -3 \end{bmatrix} * T \quad \textbf{\textit{and}} \quad \begin{bmatrix} 2 & 14 & 13 \end{bmatrix} = \begin{bmatrix} -2 & 9 & 7 \end{bmatrix} * T$$

TRANSLATION

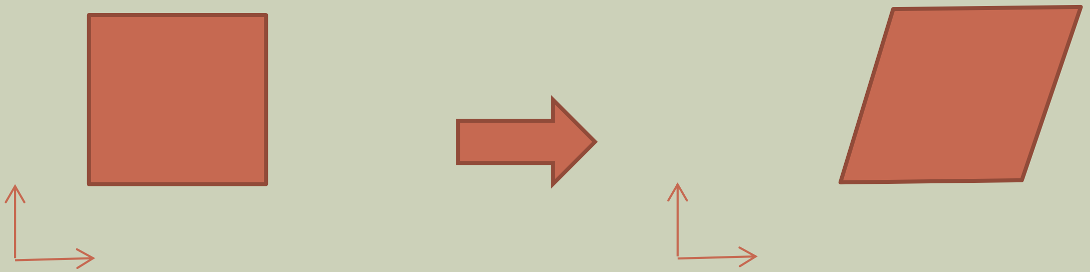
- Up to this point we've been looking at **linear transformations** (you can do these with a 3x3 (for 3d) matrix).
  - These <u>only</u> transform points *about the origin*.
- Translation is an example of an **affine transformation** (a linear transform followed by a translation).
  - You need to look at the problem in a higher dimension…
  - …enter **homogeneous space**!

TRANSLATION, CONT.

- But, by using so-called **homogeneous coordinates**, we can.
- To visualize this, we'll look at a (x-)shear matrix in 2d.

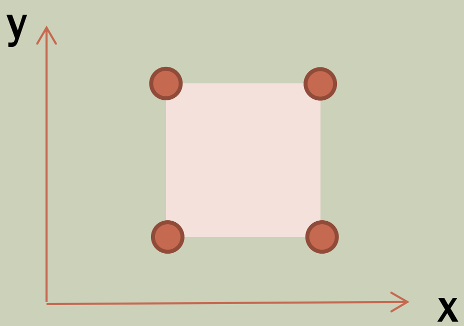
$$\begin{bmatrix} 1 & 0 \\ shearX & 1 \end{bmatrix}$$

- Suppose shearX is 0.5:

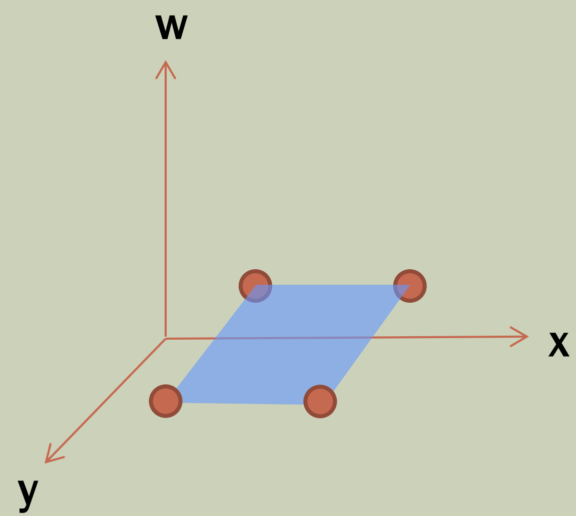


TRANSLATIONS, CONT.

