Lecture 3: Transformations 1

CS 174A: Introduction to Computer Graphics

Lecture 2 - Recap

- Vectors,
  - Vector Basics
  - Vector Addition, Subtraction
  - Vector Multiplication
    - Dot product
    - Cross product
  - Constructing Coordinate Frames
- Matrix
  - Matrix Basics
  - Matrix Addition
  - Matrix Multiplication
  - Matrix Properties
- Points vs Vectors
- Lines and Planes

Motivation

- Many graphics concepts need basic math, such as linear algebra
  - Vectors (dot products, cross products,…)
  - Matrices (matrix-matrix, matrix-vector mult., …)
  - Operations like translation, or rotation of the points that form an object are most efficiently performed as a matrix-vector multiply
  - Thus, we will go out of our way to write everything as a matrix-vector multiply
- Chapters 2.4 (vectors) and 5.2 (matrices)
  - *Fundamentals of Computer Graphics* by Shirley and Marschner
  - Worthwhile to read all of chapters 2 and 5
- Should be refresher on very basic material for most of you

Vectors: Addition and Multiplication

- Addition $\mathbf{x}+\mathbf{y}=[x_1+y_1 \quad \cdots \quad x_n+y_n \quad \cdots \quad x_N+y_N]^T \quad 1 \le n \le N$
- Multiplication with scalar (scaling)

$$a\mathbf{x}=[ax_1 \quad \cdots \quad ax_n \quad \cdots \quad ax_N]^T \quad a, x_n \in \Re$$

- Properties

$\mathbf{u}+\mathbf{v}=\mathbf{v}+\mathbf{u}$	commutative
$(\mathbf{u}+\mathbf{v})+\mathbf{w}=\mathbf{u}+(\mathbf{v}+\mathbf{w})$	associative
$a(\mathbf{u}+\mathbf{v})=a\mathbf{u}+a\mathbf{v}, \quad a \in \Re$	distributive
$\mathbf{u}-\mathbf{u}=0$	inverse

Special Cases

- Linear combination
  - $\mathbf{w} = a_1\mathbf{v}_1 + \ldots a_m\mathbf{v}_m$, $a_1,\ldots,a_m$ in R
- Affine combination:
  - A linear combination for which $a_1+\ldots+a_m = 1$
- Convex combination
  - An affine combination for which $a_i \geq 0$ for i = 1,…,m
- Vectors $\mathbf{v}_1, \ldots, \mathbf{v}_m$ are called *linearly independent,* if

$a_1\mathbf{v}_1+\ldots+a_m\mathbf{v}_m = 0$ ***iff*** $a_1 = a_2 = \ldots = a_m = 0$

Vectors: Dot (Scalar) Product

Motivation:

- Find angle between two vectors
  - e.g. cosine of angle between light source and surface for shading
- Finding projection of one vector on another
  - e.g. coordinates of a point in various coordinate frameworks -- object coordinate framework, world coordinate framework, camera coordinate framework, image coordinate framework
- Advantage: computed easily in cartesian components

Dot Product

Definition: $\mathbf{a}, \mathbf{b} \in \Re^N$

$$\mathbf{a} \cdot \mathbf{b} = \sum_{n=1}^{N} a_n b_n = \|\mathbf{a}\|\|\mathbf{b}\|\cos\theta$$

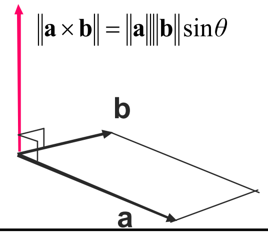
Properties

1. Symmetry: $\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a}$
2. Linearity: $(\mathbf{a} + \mathbf{b}) \cdot \mathbf{c} = \mathbf{a} \cdot \mathbf{c} + \mathbf{b} \cdot \mathbf{c}$
3. Homogeneity: $(s\mathbf{a}) \cdot \mathbf{b} = s(\mathbf{a} \cdot \mathbf{b})$
4. $|\mathbf{b}|^2 = \mathbf{b} \cdot \mathbf{b}$
5. $\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}|\cos(\theta)$

Cross (vector) product

- Motivation: Useful in constructing coordinate systems (later)
  - Problem: Construct a orthonormal camera coordinate frame into which to transform world objects, given **a** (viewing direction), and a second vector **b** (up direction of camera)
- Cross product is a vector orthogonal to the two initial vectors
- Direction determined by *right-hand rule*

$$\|\mathbf{a} \times \mathbf{b}\| = \|\mathbf{a}\|\|\mathbf{b}\|\sin\theta$$

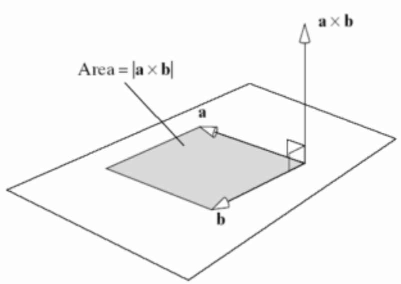


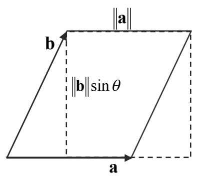
Cross Product

Geometric Definition: $\|\mathbf{a} \times \mathbf{b}\| = \|\mathbf{a}\|\|\mathbf{b}\| \sin \theta$

