CMSC427 Transformations I

Credit: slides 9+ from Prof. Zwicker

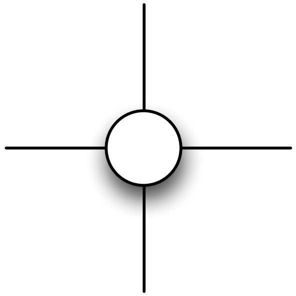
Transformations: outline

- Types of transformations
  - Specific: translation, rotation, scaling, shearing
  - Classes: rigid, affine, projective
- Representing transformations
  - Unifying representation with homogeneous coordinates
  - Transformations represented as matrices
- Composing transformations
  - Sequencing matrices
  - Sequencing using OpenGL stack model
- Transformation examples
  - Rotating or scaling about a point
  - Rotating to a new coordinate frame
- Applications
  - Modeling transformations (NOW)
  - Viewing transformations (LATER)

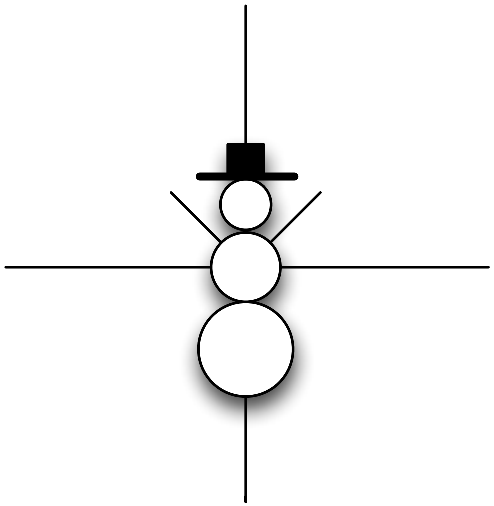
Modeling with transformations

- Create instance of object in object coordinate space
  - Create circle at origin
- Transform object to world coordinate space
  - Scale by 1.5
  - Move down by 2 unit
- Do so for other objects
  - Two rects make hat
  - Three circles make body
  - Two lines make arms

