Lecture 4-5
Transformations, Projections, and Viewing

1.1 Points and Vectors

- Points
  - A solid object with infinitely small size
    → a mathematical abstraction
  - Use points to define locations, to describe trajectories of objects, and model geometric shapes…
- Vectors
  - Length (magnitude) + direction : e.g. velocity
  - Add two vectors → parallelogram rule of analytical geometry
  - Multiply a vector by a scalar → change magnitude not direction
  - Vectors and their operations (addition and scalar multiplication) → a vector space

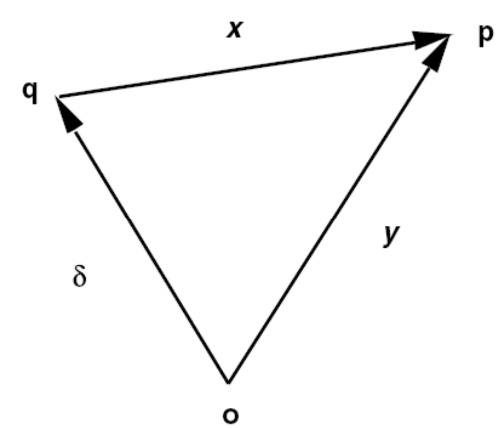
1.1 Points and Vectors

- Points and Vectors
  - Pick an origin o, each point p corresponds to a vector x

    (since each p and o defines a length and a direction)

    → produces vectors by point difference

  - The correspondence depends on the origin
    - $y = x + \delta$, $\delta = q - o$



Change of origin

1.1 Point Space→Vector Space

- inner product (dot product): xy
- norm (length) of a vector: $\|x\| = \sqrt{x.x}$
- a unit vector ← length equals unity
- orthogonal vectors ← dot product is zero
- orthonormal bases (largest sets of unit vectors, pairwise orthogonal)

In such a basis $\{e_1, e_2, \ldots, e_n\}$:

- $\boldsymbol{x} = x_1 e_1 + x_2 e_2 + \ldots + x_n e_n$
- the inner product:

$$\boldsymbol{x}.\boldsymbol{y} = X^t Y = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

and the length becomes:

$$\|x\| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

1.2 frame

- Points correspond to Vectors, given a fixed origin
- Vectors correspond to column matrices, given a fixed basis

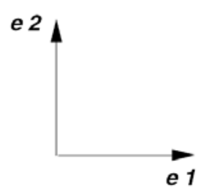
➔ Represent points using column matrices

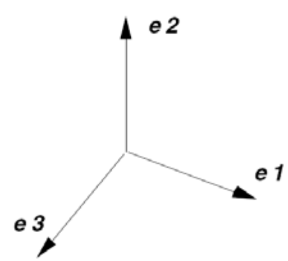
- A pair (origin, basis) is called a <u>frame</u>, or <u>coordinate system</u>
  - For a fixed frame, points → column matrices (elements of the matrix are called the coordinates of the point in that frame)

1.3 cross product

- Useful especially in 3D, defined in a right-handed, orthonormal , 3-D basis:

$$\boldsymbol{x} \times \boldsymbol{y} = (x_2 y_3 - x_3 y_2)\boldsymbol{e}_1 + (x_3 y_1 - x_1 y_3)\boldsymbol{e}_2 + (x_1 y_2 - x_2 y_1)\boldsymbol{e}_3$$





$$\boldsymbol{e}_1 \times \boldsymbol{e}_2 = \boldsymbol{e}_3$$

$$\boldsymbol{e}_2 \times \boldsymbol{e}_3 = \boldsymbol{e}_1$$

$$\boldsymbol{e}_3 \times \boldsymbol{e}_1 = \boldsymbol{e}_2$$

- Cross product of two parallel vectors is zero
- Otherwise, its magnitude = the area of the parallelogram, its direction is perpendicular to both vectors
- For completing a 3D orthonormal basis when two of its vectors are known