$\|\mathbf{a} \times \mathbf{b}\|$ parallelogram area

$\mathbf{a} \times \mathbf{b}$ perpendicular to parallelogram





Cross Product

Defined only for 3D vectors and with respect to the standard unit vectors

Algebraic Definition:

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix}$$

$$\mathbf{a} \times \mathbf{b} = (a_2 b_3 - a_3 b_2)\mathbf{i} + (a_3 b_1 - a_1 b_3)\mathbf{j} + (a_1 b_2 - a_2 b_1)\mathbf{k}$$

Properties of the Cross Product

1. $\mathbf{i} \times \mathbf{j} = \mathbf{k}, \quad \mathbf{i} \times \mathbf{k} = -\mathbf{j}, \quad \mathbf{j} \times \mathbf{k} = \mathbf{i}$
2. Antisymmetry: $\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$
3. Linearity: $\mathbf{a} \times (\mathbf{b} + \mathbf{c}) = \mathbf{a} \times \mathbf{b} + \mathbf{a} \times \mathbf{c}$
4. Homogeneity: $(s\mathbf{a}) \times \mathbf{b} = s(\mathbf{a} \times \mathbf{b})$
5. The cross product is normal to both vectors: $(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{a} = (\mathbf{a} \times \mathbf{b}) \cdot \mathbf{b} = 0$
6. $|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}||\mathbf{b}| \sin(\theta)$

Generators and Base Vectors

How many vectors are needed to generate a vector space?

- Given a vector space $\mathbf{R}^n$ we can prove that we need minimum n vectors to generate all vectors $\mathbf{v}$ in $\mathbf{R}^n$
- Any set of vectors that generate a vector space is called a ***generator set***
- A generator set with minimum size is called a ***basis*** for the given vector space

Matrix Addition

Addition:

$$\mathbf{A}_{m\times n} + \mathbf{B}_{m\times n} = (a_{ij} + b_{ij})$$

Properties:

1. $\mathbf{A}+\mathbf{B}=\mathbf{B}+\mathbf{A}$
2. $\mathbf{A}+(\mathbf{B}+\mathbf{C})=(\mathbf{A}+\mathbf{B})+\mathbf{C}$
3. $f(\mathbf{A}+\mathbf{B})=f\mathbf{A}+f\mathbf{B}$
4. Transpose: $\mathbf{A}^T=(a_{ij})^T=(a_{ji})$

Matrix Multiplication

Definition:

$$\mathbf{C}_{m\times r} = \mathbf{A}_{m\times n}\mathbf{B}_{n\times r}$$

$$(\mathbf{C}_{ij}) = (\sum_{k=1}^{n} a_{ik}b_{kj})$$

Properties:

1. $\mathbf{AB}\neq\mathbf{BA}$ — **Matrix** order matters
2. $\mathbf{A}(\mathbf{BC})=(\mathbf{AB})\mathbf{C}$ — **Multiplication** order does NOT matter
3. $f(\mathbf{AB})=(f\mathbf{A})\mathbf{B}$
4. $\mathbf{A}(\mathbf{B}+\mathbf{C})=\mathbf{AB}+\mathbf{AC}$,
   $(\mathbf{B}+\mathbf{C})\mathbf{A}=\mathbf{BA}+\mathbf{CA}$
5. $(\mathbf{AB})^T=\mathbf{B}^T\mathbf{A}^T$
6. $(\mathbf{AB})^{-1}=\mathbf{B}^{-1}\mathbf{A}^{-1}$

Matrix Order Matters For Multiplication

$$\mathbf{AB}\neq\mathbf{BA}$$

- Example

$$\begin{bmatrix}2 & 4\\ -1 & 3\end{bmatrix}\begin{bmatrix}-5 & -3\\ 2 & 2\end{bmatrix}=$$

$$\begin{bmatrix}-5 & -3\\ 2 & 2\end{bmatrix}\begin{bmatrix}2 & 4\\ -1 & 3\end{bmatrix}=$$

Multiplication Order Does NOT Matter

- Note the order of the matrices has not changed only which multiplication is performed first has changed

$$(\mathbf{AB})\mathbf{C}=\mathbf{A}(\mathbf{BC})$$

- Example

$$\left(\begin{bmatrix}2 & 4\\ -1 & 3\end{bmatrix}\begin{bmatrix}-5 & -3\\ 2 & 2\end{bmatrix}\right)\begin{bmatrix}1 & 2\\ -3 & -1\end{bmatrix}=\begin{bmatrix}2 & 4\\ -1 & 3\end{bmatrix}\left(\begin{bmatrix}-5 & -3\\ 2 & 2\end{bmatrix}\begin{bmatrix}1 & 2\\ -3 & -1\end{bmatrix}\right)$$

Vector Multiplication in Matrix Form

- Dot product? $\mathbf{a}\cdot\mathbf{b} = \mathbf{a}^T\mathbf{b}$

$$\begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \left(a_1b_1 + a_2b_2 + a_3b_3\right) = \sum_{n=1}^{N} a_nb_n = \mathbf{a}\cdot\mathbf{b}$$

- Cross product?

$$\mathbf{a}\times\mathbf{b} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix} = \begin{bmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{bmatrix} = \mathbf{Ab} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

A is the dual matrix of vector **a**

Summary: Lines

Representations of a line (in 2D)

