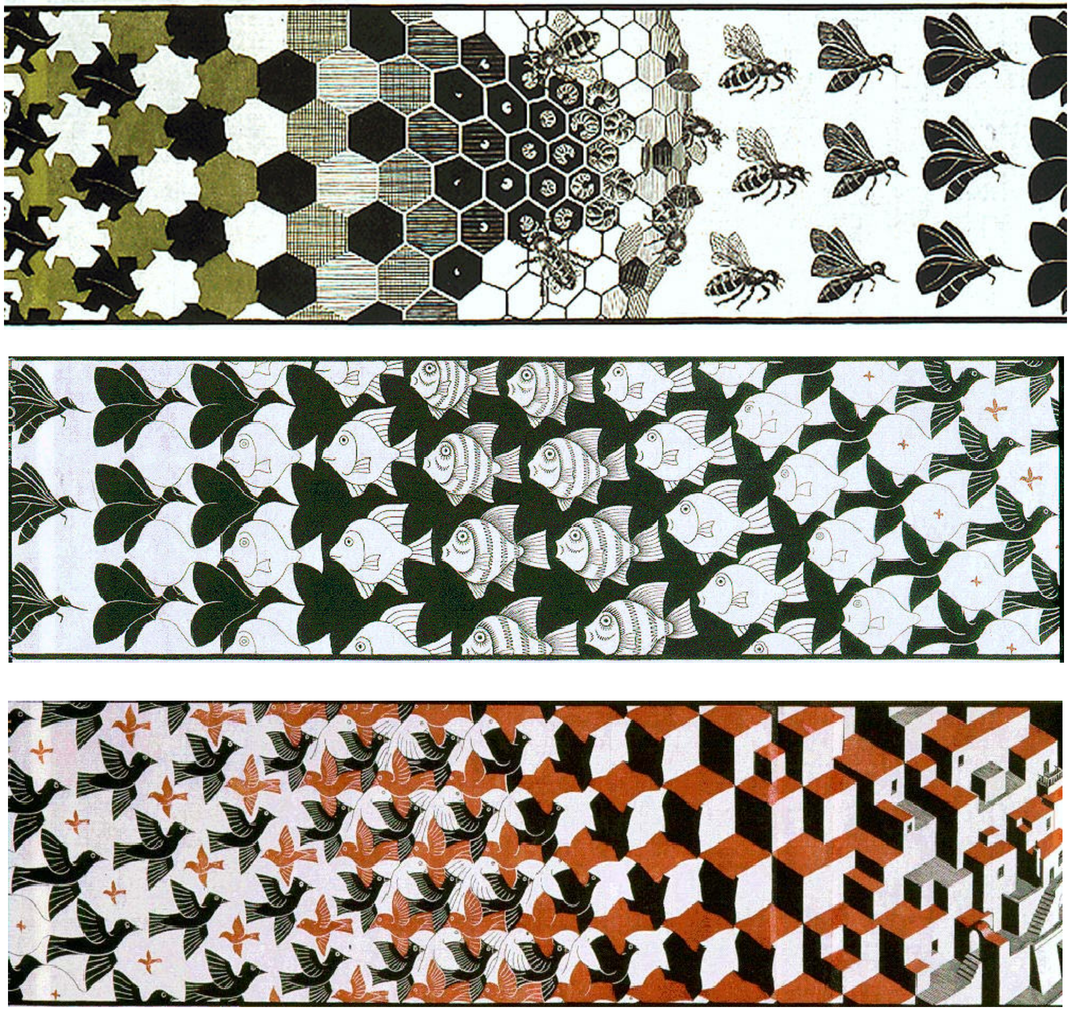
# 2D Geometrical Transformations

Foley & Van Dam, Chapter 5



# 2D Geometrical Transformations

- Translation
- Scaling
- Rotation
- Shear
- Matrix notation
- Compositions
- Homogeneous coordinates

# 2D Geometrical Transformations

**Assumption:** Objects consist of points and lines.
A point is represented by its Cartesian coordinates:

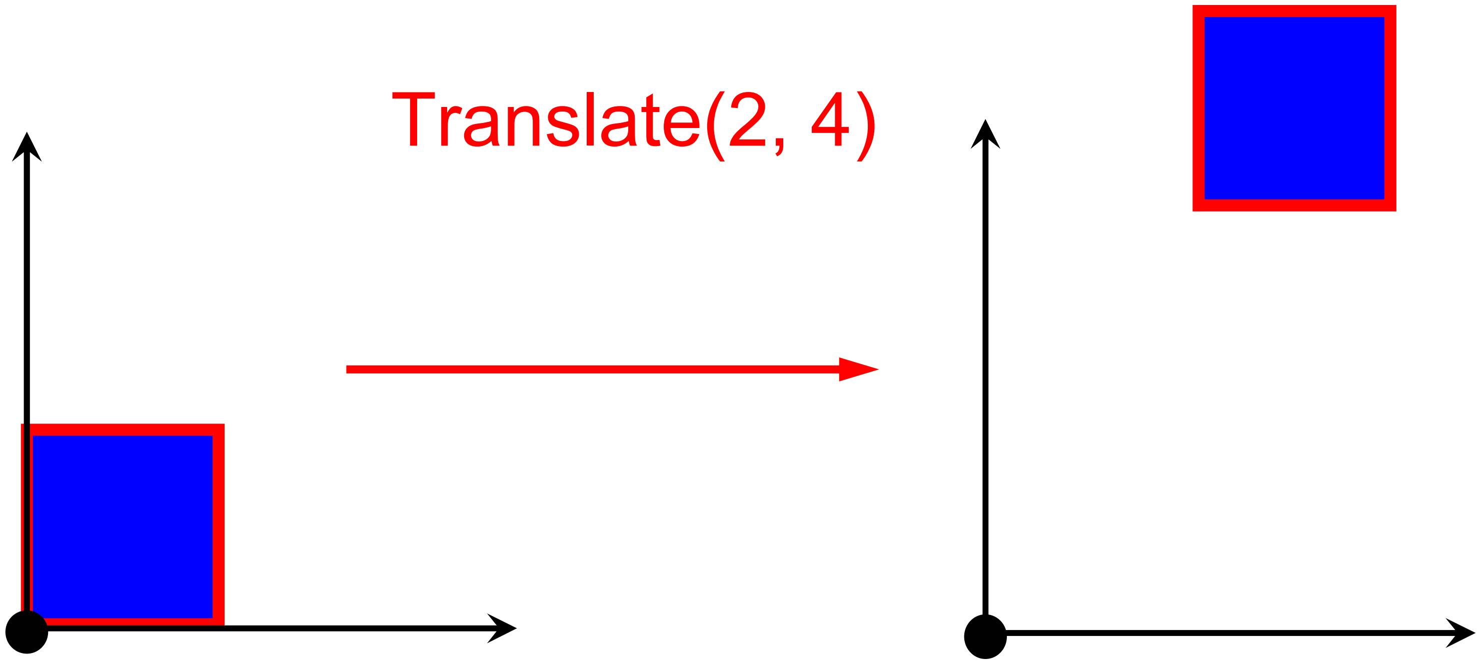
$$P = (x, y)$$

**Geometrical Transformation:**
Let $(A, B)$ be a straight line segment between the points A and B.
Let T be a general 2D transformation.
T transforms $(A, B)$ into another straight line segment $(A', B')$, where:

$$A' = TA \text{ and}$$
$$B' = TB$$

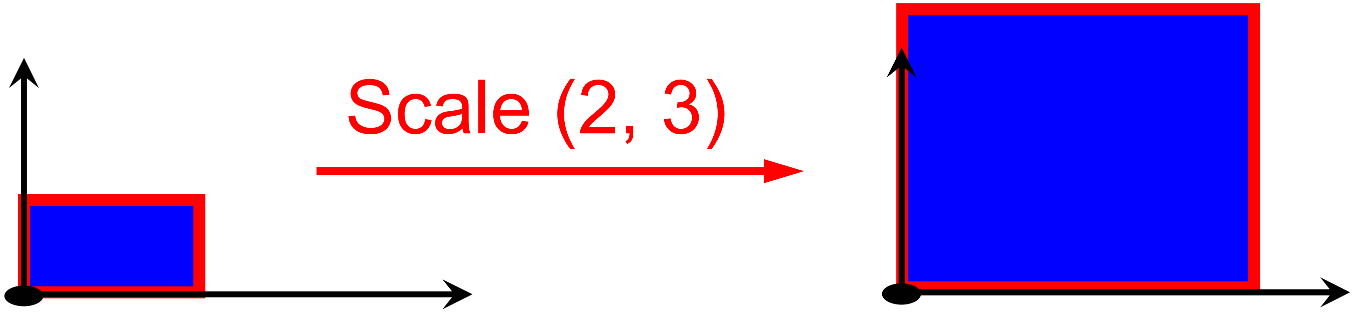
# Translation

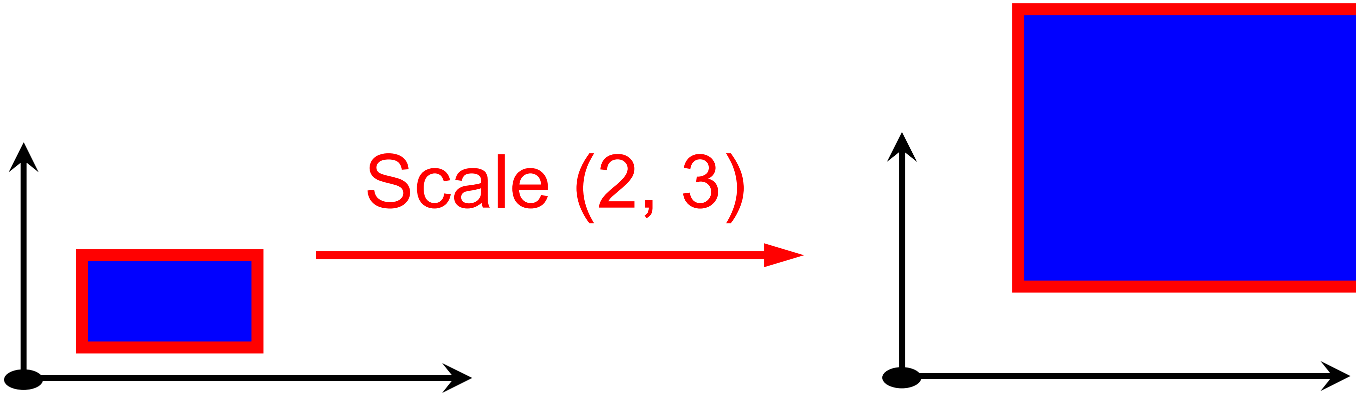
- Translate(a, b): $(x, y) \rightarrow (x+a, y+b)$



# Scale

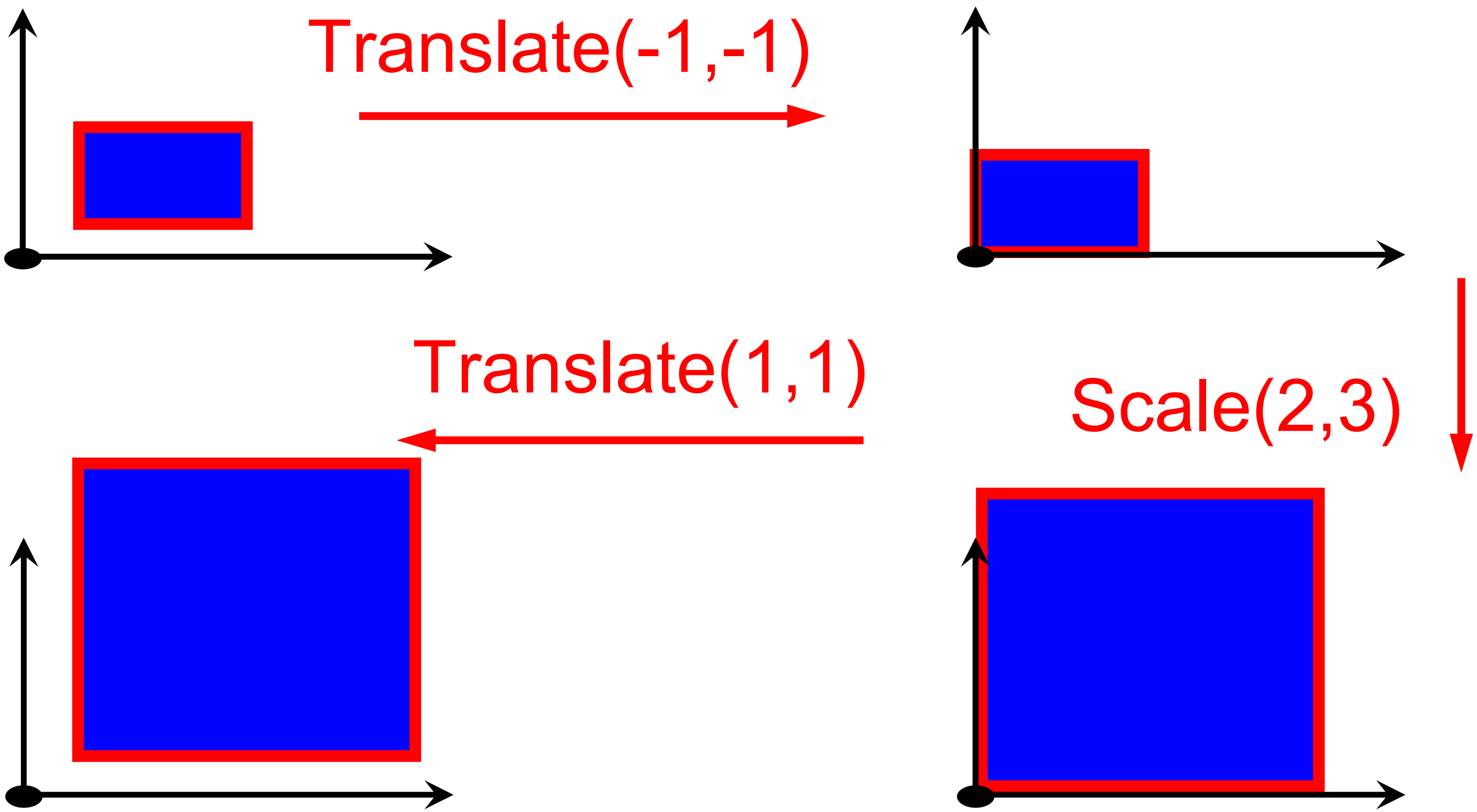
- Scale (a, b): $(x, y) \rightarrow (ax, by)$