2. Transformations

- Moving, scaling, and deforming objects are fundamental operations in geometric modeling.
- If objects are considered as sets of points, what we need are transformations that map points onto other points

…Start with some basic transformations…

Computing Transformation is important because we want:

- To compute/represent transformations when necessary
  - For rendering, interactive visualization
  - Other visual computing applications
- To analyze shapes under transformation
  - Find invariant shape properties under various transformations (applications: shape descriptors for shape retrieval, object recognition, object tracking/localization…)

geometry is the study of **invariants** under **transformations**

2.1 Linear Transformations

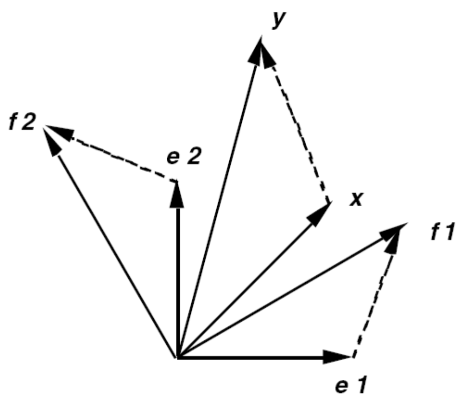
- A transformation is linear if it distributes over linear combinations, i.e. $T(a\mathbf{x} + b\mathbf{y}) = aT(\mathbf{x}) + bT(\mathbf{y})$
- Computing Linear Transformation → changing between two bases
- For a fixed basis E, each transformation to a new basis F corresponds to a square matrix
- Solving a square matrix that maps a basis to another basis is not difficult

2.1 Linear Transformations

- A transformation is linear if it distributes over linear combinations, i.e. $$T(a\mathbf{x}+b\mathbf{y}) = aT(\mathbf{x})+bT(\mathbf{y})$$

- Suppose we have two bases $$E=[\mathbf{e}_1 \quad \cdots \quad \mathbf{e}_n]$$ $$F=[f_1 \quad \cdots \quad f_n]$$

- And we want to find the linear transformation: $$T_{ef}(\mathbf{e}_i)=f_i, \quad i=1,\ldots,n.$$



- What is the effect of such a transformation on an arbitrary vector? $$\mathbf{y}=T_{ef}(\mathbf{x}) \qquad \mathbf{x}=EX^e$$

- Following the linearity definition: $$\mathbf{y}=T_{ef}(\mathbf{x})=\left[T_{ef}(\mathbf{e}_1) \quad \cdots \quad T_{ef}(\mathbf{e}_n)\right]X^e=[f_1 \quad \cdots \quad f_n]X^e$$

2.1 Linear Transformations (cont.)

$$y = T_{ef}(x) = \left[T_{ef}(e_1) \quad \cdots \quad T_{ef}(e_n) \right] X^e = \left[f_1 \quad \cdots \quad f_n \right] X^e$$

$$\Rightarrow \quad y = \left[EF_1^e \quad \cdots \quad EF_n^e \right] X^e = E\left[F_1^e \quad \cdots \quad F_n^e \right] X^e$$