- Object coordinate space

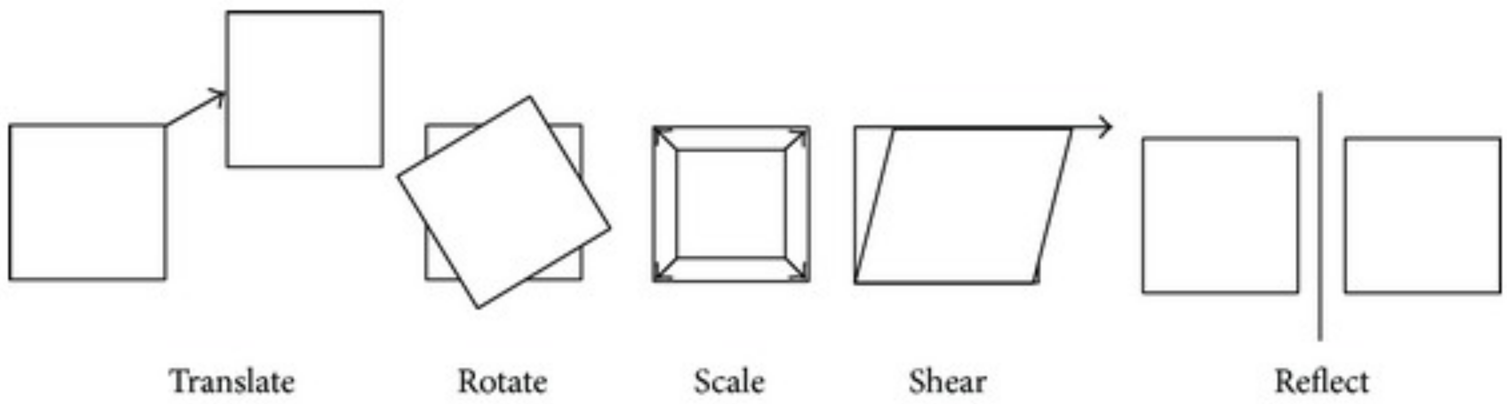


- World coordinate space



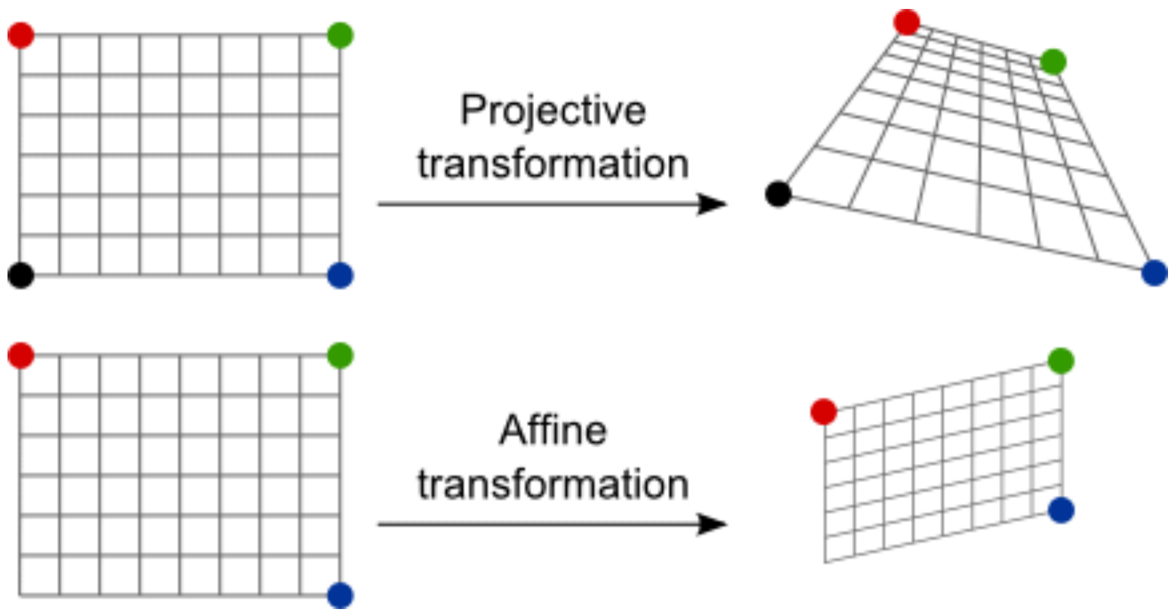
Classes of transformations

- Rigid
  - Translate, rotate, uniform scale
  - No distortion to object
- Affine
  - Translate, rotate, scale (non-uniform), shear, reflect
  - Limited distortions
  - Preserve parallel lines



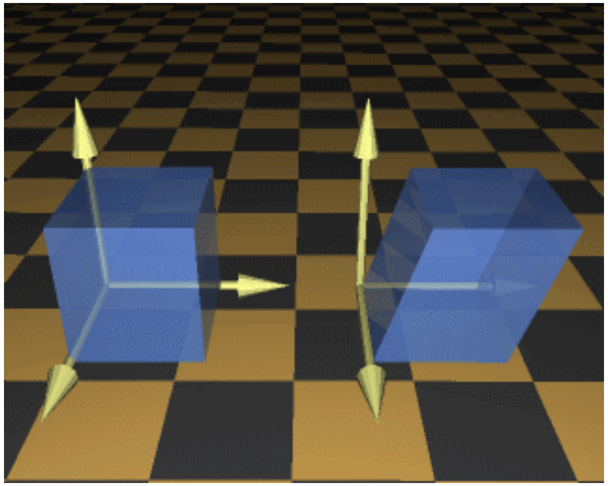
Classes of transformations

- Affine
  - Preserves parallel lines
- Projective
  - Foreshortens
  - Lines converge
  - For viewing/rendering