- Explicit - we have one dependent variable on the lefthand side of an equation, and all the independent variables and constants on the righthand side of the equation

$$y = \frac{dy}{dx}(x - x_0) + y_0$$

- Implicit - An implicit equation is an algebraic line/curve formed by points that satisfy an equation. Implicits TEST their *(x,y)* input values and return true if the implicit relationship is satisfied. The general solution of implicit equa tions requires a search!

$$F(x,y) = (x - x_0)dy - (y - y_0)dx$$

$$\text{if} \quad \begin{array}{lll} F(x,y) = 0 & \text{then} & (x,y) \text{ is on line} \\ F(x,y) > 0 & & (x,y) \text{ is below line} \\ F(x,y) < 0 & & (x,y) \text{ is above line} \end{array}$$

- Parametric - also called generating functions, they generate coordinate pairs as output, given parameter values as input.

$$x(t) = x_0 + t(x_1 - x_0)$$
$$y(t) = y_0 + t(y_1 - y_0)$$
$$t \in [0,1]$$

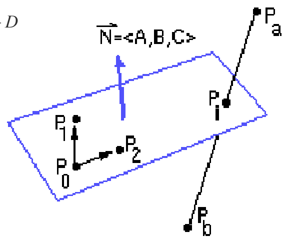
$$P(t) = P_0 + t(P_1 - P_0), \text{ or}$$
$$P(t) = (1-t)P_0 + tP_1$$

Summary: Planes

Plane equations

Implicit $F(x,y,z) = Ax + By + Cz + D = 0 \; = \mathbf{N}\bullet P + D$

Points on Plane $F(x,y,z) = 0$



Explicit $z = -(A/C)x - (B/C)y - D/C, \; C \neq 0$

Parametric

$Plane(s,t) = P_0 + s(P_1 - P_0) + t(P_2 - P_0)$

P_0, P_1, P_2 not collinear

or

$Plane(s,t) = P_0 + sV_1 + tV_2$ where V_1, V_2 are basis vectors

convex combination defines a triangle:

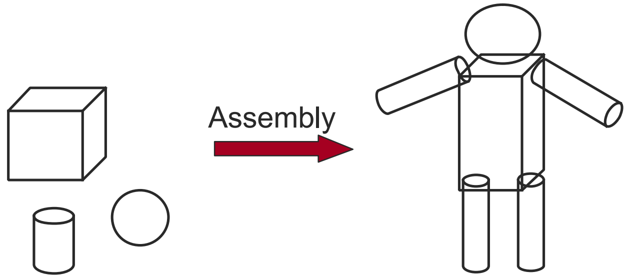
$Plane(s,t) = (1 - s - t)P_0 + sP_1 + tP_2$

Transformations

Modeling

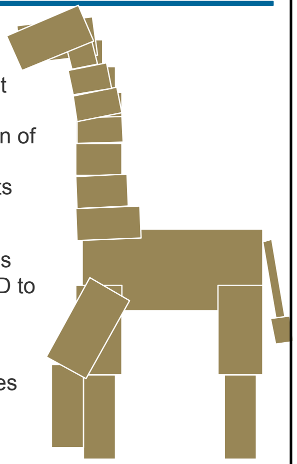
Geometric Primitives
- Points
- Lines
- Planes
- Polygons
- Parametric surfaces
- Implicit surfaces
- Etc.

Modeling Transformations



Why Study Transforms?

- Transforms are useful for modeling objects.
  - Transforms can be used place objects at correct location in the world
  - Transforms can describe relative position of connected body parts
  - Transforms can scale, rotate, etc. objects
- Viewing
  - World coordinates to camera coordinates
  - Parallel / perspective projections from 3D to 2D
- The math for doing all these things
  - Represent transformations using matrices and matrix-vector multiplications.
  - ch 6 -- worthwhile but not essential

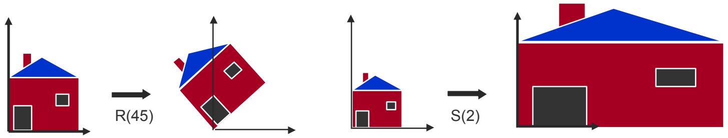


Lecture Outline

- ***2D Linear Transforms***
- Homogeneous Coordinates
- 2D Affine Transforms
- Composite Transforms

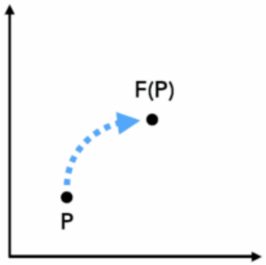
Transformations

- transforming an object = transforming all its points
- transforming a polygon = transforming its vertices



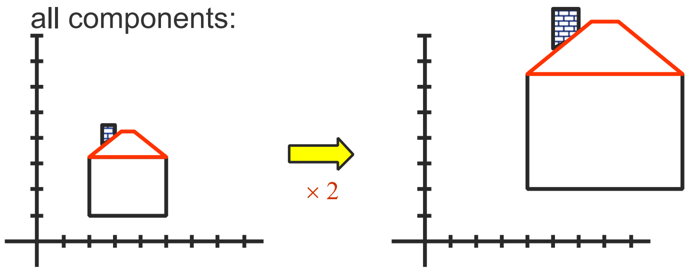
What Are Transforms?