- Meaning the components of y in basis E are

$$M^e = \left[F_1^e \quad \cdots \quad F_n^e \right]$$

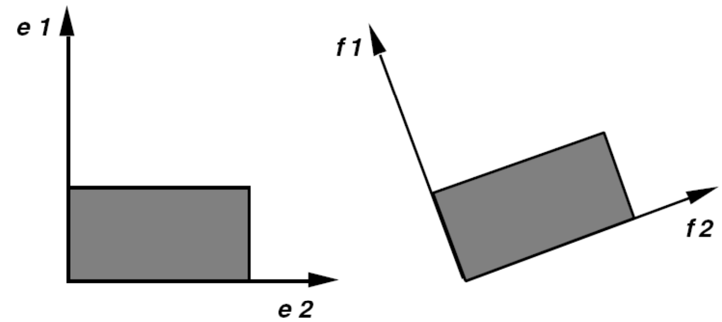
$$Y^e = M^e X^e$$

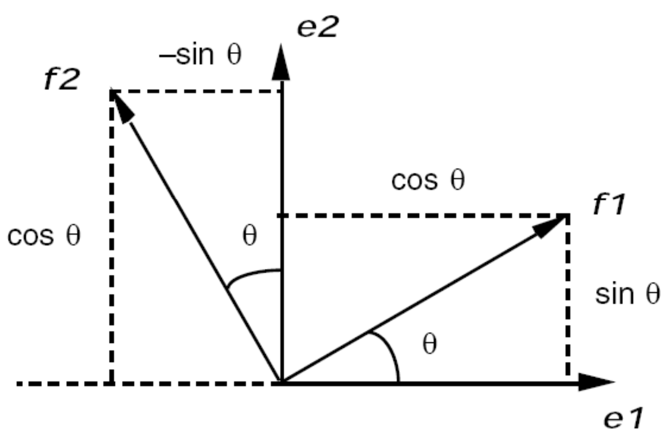
- These two equations show:
  - For a fixed basis E, <u>each transformation corresponds to a square matrix</u> (correspondence between linear transformation and square matrices)
  - They give us computational tools for evaluating transformation effect on a vector (simply multiply the corresponding matrix)
  - The way to construct this matrix mapping a basis to another basis is convenient

2.1 Linear Transformations (Example)

A Rotation Example:

- Coordinate frames are usually attached to objects, suppose we want to orient left rectangle such that it aligns with the right rectangle
- E.g. find transformation T such that
  - T(e1)=f1, T(e2)=f2





- Solve a linear system, based on elementary trigonometry
- M*e1 = f1, M*e2=f2

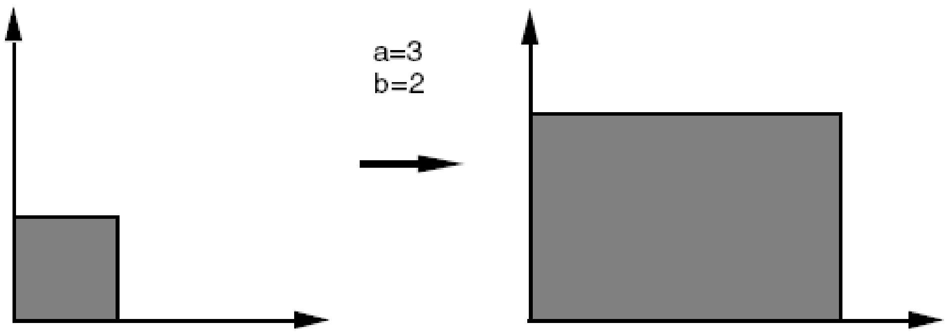
$$F_1^e = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \quad F_2^e = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}$$

$$M^e = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

2.1 Specific Linear Transformations

- Several most common linear 2D transformations
  - Scaling
  - Rotation
  - Shear
  - Reflection -- scaling with negative factors
  - Orthographic projection

2.1.1 Scaling



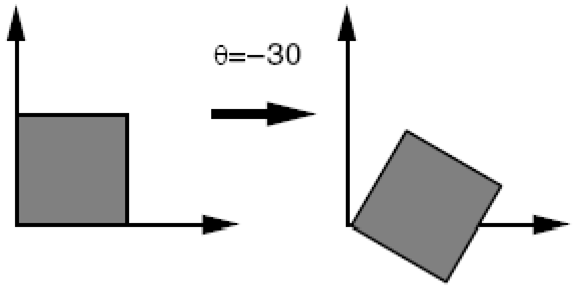
- 2D Transformation matrix: $\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}$
- Scaling a vector: $\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$
  - Scaling factors: a and b, along x and y axes
  - If a=b → uniform (isotropic) scaling

    <u>shape preserved</u>, only size changed
  - If a=b=1 → identity transform
- Directly extendible to 3D