Classes of transformations: summary

- Affine
  - Reshape, size object
- Rigid
  - Place, move object
- Projective
  - View object
  - *Later …*
- *Non-linear, arbitrary*
  - *Twists, pinches, pulls*
  - *Not in this unit*



First try: scale and rotate vertices in vector notation

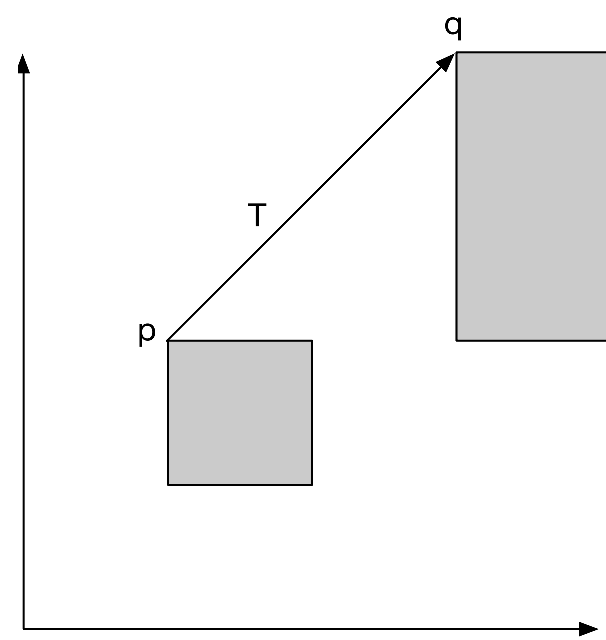
- Scale a point p by s and translate by T
- Vector multiplication and addition
- Repeat and we get

$$q = s_2(s * p + T) + T_2$$

- Gets unwieldy
- Instead – unify notation with homogeneous coordinates and matrices

$$q = s * p + T$$

$$q = 2 * (2,3) + < 2,2 >$$
$$q = (6,8)$$



Matrix practice

$$M = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \qquad MR = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix} =$$

$$R = \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix} \qquad RM = \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} =$$

$$P = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \qquad MP = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} =$$

Matrix practice

$$M = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} \qquad MR = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix} = \begin{bmatrix} 2 & 7 \\ 1 & 6 \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix} \qquad RM = \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 3 & 6 \end{bmatrix}$$

$$P = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \qquad MP = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 8 \end{bmatrix}$$

Matrix transpose and column vectors

$$R^T = \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix}^T =$$

$$H^T = \begin{bmatrix} 2 & 1 & 3 \\ 4 & 1 & 5 \end{bmatrix}^T =$$

$$P = \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 & 3 \end{bmatrix}^T$$

Matrix transpose and column vectors

$$R^T = \begin{bmatrix} 1 & 1 \\ 0 & 3 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 \\ 1 & 3 \end{bmatrix}$$

$$H^T = \begin{bmatrix} 2 & 1 & 3 \\ 4 & 1 & 5 \end{bmatrix}^T = \begin{bmatrix} 2 & 4 \\ 1 & 1 \\ 3 & 5 \end{bmatrix}$$

$$P = \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 & 3 \end{bmatrix}^T$$

Matrices

Abstract point of view

- Mathematical objects with set of operations
  - Addition, subtraction, multiplication, multiplicative inverse, etc.
- Similar to integers, real numbers, etc.

But

- Properties of operations are different
  - E.g., multiplication is not commutative
- Represent different intuitive concepts
  - Scalar numbers represent distances
  - Matrices can represent coordinate systems, rigid motions, in 3D and higher dimensions, etc.