- Just functions acting on points
- (x',y',z') = F(x,y,z)
- **P**' = F(**P**)



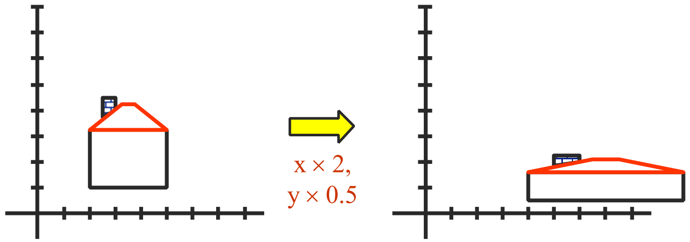
Scaling

- scaling a coordinate means multiplying each of its components by a scalar
- uniform scaling means this scalar is the same for all components:



Scaling

- non-uniform scaling: different scalars per component:



- how can we represent this in matrix form?

Scaling

- scaling operation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

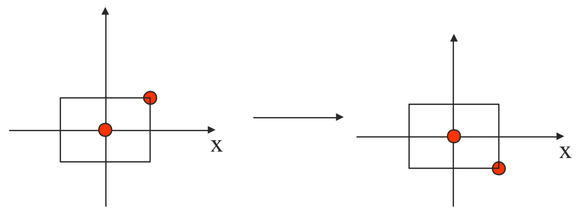
- or, in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

Reflection

- reflect across x axis
  - mirror

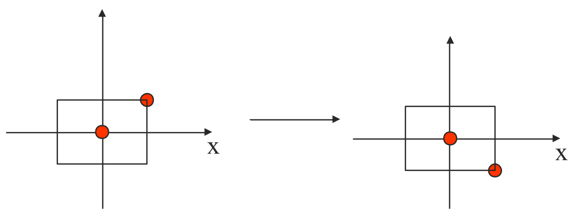
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} ? \\ ? \end{bmatrix}$$



Reflection

- reflect across x axis
  - mirror

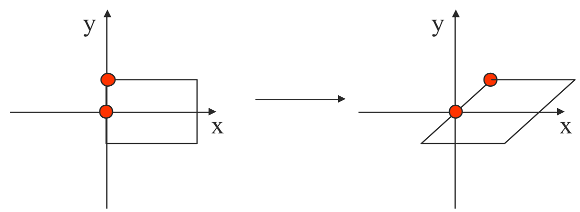
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$



Shear

- shear along x-axis
  - push points to right in proportion to height

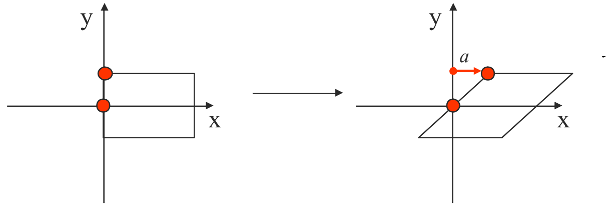
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



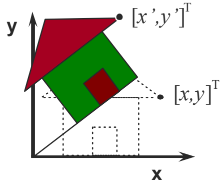
Shear

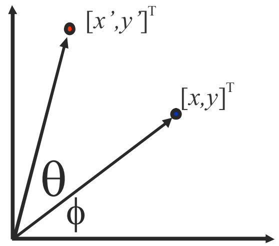
- shear along x-axis
  - push points to right in proportion to height

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



2D Rotation





Goal: Represent $[x',y']^T$ in terms of $[x,y]^T$ & θ

Represent *(x,y)* in polar coordinates

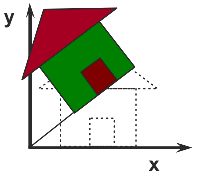
$x = r \cos(\phi)$

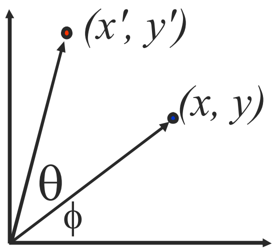
$y = r \sin(\phi)$

$x' = r \cos(\phi + \theta)$

$y' = r \sin(\phi + \theta)$

2D Rotation





Goal: Represent (x',y') in terms of (x,y) & θ

Represent *(x,y)* in polar coordinates

$x = r \cos(\phi)$

$y = r \sin(\phi)$

$x' = r \cos(\phi + \theta)$

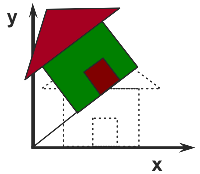
$y' = r \sin(\phi + \theta)$

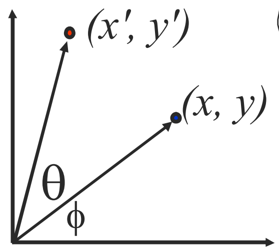
Trig Identity…

$x' = r \cos(\phi)\cos(\theta) - r \sin(\phi)\sin(\theta)$

$y' = r \sin(\phi)\cos(\theta) + r \cos(\phi)\sin(\theta)$

2D Rotation





Goal: Represent (x',y') in terms of (x,y) & θ

Represent *(x,y)* in polar coordinates

$y = r \sin(\phi)$

$x' = r \cos(\phi + \theta)$

$y' = r \sin(\phi + \theta)$

Trig Identity…

$x' = r \cos(\phi)\cos(\theta) - r \sin(\phi)\sin(\theta)$

$y' = r \sin(\phi)\cos(\theta) + r \cos(\phi)\sin(\theta)$

Substitute $r \cos(\phi)$ with x, $r \sin(\phi)$ with y

$$\boxed{\begin{aligned} x' &= x\,\mathbf{cos}(\theta) - y\,\mathbf{sin}(\theta) \\ y' &= x\,\mathbf{sin}(\theta) + y\,\mathbf{cos}(\theta) \end{aligned}}$$