- Represent 2d points in 3d space, with w as 1.0



2D

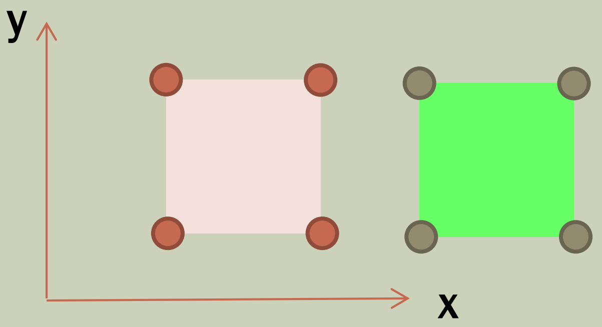


3D

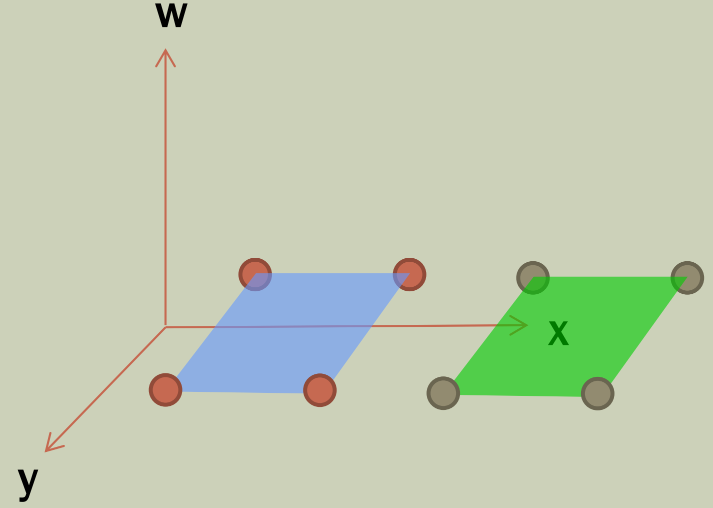
TRANSLATIONS, CONT.

- Now transform by a 3d shear in the x direction by 2.0 units

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 0 & 1 \end{bmatrix}$$



2D



3D

TANSLATIONS, CONT.

- So…a shear in 3d, produces a *translation* in 2d.
- We can do a translation in 3d by shearing in 4d.
  - We can't really visualize it – you just have to trust the math☺

$$\boxed{T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix}}$$

TRANSLATION, CONT.

- Note, if you put a w of 0, the translation doesn't affect the vector
  - Useful for vectors.
- W should be:
  - **1** if this is a point.
  - **0** if this is a vector (direction + magnitude).
- We'll see another application of this in perspective projections.
  - In this case we'll end up with w values *after the transformation* that are not 0 nor 1.
  - In this case we'll have to do a w-divide (homogeneous divide)

TRANSLATION, CONT.

- Now, we can do translation, by using a 4x4 translation matrix like this:
- Take our problem cases from before:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$

$$\begin{bmatrix} 5 & 7 & 3 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 2 & -3 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 4 & 5 & 6 & 1 \end{bmatrix}$$

Because it's a point

$$\begin{bmatrix} 2 & 14 & 13 & 1 \end{bmatrix} = \begin{bmatrix} -2 & 9 & 7 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 4 & 5 & 6 & 1 \end{bmatrix}$$

It's still a point

TRANSLATION, CONT.

- An extra bonus:

$$\begin{bmatrix} \boxed{\begin{matrix} 4 \\ 0 \\ 0 \end{matrix}} \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 4 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

This one's a vector

It's still a vector

Note how the vector is unchanged. Remember that you can re-draw a vector anywhere and it still has the same length and magnitude – here's proof!