Practical point of view

- Rectangular array of numbers

$$\mathbf{M} = \begin{bmatrix} m_{1,1} & m_{1,2} & \dots & m_{1,n} \\ m_{2,1} & m_{2,2} & \dots & m_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{m,1} & m_{2,2} & \dots & m_{m,n} \end{bmatrix} \in \mathbf{R}^{m \times n}$$

- Square matrix if $\mathbf{m} = \mathbf{n}$
- In graphics often $\mathbf{m} = \mathbf{n} = 3, \mathbf{m} = \mathbf{n} = 4$

Matrix addition

$$\mathbf{A}+\mathbf{B}=\begin{bmatrix} a_{1,1}+b_{1,1} & a_{1,2}+b_{1,2} & \dots & a_{1,n}+b_{1,n} \\ a_{2,1}+b_{2,1} & a_{2,2}+b_{2,2} & \dots & a_{2,n}+b_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1}+b_{m,1} & a_{2,2}+b_{2,2} & \dots & a_{m,n}+b_{m,n} \end{bmatrix}$$

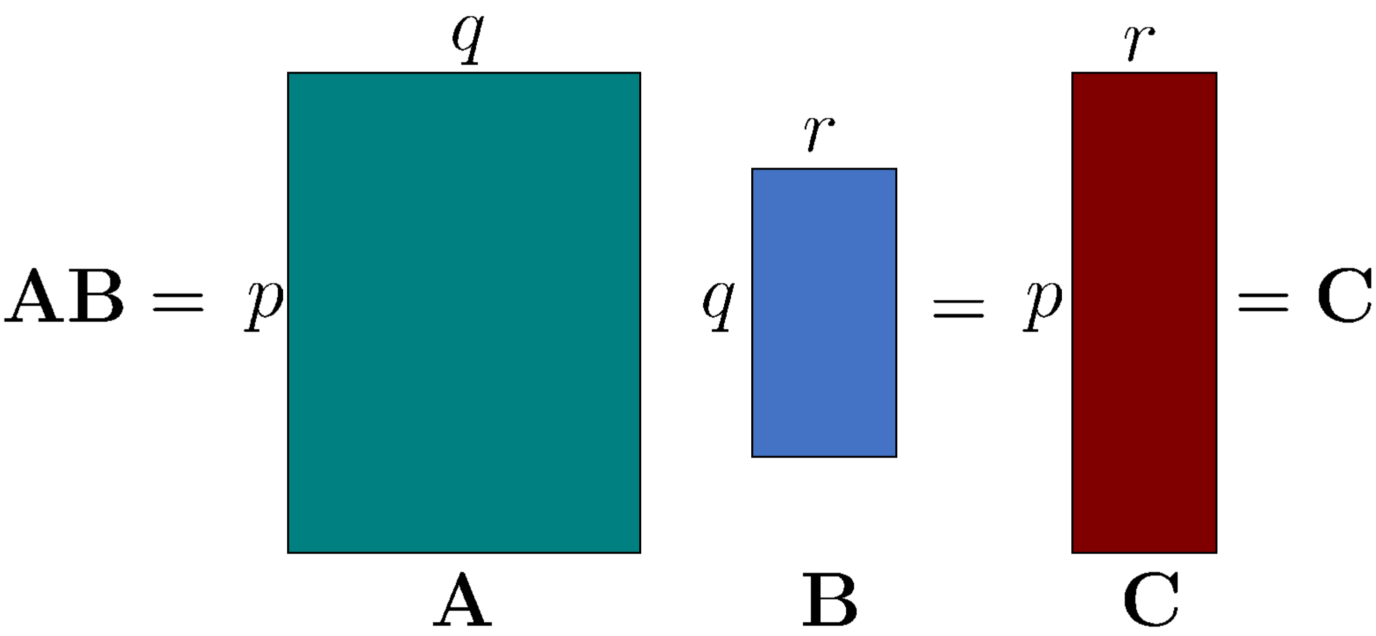
$$\mathbf{A},\mathbf{B}\in\mathbf{R}^{m\times n}$$

Multiplication with scalar

$$s\mathbf{M} = \mathbf{M}s = \begin{bmatrix} sm_{1,1} & sm_{1,2} & \dots & sm_{1,n} \\ sm_{2,1} & sm_{2,2} & \dots & sm_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ sm_{m,1} & sm_{2,2} & \dots & sm_{m,n} \end{bmatrix}$$