2D Rotation Matrix

- $\boxed{\begin{aligned} x' &= x\cos(\theta) - y\sin(\theta) \\ y' &= x\sin(\theta) + y\cos(\theta) \end{aligned}}$ easy to capture in matrix form :

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \qquad\qquad \begin{bmatrix} x \\ y \end{bmatrix}$$

- even though $\sin(\theta)$ and $\cos(\theta)$ are nonlinear functions of θ ,
  - x' is a linear combination of x and y
  - y' is a linear combination of x and y

2D Rotations

- Linear $R(\theta_1+\theta_2)=R(\theta_1)+R(\theta_2)$
- 2D rotations are Commutative
- 3D Rotations are NOT Commutative (next lecture)

Linear Transforms in Matrix Form

$$x' = a\,x + b\,y$$
$$y' = c\,x + d\,y$$

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

$$\mathbf{x}' = \mathbf{M}\,\mathbf{x}$$

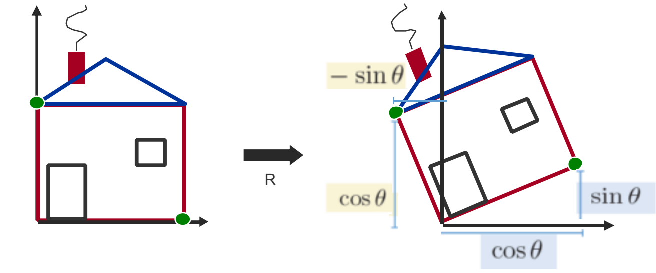
2D Coordinate Systems

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

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

- Can interpret the columns of the matrix as the x and y axes of the coordinate system

Rotation Matrix as a Coordinate System



$\mathbf{R}_\theta =$

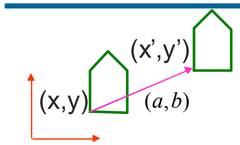
Linear Transformations

- linear transformations are combinations of elementary transformations:
  - scale
  - shear
  - rotation

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

- properties of linear transformations
  - satisifes T(s**x**+t**y**) = s T(**x**) + t T(**y**)
  - origin maps to origin
  - lines map to lines
  - parallel lines remain parallel
  - ratios are preserved
  - closed under composition

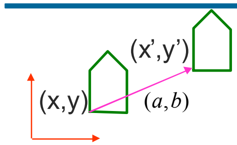
2D Translation



vector addition

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

2D Translation

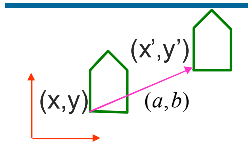


vector addition

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

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix}} \begin{bmatrix} x \\ y \end{bmatrix} \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}}_{\textit{rotation matrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D Translation



vector addition

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

matrix multiplication

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

matrix multiplication

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}}_{\textit{rotation matrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

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

Translation as a multiplication matrix??

Challenge

- matrix multiplication
  - for everything except translation
  - how to do everything with multiplication?
    - If every transform is a matrix multiplication then one can create composite transformations which are just a series of matrix multiplications, no special cases
- solution: **homogeneous coordinates**
  - represent 2D coordinates (x,y) with 3-tuple (x,y,1)

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

Translation as a multiplication matrix!

Lecture Outline

- 2D Linear Transforms
- ***Homogeneous Coordinates***
- 2D Affine Transforms
- Composite Transforms

Homogeneous Coordinates

- may seem unintuitive, but they make graphics operations much easier
- allow ***all*** transformations to be expressed through matrix multiplication
  - Specifically, we can express ***translations*** as a matrix multiplication
  - we'll see even more later...
- use 3x3 matrices for 2D transformations
  - use 4x4 matrices for 3D transformations

Points vs Vectors

What is the difference?

- Points have location, but no size or direction
- Vectors have size and direction, but no location

Problem: Yet, we represent both as triplets!

Convention

Vectors and Points are represented as column matrices

$$P = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix} \qquad \mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{bmatrix}$$

Switching Representations

Normal to homegeneous:

- Vector: append as fourth coordinate 0
- Point: append as fourth coordinate 1

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \rightarrow \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{bmatrix}$$

$$P = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} \rightarrow \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix}$$

Switching Representations

Homegeneous to normal:

- Vector: remove fourth coordinate (0)

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{bmatrix} \rightarrow \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}$$

- Point: remove fourth coordinate (1)

$$P = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

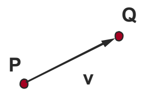
Relationship Between Points and Vectors

A difference between two points is a vector:

Q – P = v

We can consider a point as a base point plus a vector offset:

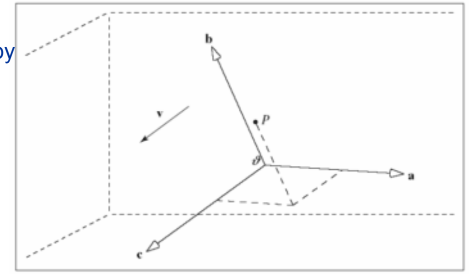
Q = P + v



Coordinate Frames

Coordinate Frame defined by the matrix:

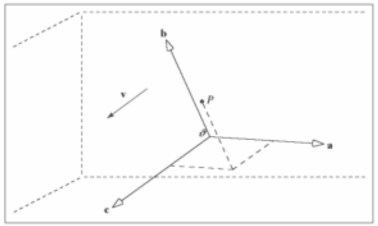
[**a b c** O]



$$\mathbf{v} = v_1\mathbf{a} + v_2\mathbf{b} + v_3\mathbf{c}$$

$$P = O + p_1\mathbf{a} + p_2\mathbf{b} + p_3\mathbf{c}$$

Homogeneous Representation: Points & Vectors



Coord Frame Rep | **Homogeneous Rep**

$$\mathbf{v} = v_1\mathbf{a} + v_2\mathbf{b} + v_3\mathbf{c} \qquad \mathbf{v} = [\mathbf{a}\ \ \mathbf{b}\ \ \mathbf{c}\ \ O]\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ 0 \end{bmatrix}$$

$$P = O + p_1\mathbf{a} + p_2\mathbf{b} + p_3\mathbf{c} \qquad P = [\mathbf{a}\ \ \mathbf{b}\ \ \mathbf{c}\ \ O]\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{bmatrix}$$

Does the Homogeneous Representation Support Operations?

- Valid operation if the last coordinate is 0 or a 1
  - **vector + vector = vector**
  - **point – point = vector**
  - **point + vector = point**
  - **point + point = ??**

Linear Combination of Points

Points P, Q scalars f, g*:*

$$fP+gQ = f\,[p_1,p_2,p_3,1]^T + g[q_1,q_2,q_3,1]^T$$
$$= [fp_1+gq_1,\ fp_2+gq_2,\ fp_3+gq_3,\ f+g]^T$$

What is this?

- If $(f+g) = 0$ then vector!
- If $(f+g) = 1$ then point!

Affine Combinations of Points

Definition:

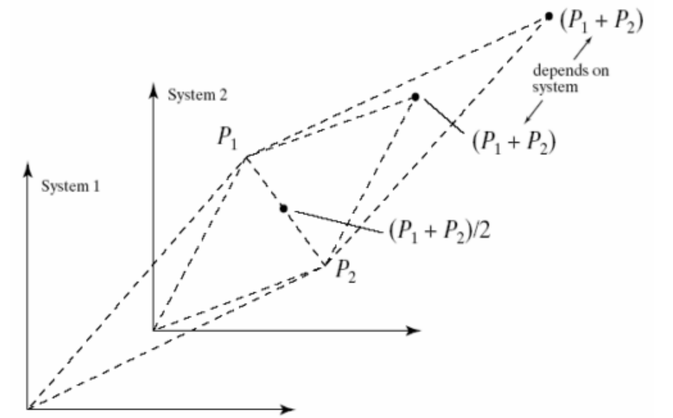
Points P_i: i = 1,…,n

Scalars f_i: i = 1,…,n

$f_1P_1+ \ldots + f_nP_n$ iff $f_1+ \ldots +f_n = 1$

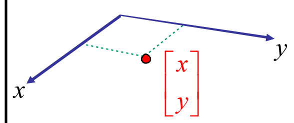
Example: $0.5P_1 + 0.5P_2$

Geometric Explanation



Homogeneous Coordinates Geometrically

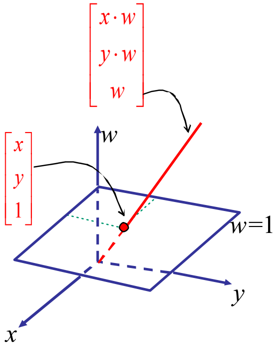
- point in 2D cartesian



Homogeneous Coordinates Geometrically

homogeneous **cartesian**

$$(x, y, w) \xrightarrow{/\,w} \left(\frac{x}{w}, \frac{y}{w}\right)$$

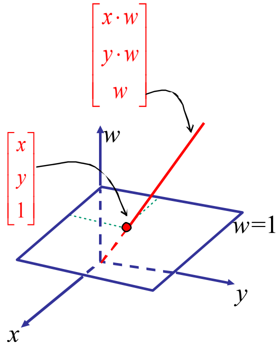


- point in 2D cartesian + weight w = point P in 3D homog. coords
- multiples of (x,y,w)
  - form a line L in 3D
  - all homogeneous points on L represent same 2D cartesian point
  - example: (2,2,1) = (4,4,2) = (1,1,0.5)

Homogeneous Coordinates Geometrically

homogeneous **cartesian**

$$(x, y, w) \xrightarrow{/\,w} \left(\frac{x}{w}, \frac{y}{w}\right)$$



- homogenize to convert homog. 3D point to cartesian 2D point:
  - divide by w to get (x/w, y/w, 1)
  - projects line to point onto w=1 plane
  - like normalizing, one dimension up
- when w=0, consider it as direction
  - points at infinity
  - these points cannot be homogenized
  - lies on x-y plane
- (0,0,0) is undefined

Lecture Outline

- 2D Linear Transforms
- Homogeneous Coordinates
- ***2D Affine Transforms***
- Composite Transforms