2.1.2 Rotation and Shear

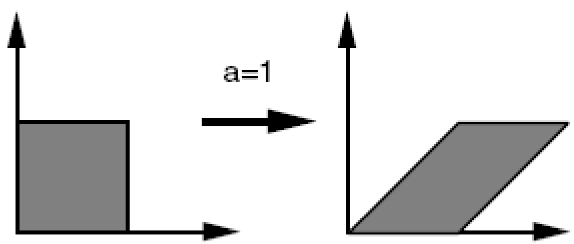
- Rotation in 2D:

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$



- Shear: let one of the off-diagonal elements of the scaling transformation matrix be non-zero

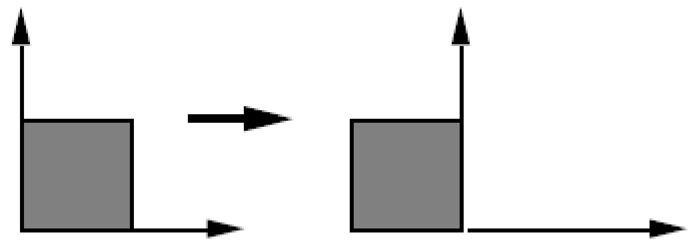
$$\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x+ay \\ y \end{bmatrix}$$



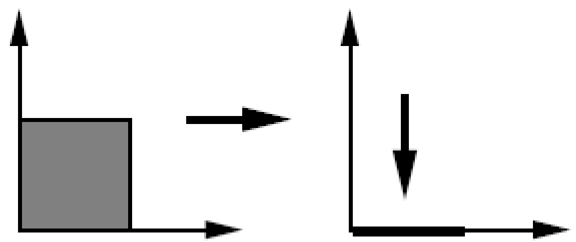
2.1.3 Reflection and Orthographic Projection

- Reflection = scaling with negative factors

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



- Orthographic Projection



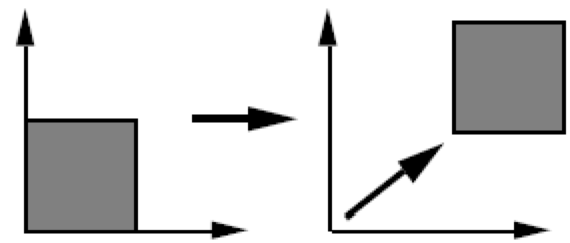
$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ 0 \end{bmatrix}$$

  - Does not map a basis onto another basis
  - Singular transformation (can't be inverted, lost all depth information along a direction)

2.2 Translation

- simply

$$\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \delta_1 \\ \delta_2 \end{bmatrix} = \begin{bmatrix} x+\delta_1 \\ y+\delta_2 \end{bmatrix}$$



  - Not a linear transformation
    (can't be computed by 2*2 matrix multiplication)

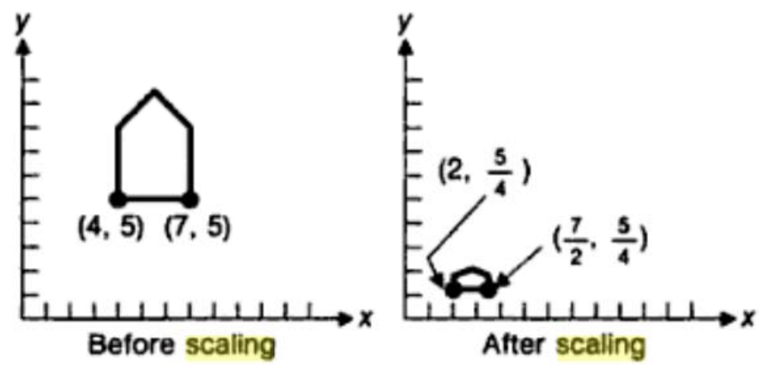
- <u>Affine Transformations</u>
  = translations + non-degenerate linear transformations
- <u>Rigid Transformations</u> = translations + rotations
  - If a transformation preserves distance → called isometries

2.3 Rotation/Scaling Center

- They are about the origin
  - Rotation: the whole space rotates
  - Scaling: the object will be farther or closer to the origin depending on the scales
- To transform around the shape center, or an any given point p: (3-step Composition)
  1) Translate p to the origin
  2) Conduct the transformation
  3) Translate the object back

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$



2.4 Homogeneous Coordinates

- Translations and linear transformations can be treated more uniformly if we introduce a different system of coordinates: homogeneous coordinates.
- Use the 2D example in the following, but can be generalized to n-D straightforwardly.