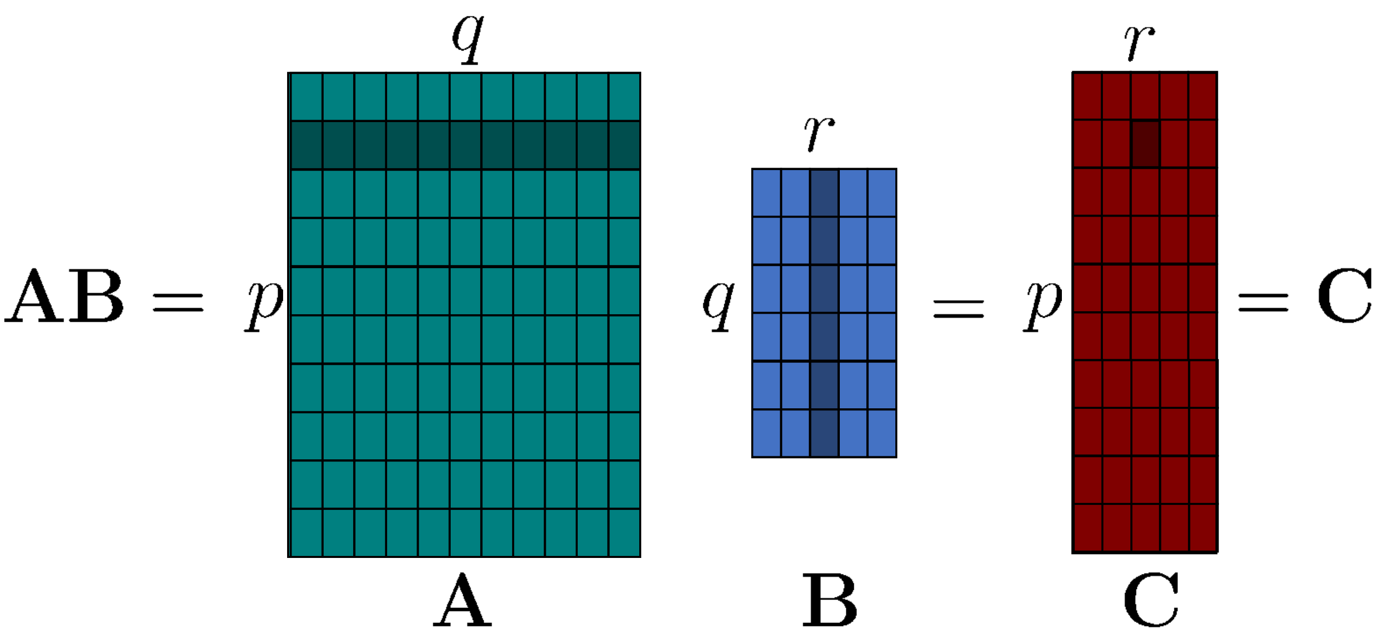
Matrix multiplication

$$\mathbf{AB} = \mathbf{C}, \quad \mathbf{A} \in \mathbf{R}^{p,q}, \mathbf{B} \in \mathbf{R}^{q,r}, \mathbf{C} \in \mathbf{R}^{p,r}$$