Affine Transformations

- Affine transformation = linear transformation + translation

$$\begin{pmatrix}x'\\y'\end{pmatrix} = \overbrace{\begin{pmatrix}a & b\\c & d\end{pmatrix}}^{\text{Rotation, Scale, Shear}} \cdot \begin{pmatrix}x\\y\end{pmatrix} + \overbrace{\begin{pmatrix}t_x\\t_y\end{pmatrix}}^{\text{Translation}}$$

- Affine transformation using homogenous coordinates is only a matrix multiplication:

$$\begin{pmatrix}x'\\y'\\1\end{pmatrix} = \begin{pmatrix}\overbrace{\boxed{\begin{matrix}a & b\\c & d\end{matrix}}}^{\text{Rotation, Scale Shear}} & \overbrace{\boxed{\begin{matrix}t_x\\t_y\end{matrix}}}^{\text{Translation}}\\ \begin{matrix}0 & 0\end{matrix} & 1\end{pmatrix} \cdot \begin{pmatrix}x\\y\\1\end{pmatrix}$$

General Form of 2D Affine Transformation

Points:

$$\begin{bmatrix} Q_x \\ Q_y \\ 1 \end{bmatrix} = \begin{bmatrix} \overset{\text{Rotation, Scale, Shear}}{\boxed{\begin{matrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{matrix}}} & \overset{\text{Translation}}{\boxed{\begin{matrix} m_{13} \\ m_{23} \end{matrix}}} \\ \begin{matrix} 0 & \quad 0 \end{matrix} & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ 1 \end{bmatrix}$$

Vectors:

$$\begin{bmatrix} W_x \\ W_y \\ 0 \end{bmatrix} = \begin{bmatrix} \overset{\text{Rotation, Scale, Shear}}{\boxed{\begin{matrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{matrix}}} & \overset{\text{Translation}}{\boxed{\begin{matrix} m_{13} \\ m_{23} \end{matrix}}} \\ \begin{matrix} 0 & \quad 0 \end{matrix} & 1 \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ 0 \end{bmatrix}$$

Linear Transformation ➔ Affine Transformation

- Rewrite linear transforms written for a cartesian coordinate system to transforms for the homogeneous coordinate system:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & 0 \\ d & e & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\mathbf{M}_{scale} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{shear\_x} = \begin{bmatrix} 1 & shx & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{M}_{rotation} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

All Elementary Affine Transformations

- Scale, shear, rotation,translation

x' = Mx

$$\mathbf{M}_{scale} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{shear\_x} = \begin{bmatrix} 1 & shx & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{M}_{rotation} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{translation} = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix}$$

- Translation uses rightmost column

General Form of 2D Affine Transformations

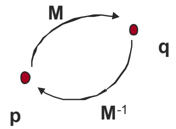
Transformation as a matrix multiplication

$$\begin{bmatrix} Q_x \\ Q_y \\ 1 \end{bmatrix} = \begin{bmatrix} \overset{\text{Rotation, Scale, Shear}}{\boxed{\begin{matrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{matrix}}} & \overset{\text{Translation}}{\boxed{\begin{matrix} m_{13} \\ m_{23} \end{matrix}}} \\ \begin{matrix} 0 & \quad 0 \end{matrix} & 1 \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ 1 \end{bmatrix}$$

$$Q = \mathrm{M}P$$

Inverse of a Transformation

Inverse transformation: $\mathbf{q} = \mathbf{M}\mathbf{p}$, $\mathbf{p} = \mathbf{M}^{-1}\mathbf{q}$



We can use Cramer's rule to invert **M**, *or we can be smarter about it*

Inverse of Translation

$$\mathbf{q} = \mathbf{M}(\mathbf{t})\mathbf{p}$$

$$\mathbf{p} = \mathbf{M}^{-1}(\mathbf{t})\mathbf{q} = \mathbf{M}(-\mathbf{t})\mathbf{p}$$

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Inverse of Scaling

$$\mathbf{q} = \mathbf{M}(s_x, s_y)\mathbf{p}$$

$$\mathbf{p} = \mathbf{M}^{-1}(s_x, s_y)\mathbf{q} = \mathbf{M}(\tfrac{1}{s_x}, \tfrac{1}{s_y})\mathbf{q}$$

$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{s_x} & 0 & 0 \\ 0 & \frac{1}{s_y} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Inverse of a Shear in x

$$\mathbf{q} = \mathbf{M}_{\text{shear in }\mathbf{x}}(a)\mathbf{p}$$

$$\mathbf{p} = \mathbf{M}^{-1}_{\text{shear in }\mathbf{x}}(a)\mathbf{q} = \mathbf{M}_{\text{shear in }\mathbf{x}}(-a)\mathbf{q}$$

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

Inverse of Rotation

$$\mathbf{q} = \mathbf{M}(\theta)\mathbf{p}$$

$$\mathbf{p} = \mathbf{M}^{-1}(\theta)\mathbf{q} = \mathbf{M}(-\theta)\mathbf{p}$$

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Affine Transformations

- ***Any*** affine transformation is a combination of
  - translation and/or
  - linear transformations:
    - scale, shear, rotation

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- properties of affine transformations
  - origin does not necessarily map to origin
  - lines map to lines
  - parallel lines remain parallel
  - relative ratios of points on a line are preserved
  - closed under composition - combination of affine transformations is an affine transform

Lecture Outline

- 2D Linear Transforms
- Homogeneous Coordinates
- 2D Affine Transforms
- ***Composite Transforms***

Composing 2D Affine Transformations

Composing two affine transformations produces an affine transformation

$$\mathbf{q} = T_2(T_1\mathbf{p}))$$

In matrix form:

$$\mathbf{q} = \mathbf{M}_2(\mathbf{M}_1\mathbf{p}) = (\mathbf{M}_2\mathbf{M}_1)\mathbf{p} = \mathbf{M}\mathbf{p}$$

Which transformation happens first?