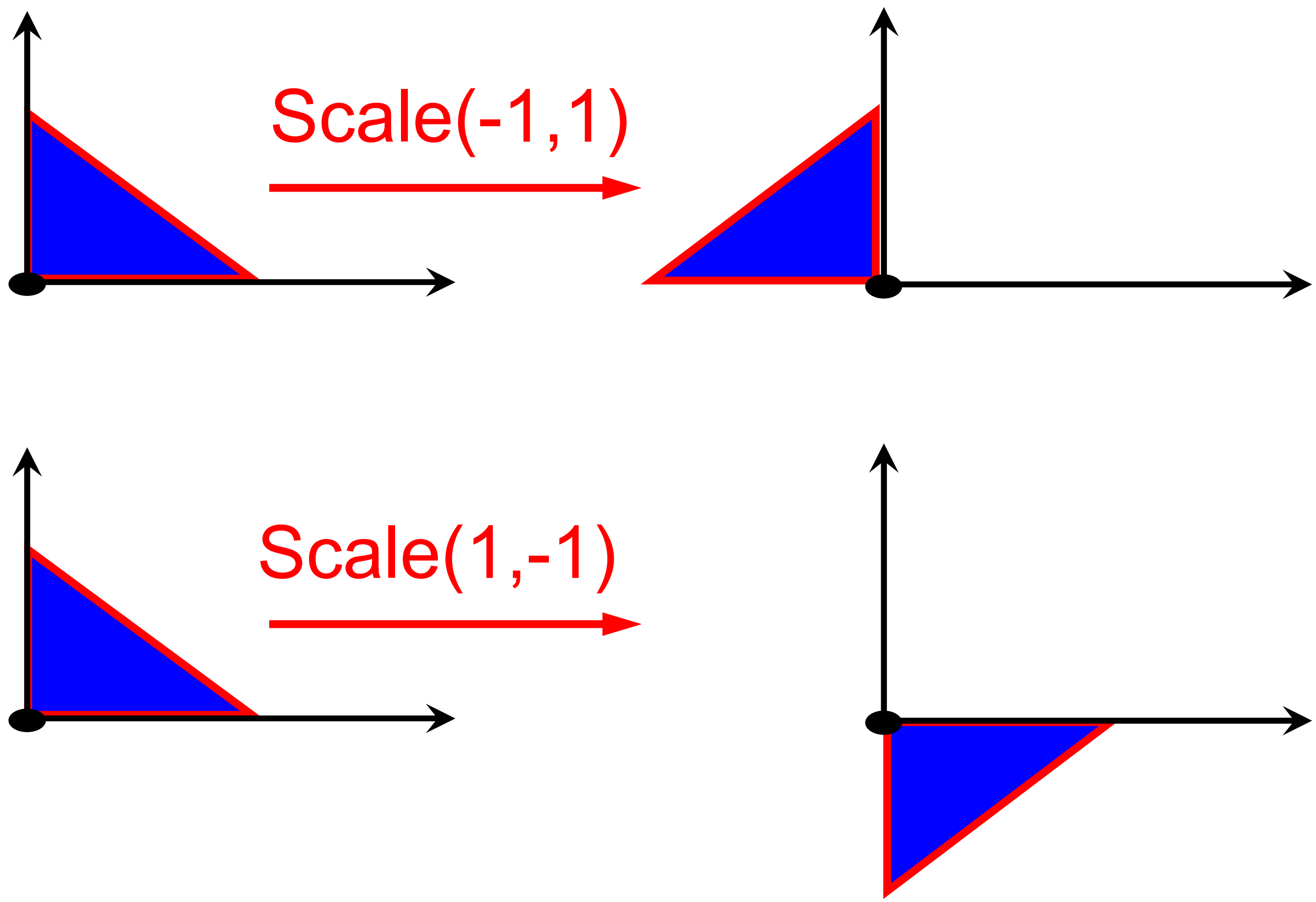
# Scale

- How can we scale an object without moving its origin (lower left corner)?