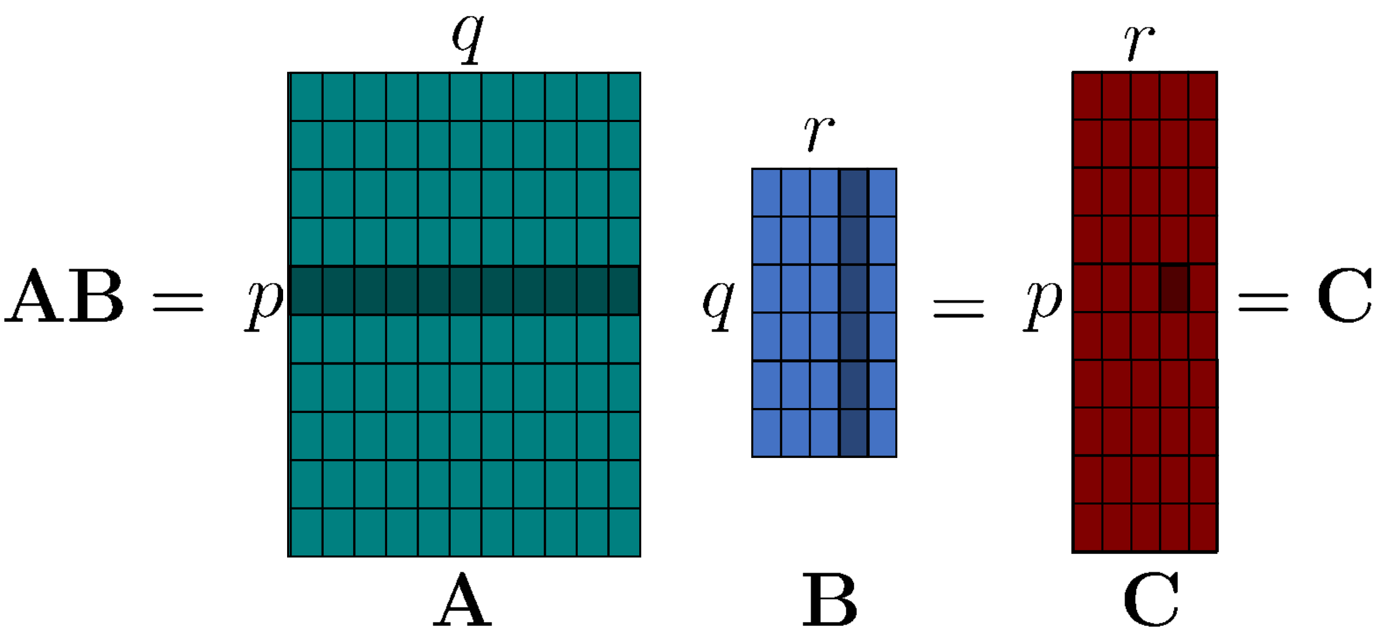
Matrix multiplication

$$\mathbf{AB} = \mathbf{C}, \quad \mathbf{A} \in \mathbf{R}^{p,q}, \mathbf{B} \in \mathbf{R}^{q,r}, \mathbf{C} \in \mathbf{R}^{p,r}$$



Matrix multiplication

$$\mathbf{AB} = \mathbf{C}, \quad \mathbf{A} \in \mathbf{R}^{p,q}, \mathbf{B} \in \mathbf{R}^{q,r}, \mathbf{C} \in \mathbf{R}^{p,r}$$

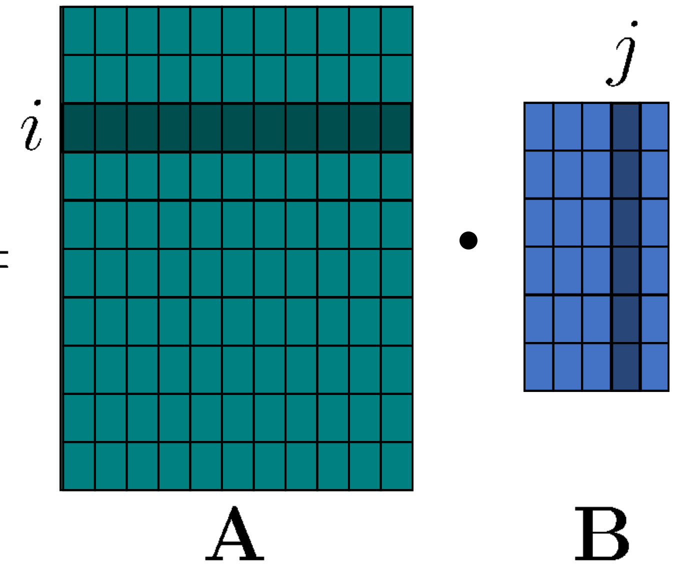


Matrix multiplication

$$\mathbf{AB} = \mathbf{C}, \quad \mathbf{A} \in \mathbf{R}^{p,q}, \mathbf{B} \in \mathbf{R}^{q,r}, \mathbf{C} \in \mathbf{R}^{p,r}$$