Inverting Composite Transforms

- Say I want to invert a combination of 3 transforms

$$\mathbf{x}' = (\mathbf{M}_3\mathbf{M}_2\mathbf{M}_1)\mathbf{x} = \mathbf{M}\mathbf{x}$$

- Option 1: Find composite matrix, invert $\mathbf{x} = \mathbf{M}^{-1}\mathbf{x}'$
- Option 2: Invert each transform ***and swap order***

$$\mathbf{x}' = \mathbf{M}_3\mathbf{M}_2\mathbf{M}_1\mathbf{x}$$

$$\mathbf{M}_3^{-1}\mathbf{x}' = (\mathbf{M}_3^{-1}\mathbf{M}_3)\mathbf{M}_2\mathbf{M}_1\mathbf{x}$$

$$\mathbf{M}_2^{-1}\mathbf{M}_3^{-1}\mathbf{x}' = (\mathbf{M}_2^{-1}\mathbf{M}_2)\mathbf{M}_1\mathbf{x}$$

$$\mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\mathbf{M}_3^{-1}\mathbf{x}' = (\mathbf{M}_1^{-1}\mathbf{M}_1)\mathbf{x}$$

$$\mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\mathbf{M}_3^{-1}\mathbf{x}' = \mathbf{x}$$

$$\mathbf{M}^{-1} = (\mathbf{M}_3\mathbf{M}_2\mathbf{M}_1)^{-1} = \mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\mathbf{M}_3^{-1}$$

transformation_game.jar

Main Points

Any affine transformation can be performed as series of elementary transformations

Affine transformations are the main modeling tool in graphics

Make sure you understand the order

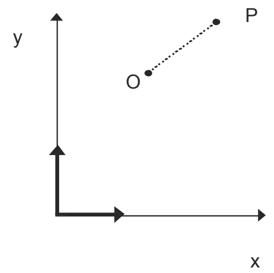
Composite 2D Transformations

The following are composite 2D transformations:

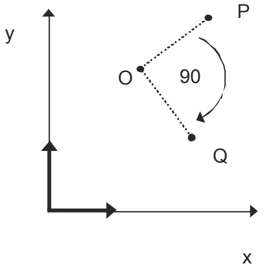
- Rotation about an arbitrary pivot point
- Scaling around an arbitrary point
- Reflection
- Reflection about a tilted line

Example of 2D Composite Transformation

Rotate -90 deg around an arbitrary point O:

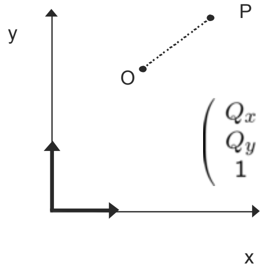


Rotate Around an Arbitrary Point



Rotate Around an Arbitrary Point

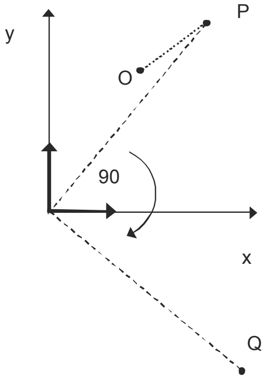
We know how to rotate around the origin



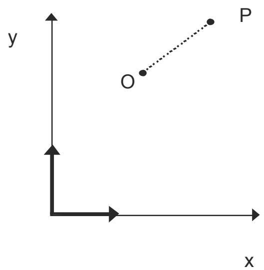
$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$

Rotate Around an Arbitrary Point

…but that is not what we want to do!

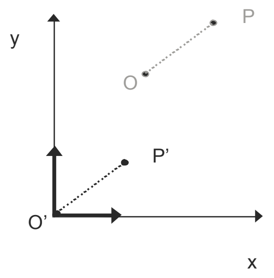


So What Do We Do?



Transform it to the Known Case

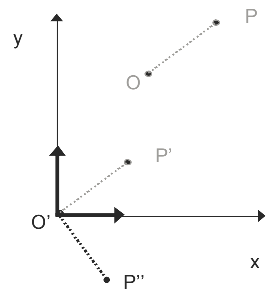
Translate***(-O_x,-O_y)***



Second Step: Rotation

Translate(-$\boldsymbol{O_x}$,-$\boldsymbol{O_y}$)

Rotate(-90)



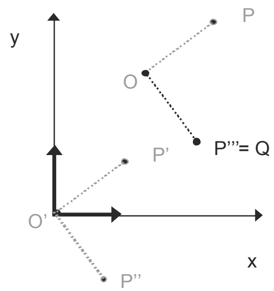
Finally, Put Everything Back

Push the following <u>matrices on the stack</u>:

Translate(-O_x,-O_y)

Rotate(-90)

Translate(O_x,O_y)



Rotation About Arbitrary Point

<u>Composite transformation representation:</u>

$\mathbf{M} = \mathbf{T}(O_x,O_y)\ \mathbf{R}(-90)\ \mathbf{T}(-O_x,-O_y)$

Order is IMPORTANT!