# Reflection

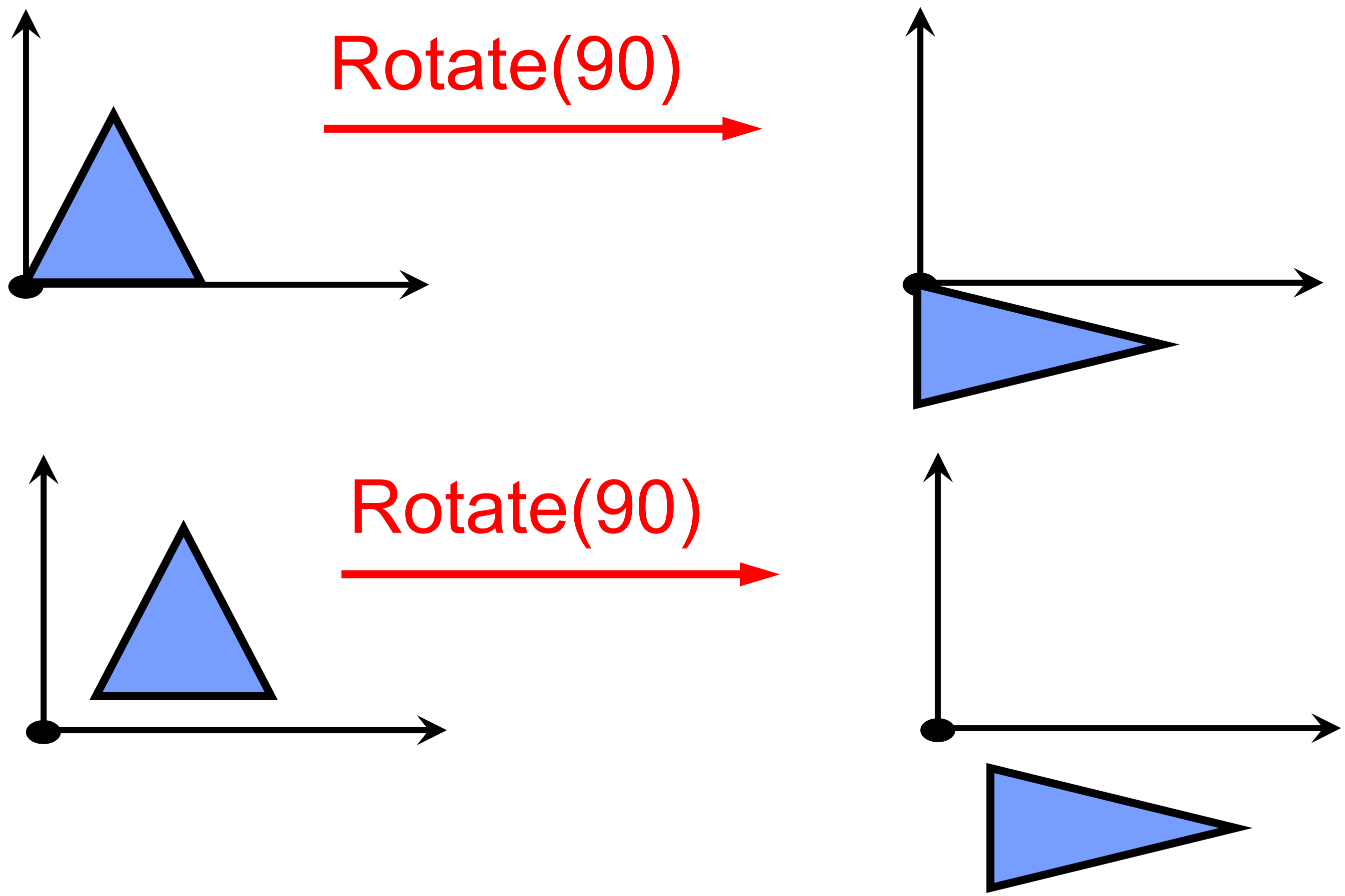
- Special case of scale



# Rotation

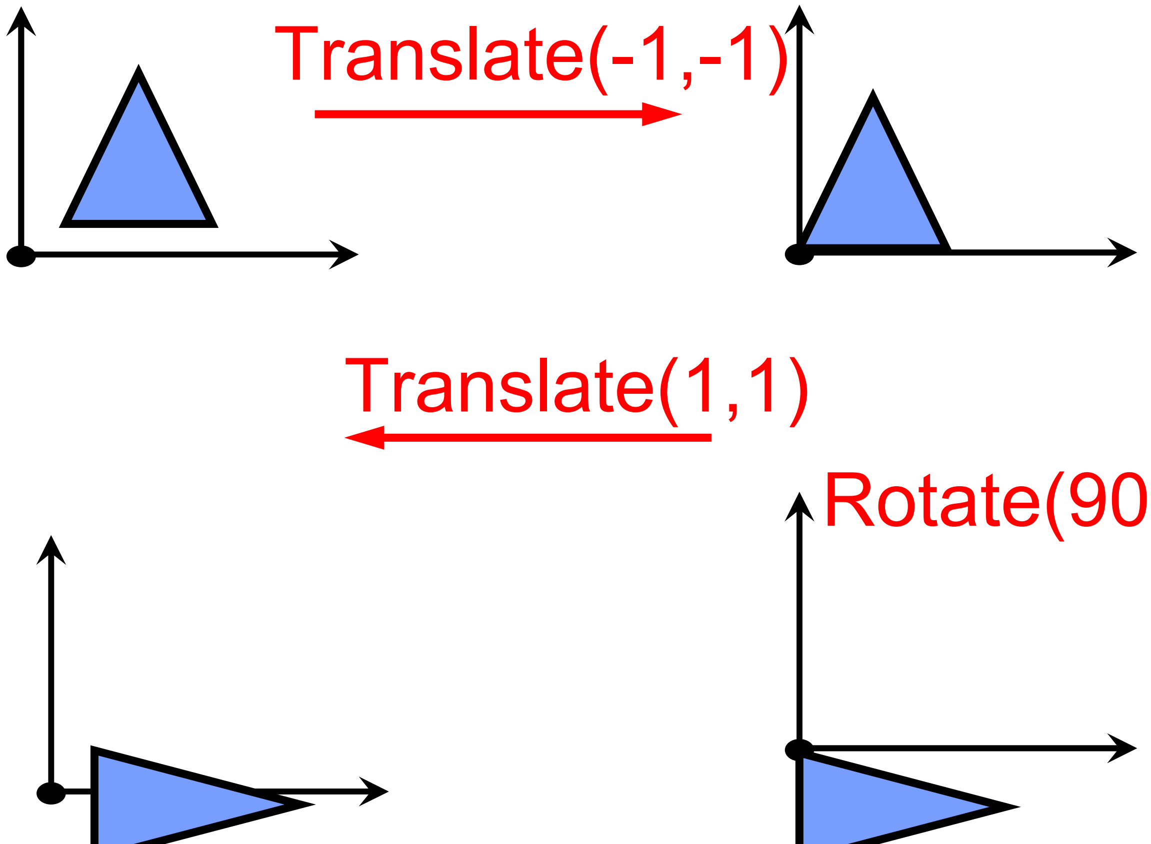
- Rotate($\theta$):

$$(x, y) \rightarrow (x \cos(\theta)+y \sin(\theta), -x \sin(\theta)+y \cos(\theta))$$