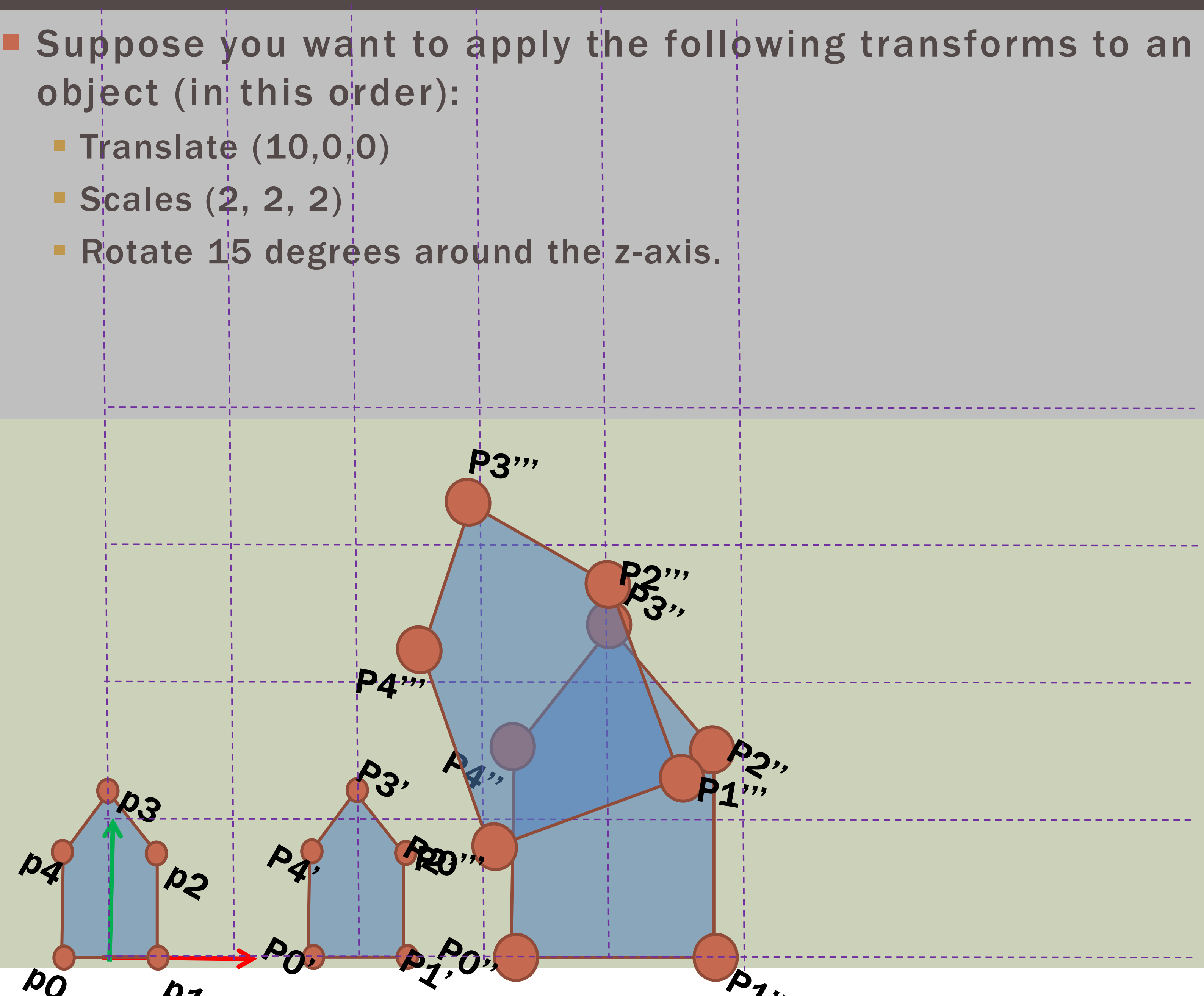
4X4 TRANSFORMATION MATRICES

- OK, so now that we're using 4x4 matrices for translation, what about all the other matrices?
- You can use 4x4 matrix, with the "old" 3x3 (linear) matrix embedded in the upper-left.
  - Or…think of it this affine transformation as a linear transform (the 3x3 upper-left part) and a 0-vector translation afterwards.
- Example: RotZ now becomes:

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

TRANSFORM MATRIX CONCATENATION (5.6)

- Suppose you want to apply the following transforms to an object (in this order):
  - Translate (10,0,0)
  - Scales (2, 2, 2)
  - Rotate 15 degrees around the z-axis.



TRANSFORM MATRIX CONCATENATION (8.7)

- Numerically: $\vec{p}_0 = [-2 \quad 0 \quad 0 \quad 1], \vec{p}_1 = [2 \quad 0 \quad 0 \quad 1], \vec{p}_2 = [2 \quad 3.5 \quad 0 \quad 1]$
  $\vec{p}_3 = [0 \quad 6 \quad 0 \quad 1], \vec{p}_4 = [-2 \quad 3.5 \quad 0 \quad 1]$

- After Multiplying by the translation (10,0,0) matrix:

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 10 & 0 & 0 & 1 \end{bmatrix}$$

$$\vec{p}_0' = [8 \quad 0 \quad 0 \quad 1], \vec{p}_1' = [12 \quad 0 \quad 0 \quad 1], \vec{p}_2' = [12 \quad 3.5 \quad 0 \quad 1]$$
$$\vec{p}_3' = [10 \quad 6 \quad 0 \quad 1], \vec{p}_4' = [8 \quad 3.5 \quad 0 \quad 1]$$

- After multiplying by the scale (2,2,2) matrix:

$$S = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\vec{p}_0'' = [16 \quad 0 \quad 0 \quad 1], \vec{p}_1'' = [24 \quad 0 \quad 0 \quad 1], \vec{p}_2'' = [24 \quad 7 \quad 0 \quad 1]$$
$$\vec{p}_3'' = [20 \quad 12 \quad 0 \quad 1], \vec{p}_4'' = [16 \quad 7 \quad 0 \quad 1]$$

- After multiplying by the rotation of 15 degrees around the z axis:

$$\vec{p}_0''' = [15.455 \quad 4.14 \quad 0 \quad 1], \vec{p}_1''' = [23.182 \quad 6.212 \quad 0 \quad 1], \vec{p}_2''' = [21.37 \quad 12.973 \quad 0 \quad 1]$$
$$p_3''' = [16.213 \quad 16.767 \quad 0 \quad 1], \vec{p}_4''' = [13.643 \quad 10.9 \quad 0 \quad 1]$$

$$R = \begin{bmatrix} \cos(15) & \sin(15) & 0 & 0 \\ -\sin(15) & \cos(15) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

TRANSFORM MATRIX CONCATENATION, CONT.

- You can get the same *net* effect by **concatenating** these 3 transforms.

$$C = T * S * R = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 10 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos(15) & \sin(15) & 0 & 0 \\ -\sin(15) & \cos(15) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 20 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos(15) & \sin(15) & 0 & 0 \\ -\sin(15) & \cos(15) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2*\cos(15) & 2*\sin(15) & 0 & 0 \\ -2*\sin(15) & 2*\cos(15) & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 20*\cos(15) & 20*\sin(15) & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1.93 & 0.52 & 0 & 0 \\ -0.52 & 1.93 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 19.32 & 5.18 & 0 & 1 \end{bmatrix}$$

VERY IMPORTANT: In a right-handed system, multiply in the <u>reverse</u> order you want to apply the transform (In a left-handed system, you do it in the forward order).

You <u>won't</u> get the same composite matrix if you don't follow this rule.

TRANSFORM MATRIX CONCATENATION, CONT.

- Let's test it.

$$\vec{p}_0 = \begin{bmatrix} -2 & 0 & 0 & 1 \end{bmatrix}, \vec{p}_1 = \begin{bmatrix} 2 & 0 & 0 & 1 \end{bmatrix}, \vec{p}_2 = \begin{bmatrix} 2 & 3.5 & 0 & 1 \end{bmatrix}$$
$$\vec{p}_3 = \begin{bmatrix} 0 & 6 & 0 & 1 \end{bmatrix}, \vec{p}_4 = \begin{bmatrix} -2 & 3.5 & 0 & 1 \end{bmatrix}$$

$$C = \begin{bmatrix} 1.93 & 0.52 & 0 & 0 \\ -0.52 & 1.93 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 19.32 & 5.18 & 0 & 1 \end{bmatrix}$$

Transform each of the points by C, and we get…

$$\vec{p}_0' = \begin{bmatrix} 15.455 & 4.14 & 0 & 1 \end{bmatrix}, \vec{p}_1' = \begin{bmatrix} 23.182 & 6.212 & 0 & 1 \end{bmatrix}, \vec{p}_2' = \begin{bmatrix} 21.37 & 12.973 & 0 & 1 \end{bmatrix}$$
$$\vec{p}_3' = \begin{bmatrix} 16.213 & 16.767 & 0 & 1 \end{bmatrix}, \vec{p}_4' = \begin{bmatrix} 13.643 & 10.9 & 0 & 1 \end{bmatrix}$$

Which is the same points we got by transforming by the individual matrices!

Big Deal, right? ***No!*** To see why, we'll need to analyze the costs…[enough time?]

SCENE GRAPHS