1. introduce an additional component and associate with the vector x the column matrix:

$$X^* = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} X \\ 1 \end{bmatrix}$$

X* → homogeneous coordinates

2.4 Homogeneous Coordinates (cont.)

2. Also add a third row and column to the linear transformation matrix $M^* = \begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix}$ i.e. $M^* = \begin{bmatrix} M & 0 \\ 0 & 1 \end{bmatrix}$

3. Multiply this augmented matrices, and get:

$$M^* X^* = \begin{bmatrix} M & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} = \begin{bmatrix} MX \\ 1 \end{bmatrix} = \begin{bmatrix} Y \\ 1 \end{bmatrix} = Y^*$$

Nothing changed so far.

- When elements of the third column become non-zero:

  e.g.

$$M^* = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix}$$

- We get:

$$Y^* = M^* X^* = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+a \\ y+b \\ 1 \end{bmatrix}$$

2.4 Homogeneous Coordinates (cont.)

- Now we have uniform treatment of translation and rotation
  - only one procedure needed to process both
  - one matrix-multiplication hardware works for both
  - Also deal with projections (for displaying 3D objects, later)

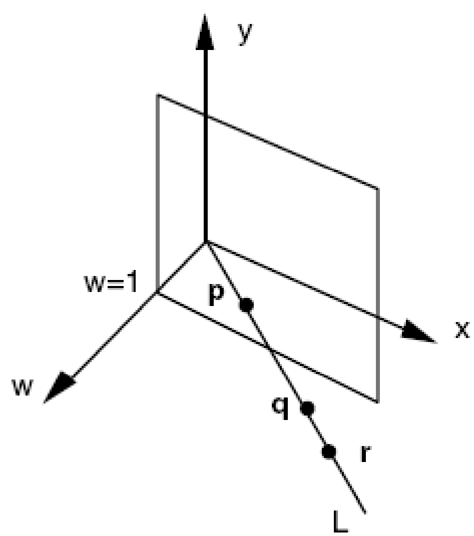
2.4 Homogeneous Coordinates (cont.)

-- Geometric Interpretation

(1) generalize the coordinates of an Euclidean point p:

$$P^{+} = \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Increase the dimension of the original space by 1
- The original standard Euclidean plane is at w=1