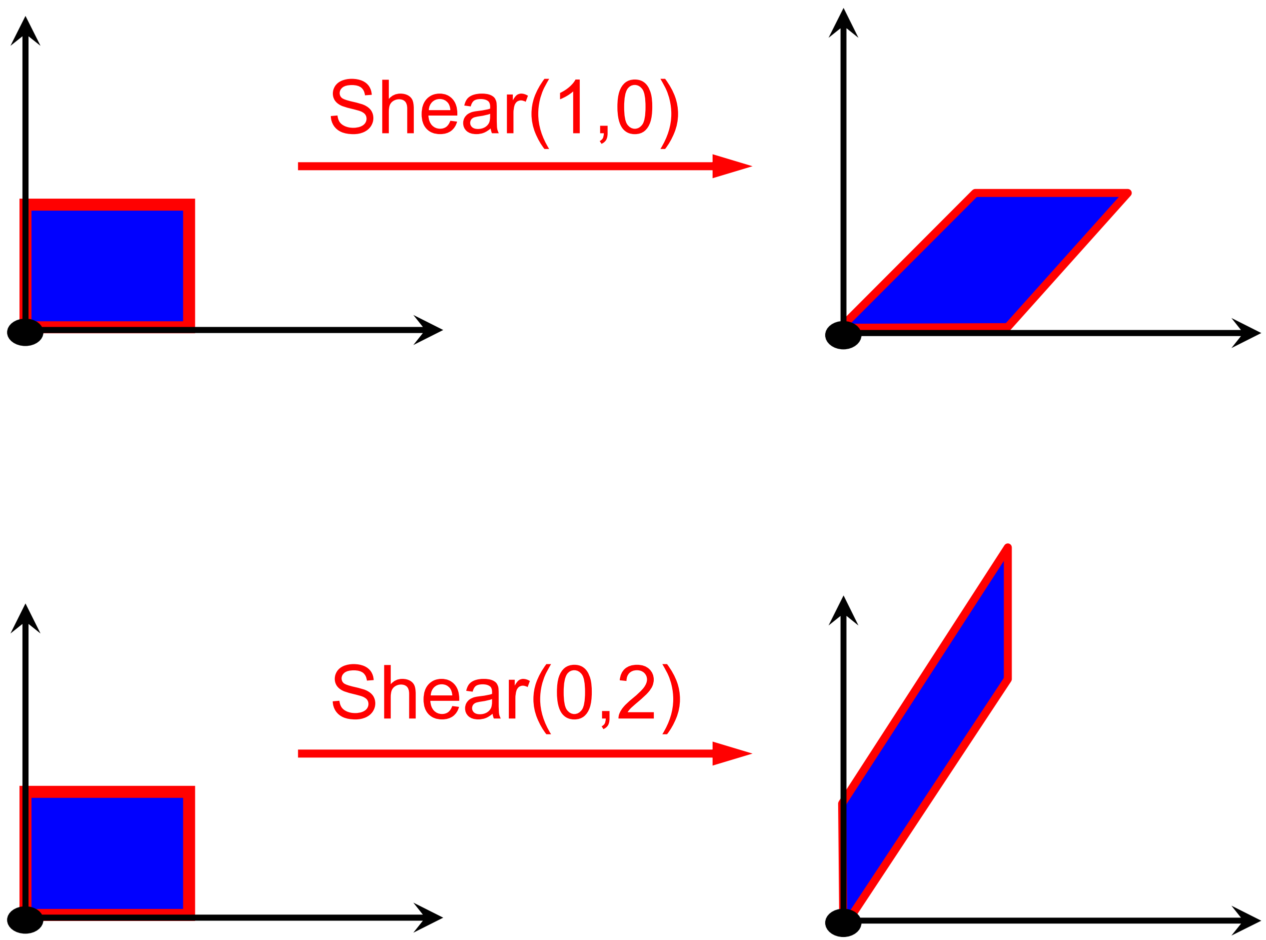
# Rotation

- How can we rotate an object without moving its origin (lower left corner)?



# Shear

- Shear (a, b): $(x, y) \rightarrow (x+ay, y+bx)$

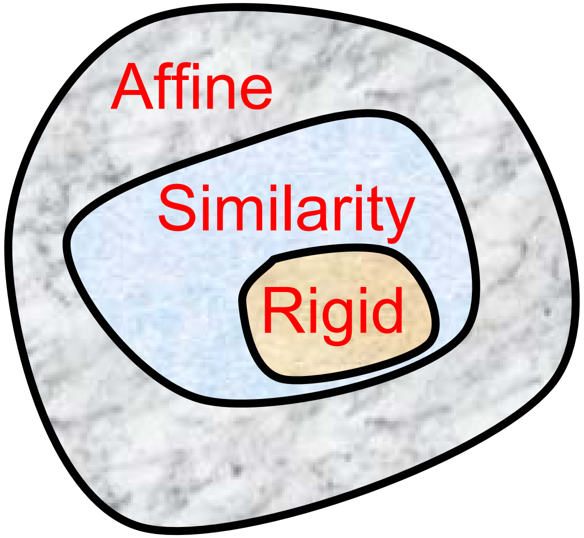


# Classes of Transformations

- Rigid transformation (distance preserving):
  Translation + Rotation
- Similarity transformation (angle preserving):
  Translation + Rotation + Uniform Scale
  - Affine transformation (parallelism preserving):
    Translation + Rotation + Scale + Shear

All above transformations are groups where

Rigid $\subset$ Similarity $\subset$ Affine



# Matrix Notation

- Let's treat a point ($x$, $y$) as a 2x1 matrix (column vector):

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

- What happens when this vector is multiplied by a 2x2 matrix?

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax + by \\ cx + dy \end{bmatrix}$$

# 2D Transformations

- 2D object is represented by points and lines that join them

- Transformations can be applied only to the the points defining the lines

- A point (*x*, *y*) is represented by a 2x1 column vector, so we can represent 2D transformations by using 2x2 matrices:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

# Scale

- Scale (a, b): (x, y) $\rightarrow$ (ax, by)

$$\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

- If *a* or *b* are negative, we get reflection

- Inverse: $S^{-1}(a,b) = S(1/a, 1/b)$

# Reflection

- Reflection through the *y* axis:

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

- Reflection through the *x* axis:

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

- Reflection through $y = x$:

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

- Reflection through $y = -x$:

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

# Shear

- Shear (a, b): $(x, y) \rightarrow (x+ay,\ y+bx)$

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

# Rotation

- Rotate($\theta$):

(x, y) $\rightarrow$ (x cos($\theta$)+y sin($\theta$), -x sin($\theta$)+y cos($\theta$))

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

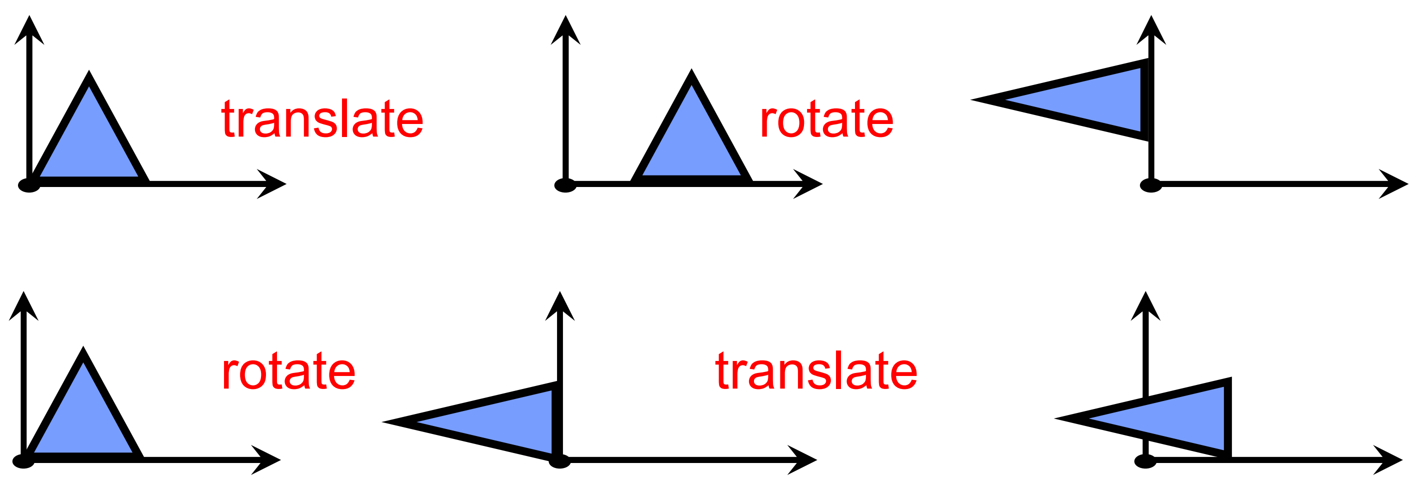
- Inverse: $R^{-1}(q) = R^{T}(q) = R(-q)$

# Composition of Transformations

- A sequence of transformations can be collapsed into a single matrix:

$$\begin{bmatrix} A \end{bmatrix}\begin{bmatrix} B \end{bmatrix}\begin{bmatrix} C \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} D \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

- Note: Order of transformations is important!



# Translation

- Translation (a, b):

$$\begin{bmatrix} x \\ y \end{bmatrix} \to \begin{bmatrix} x + a \\ y + b \end{bmatrix}$$

**Problem:** Cannot represent translation using 2x2 matrices

**Solution:**

Homogeneous Coordinates

# Homogeneous Coordinates

Is a mapping from $R^n$ to $R^{n+1}$:

$$(x, y) \rightarrow (X, Y, W) = (tx, ty, t)$$

Note: All triples ($tx$, $ty$, $t$) correspond to the same non-homogeneous point ($x$, $y$)
Example (2, 3, 1) $\equiv$ (6, 9, 3).

Inverse mapping:

$$(X, Y, W) \rightarrow \left( \frac{X}{W}, \frac{Y}{W} \right)$$

# Translation

- Translate(a, b):

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

Inverse: $T^{-1}(a, b) = T(-a, -b)$

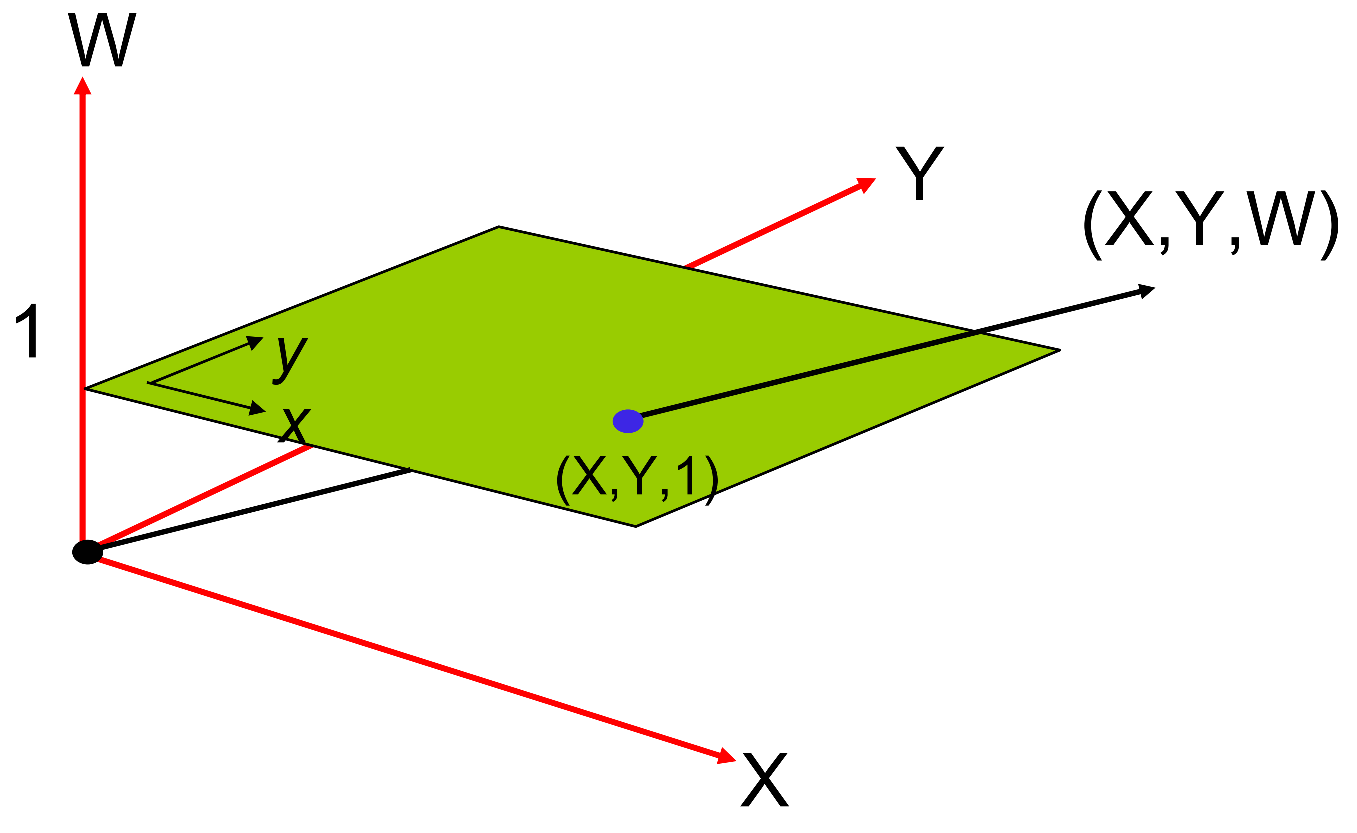
Affine transformations now have the following form:

$$\begin{bmatrix} a & b & e \\ c & d & f \\ 0 & 0 & 1 \end{bmatrix}$$

# Geometric Interpretation

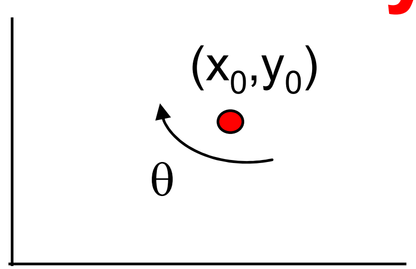
A 2D point is mapped to a line (ray) in 3D

The non-homogeneous points are obtained by projecting the rays onto the plane Z=1



# Example

## Rotation about an arbitrary point



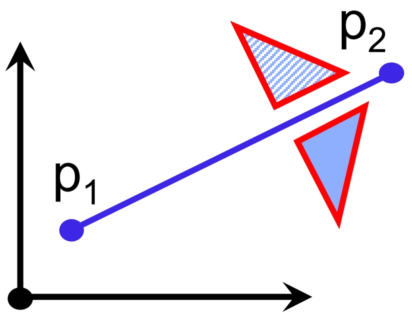

1. Translate the coordinates so that the origin is at $(x_0,y_0)$
2. Rotate by $\theta$
3. Translate back

$$\begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_0(1-\cos\theta)+y_0\sin\theta \\ \sin\theta & \cos\theta & y_0(1-\cos\theta)-x_0\sin\theta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