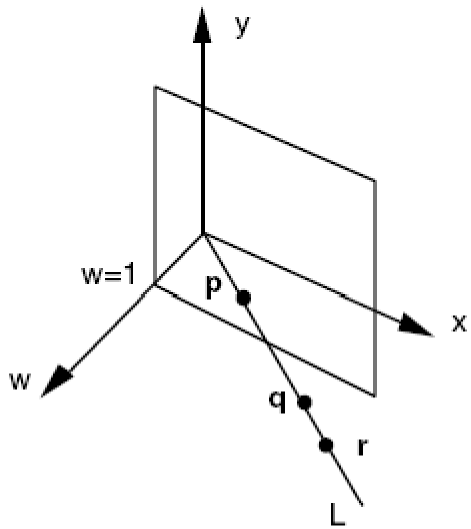


2.4 Homogeneous Coordinates (cont.)

-- Geometric Interpretation

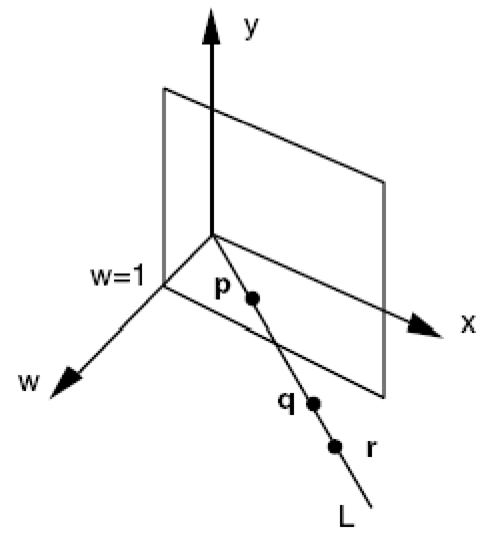
(2) Connect p with the origin (get line L)

- Each p corresponds to one line L, any point on L differ with p by a scaling
- We can always normalize the coordinate to $\begin{bmatrix} x/w \\ y/w \\ 1 \end{bmatrix}$



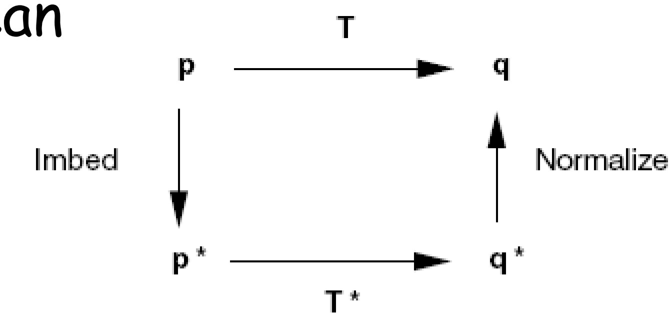
2.4 Homogeneous Coordinates (cont.)

-- Geometric Interpretation



- The set of all lines through the origin of our auxiliary 3D space is called the <u>projective plane</u>.
- The elements of the projective plane are called <u>projective points</u>. (note they are actually lines)
- Each Euclidean point p has a corresponding line L and projective point p*
- Therefore, we can manipulate Euclidean points through operations on their projective counterparts.

  (see the diagram on the right:



an affine transform T

= Imbed + projective transform + normalize)

Composition of Transformations

With homogeneous coordinates, affine transformations and their composition can be represented by matrices and their product

Examples:

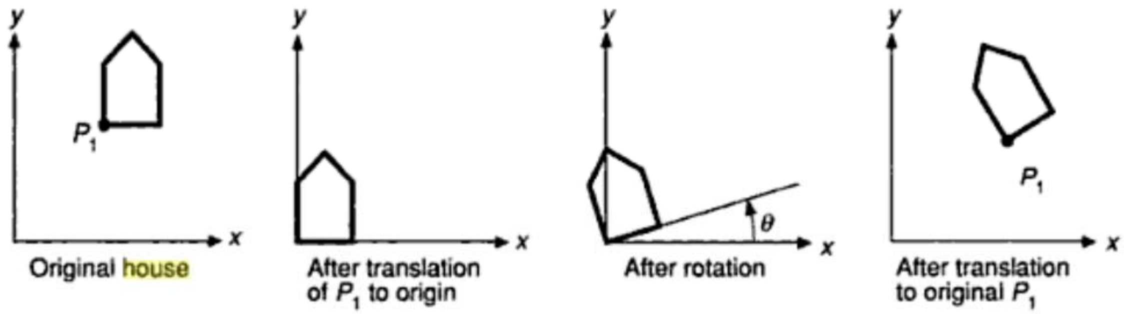
$$\begin{bmatrix} 1 & & d_{x2} \\ & 1 & d_{y2} \\ & & 1 \end{bmatrix} \bullet \begin{bmatrix} 1 & & d_{x1} \\ & 1 & d_{y1} \\ & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & d_{x1} + d_{x2} \\ & 1 & d_{y1} + d_{y2} \\ & & 1 \end{bmatrix} \qquad \begin{bmatrix} s_{x2} & & \\ & s_{y2} & \\ & & 1 \end{bmatrix} \bullet \begin{bmatrix} s_{x1} & & \\ & s_{y2} & \\ & & 1 \end{bmatrix} = \begin{bmatrix} s_{x1} \cdot s_{x2} & & \\ & s_{y1} \cdot s_{y2} & \\ & & 1 \end{bmatrix}$$

- Rigid Transformation
  - ← product of an arbitrary sequence of rotation and translation matrices
  - preserving length, angles
- Affine transformation
  - ← product of an arbitrary sequence of rotation, translation, scale, and shear matrices
  - preserving parallelism

Composition of Transformations (cont.)

- Combine fundamental R, S, and T matrices → produce desired general affine transformation
- Gain efficiency by applying a single composed transformation, rather than a series of transformations

Examples: Rotating a house about a point P_1 by an angle

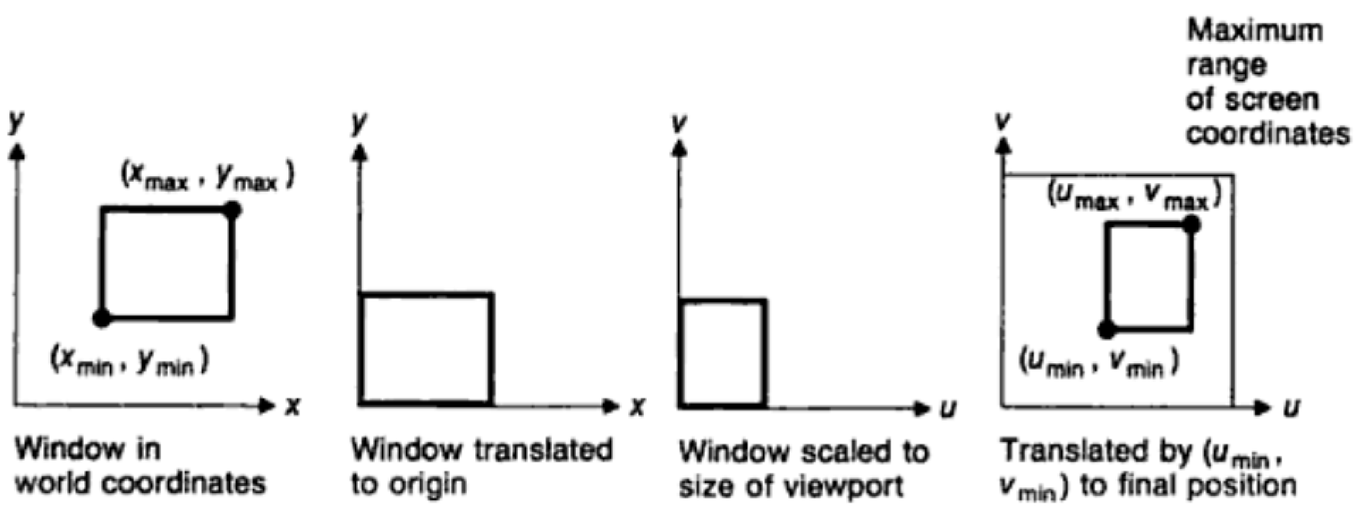


$$T(x_1, y_1) \cdot R(\theta) \cdot T(-x_1, -y_1) = \begin{bmatrix} 1 & 0 & x_1 \\ 0 & 1 & y_1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_1 \\ 0 & 1 & -y_1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_1(1 - \cos\theta) + y_1\sin\theta \\ \sin\theta & \cos\theta & y_1(1 - \cos\theta) - x_1\sin\theta \\ 0 & 0 & 1 \end{bmatrix}.$$

2D Window-To-Viewport Transformation

- We get objects (2D) represented in world-coordinate system
- We need to map them onto screen coordinates
- A common question:
  - Given a rectangular region in world coordinates (world-coordinate window)
  - A corresponding rectangular region in screen coordinates (viewport)
  - To find the transformation



Sometimes, we want shape-preserving: make $s_u = s_v$