# Example

## Reflection about an arbitrary line



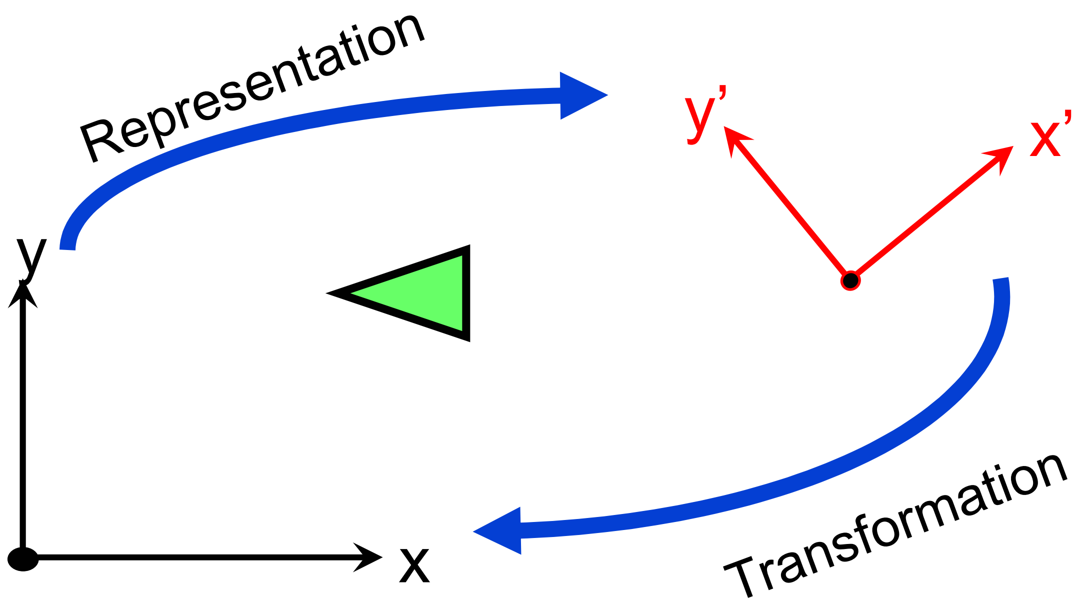

$$L = p_1 + t\,(p_2 - p_1) = t\,p_2 + (1-t)\,p_1$$

1. Translate the coordinates so that $P_1$ is at the origin
2. Rotate so that L aligns with the x-axis
3. Reflect about the x-axis
4. Rotate back
5. Translate back

# Change of Coordinates

It is often required to transform the description of an object from one coordinate system to another

**Rule:** Transform one coordinate frame towards the other in the opposite direction of the representation change



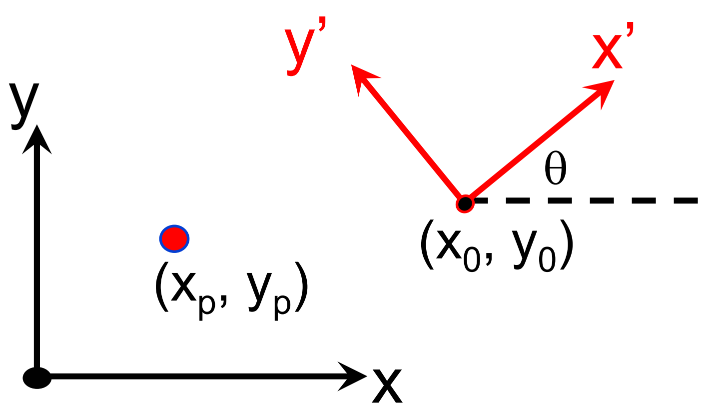
# Example

- **Change of coordinates:** Represent P = ($x_p$, $y_p$, 1) in the (x', y') coordinate system

$$P' = MP$$

Where:

$$M = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{pmatrix}$$



# Example

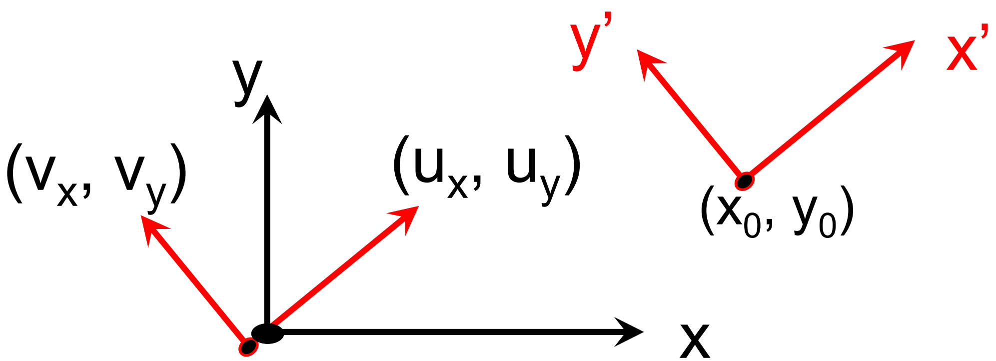
- **Change of coordinates:**

Alternative method: assume x' = (ux, uy) and y' = (vx, vy) in the (x, y) coordinate system

$$P' = MP$$

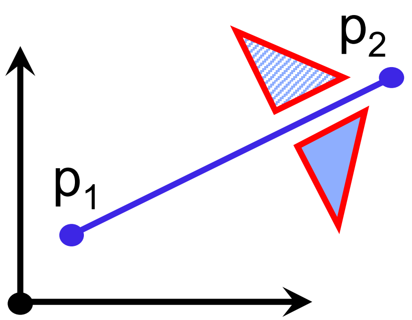
where

$$M = \begin{pmatrix} u_x & u_y & 0 \\ v_x & v_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{pmatrix}$$



# Example

## Reflection about an arbitrary line




$$L = p_1 + t\,(p_2 - p_1) = t\,p_2 + (1-t)\,p_1$$

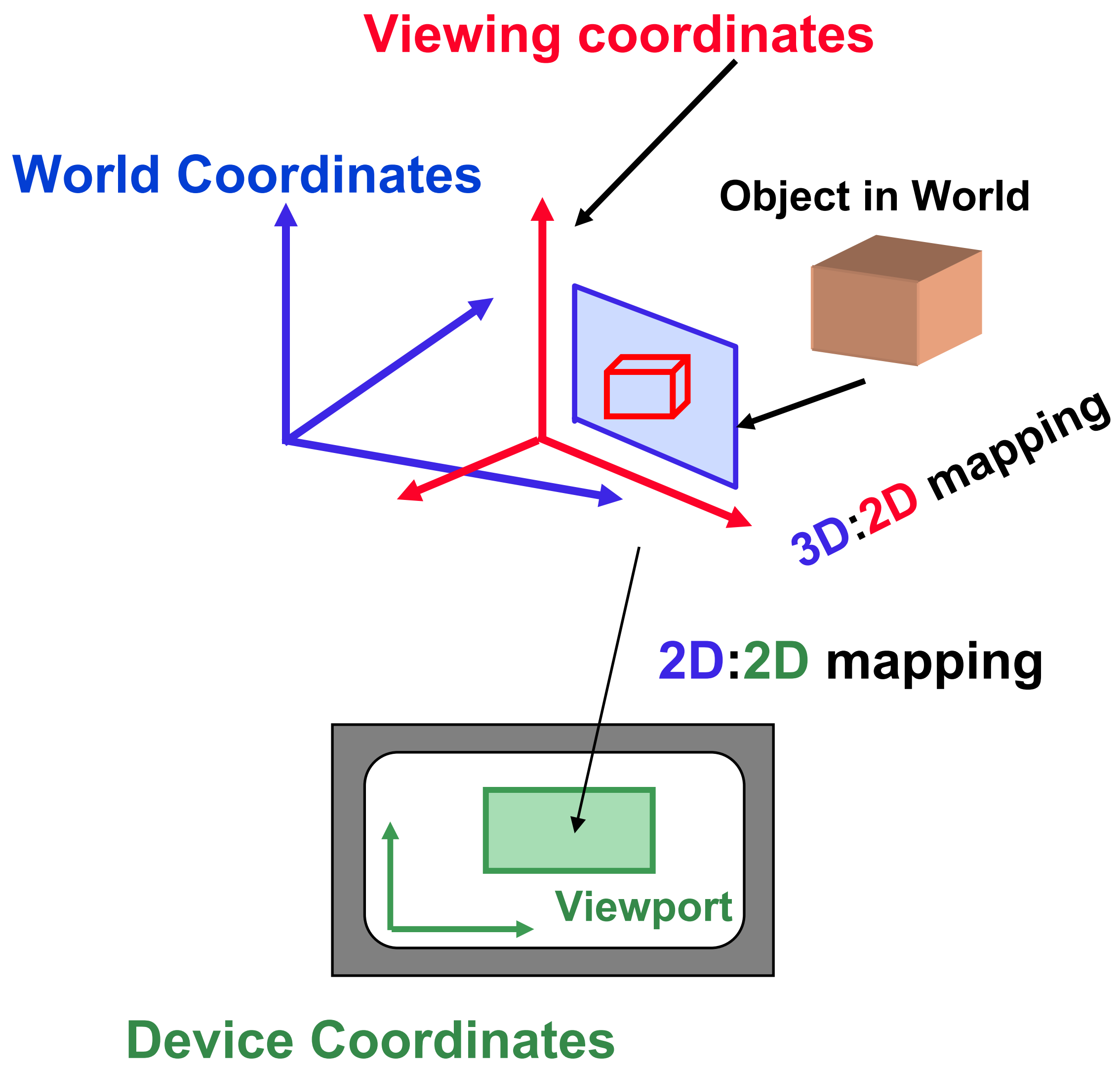
Define a coordinate systems (u, v) parallel to $P_1P_2$:

$$\mathbf{u} = \frac{p_2 - p_1}{|p_2 - p_1|} \equiv \begin{pmatrix} u_x \\ u_y \end{pmatrix}$$

$$\mathbf{v} = \begin{pmatrix} -u_y \\ u_x \end{pmatrix} = \begin{pmatrix} v_x \\ v_y \end{pmatrix}$$

$$M = \begin{pmatrix} 1 & 0 & p_{1_x} \\ 0 & 1 & p_{1_y} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_x & v_x & 0 \\ u_y & v_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u_x & u_y & 0 \\ v_x & v_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -p_{1_x} \\ 0 & 1 & -p_{1_y} \\ 0 & 0 & 1 \end{pmatrix}$$

# 3D Viewing Transformation Pipeline

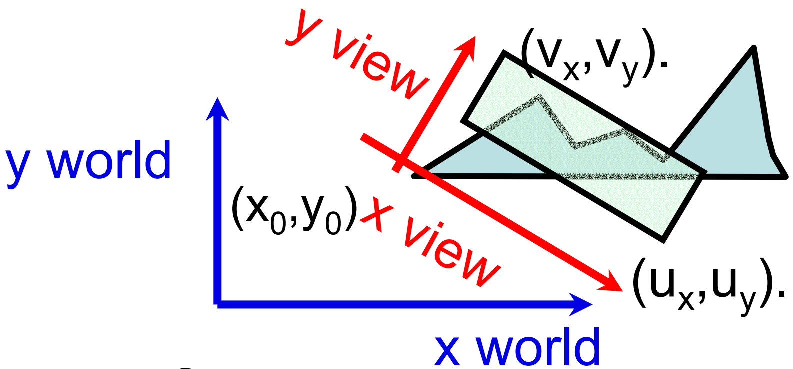


# World to Viewing Coordinates

In order to define the viewing window we have to specify:

- Windowing-coordinate **origin** $P_0 = (x_0, y_0)$
- View **vector up** $\mathbf{v} = (v_x, v_y)$

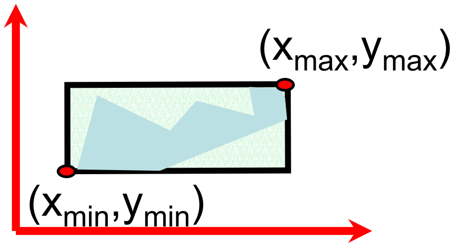
•Using **v**, we can find **u**: $\mathbf{u} = \mathbf{v} \times (0,0,1)$



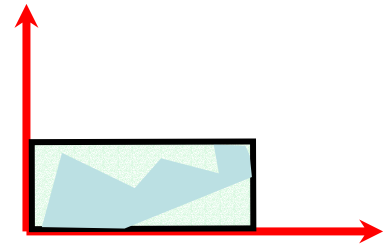

Transformation from world to viewing coordinates :

$$M_{wc-vc} = \begin{pmatrix} u_x & u_y & 0 \\ v_x & v_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -x_0 \\ 0 & 1 & -y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

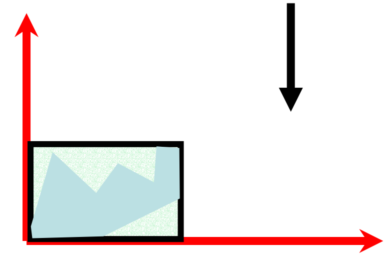
# Window to Viewport Coordinates



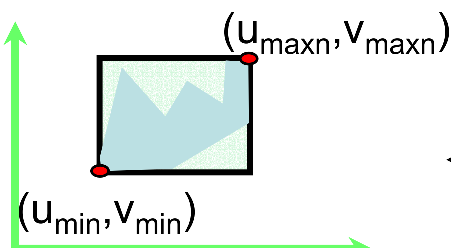

Window is Viewing Coordinates



Window translated to origin



Window scaled to Normalized Viewport size




Window scaled and translated to Viewport location in device coordinates

$$M_{vc-dc}=\begin{pmatrix}1 & 0 & u_{\min}\\ 0 & 1 & v_{\min}\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}u_{\max}-u_{\min} & 0 & 0\\ 0 & v_{\max}-v_{\min} & 0\\ 0 & 0 & 1\end{pmatrix}\underbrace{\begin{pmatrix}\dfrac{1}{x_{\max}-x_{\min}} & 0 & 0\\ 0 & \dfrac{1}{y_{\max}-y_{\min}} & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & -x_{\min}\\ 0 & 1 & -y_{\min}\\ 0 & 0 & 1\end{pmatrix}}_{\text{Normalized Device Coordinates}}$$

# Efficiency Considerations

A 2D point transformation requires 9 multiplies and 6 adds

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} ax + by + cz \\ dx + ey + fz \\ gx + hy + iz \end{bmatrix}$$

But since affine transformations have always the form:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$

The number of operations can be reduced to 4 multiplies and 4 adds

# Efficiency Considerations

The rotation matrix is:

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

When rotating of small angles θ, we can use the fact that *cos*(θ) ≅ 1 and simplify

$$\begin{bmatrix} 1 & \sin\theta \\ -\sin\theta & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + y\sin\theta \\ -x\sin\theta + y \end{bmatrix}$$

# Determinant of a Matrix

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc$$

$$\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix} = aei + bfg + cdh - ceg - afh - bdi$$

$$= a\begin{vmatrix} e & f \\ h & i \end{vmatrix} - b\begin{vmatrix} d & f \\ g & i \end{vmatrix} + c\begin{vmatrix} d & e \\ g & h \end{vmatrix}$$

If P is a polygon of area $A_P$, transformed by a matrix M, the area of the transformed polygon is $A_P * |M|$