$$(\mathbf{AB})_{i,j} = \mathbf{C}_{i,j} = \sum_{k=1}^{q} a_{i,k} b_{k,j}, \quad i \in 1..p, j \in 1..r$$

$$(\mathbf{AB})_{i,j} = \mathbf{C}_{i,j} =$$



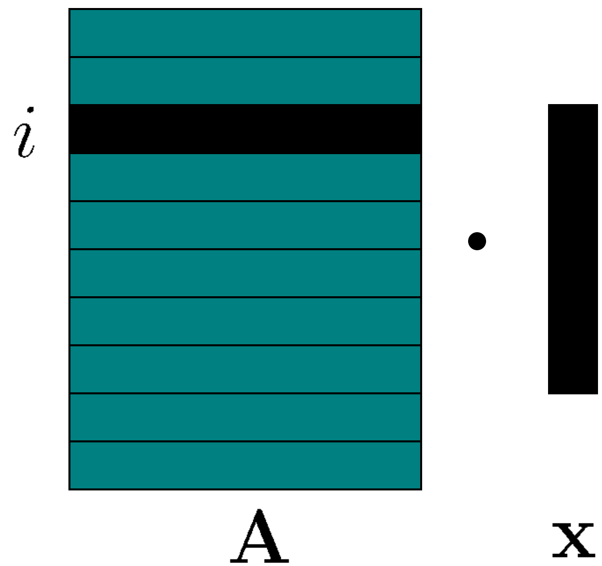
Matrix multiplication

Special case: matrix-vector multiplication

$$\mathbf{A}\mathbf{x} = \mathbf{y}, \quad \mathbf{A} \in \mathbf{R}^{p,q}, \mathbf{x} \in \mathbf{R}^{q}, \mathbf{y} \in \mathbf{R}^{p}$$

$$(\mathbf{A}\mathbf{x})_i = \mathbf{y}_i = \sum_{k=1}^{q} a_{i,k} x_k$$

$$(\mathbf{A}\mathbf{x})_i = \mathbf{y}_i =$$



Linearity

- Distributive law holds

  i.e., matrix $\mathbf{A}(s\mathbf{B} + t\mathbf{C}) = s\mathbf{A}\mathbf{B} + t\mathbf{A}\mathbf{C}$.

  http://en.wikipedia.org/wiki/Linear_map

- But multiplication is not commutative,

  in general

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

Identity matrix

$$\mathbf{I} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix} \in \mathbf{R}^{n \times n}$$

$$\mathbf{MI} = \mathbf{IM} = \mathbf{M}, \quad \text{for any } \mathbf{M} \in \mathbf{R}^{n \times n}$$

Definition

If a square matrix $\mathbf{M}$ is non-singular, there exists a unique inverse $\mathbf{M}^{-1}$ such that

$$\mathbf{M}\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$$

- Note

$$(\mathbf{MPQ})^{-1} = \mathbf{Q}^{-1}\mathbf{P}^{-1}\mathbf{M}^{-1}$$

- Computation
  - Gaussian elimination, Cramer’s rule (OctaveOnline)
  - Review in your linear algebra book, or quick summary http://www.maths.surrey.ac.uk/explore/emmaspages/option1.html

Java vs. OpenGL matrices

- OpenGL (underlying 3D graphics API used in the Java code, more later) http://en.wikipedia.org/wiki/OpenGL
  - Matrix elements stored in array of floats float M[16];
  - "Column major" ordering
- Java base code
  - "Row major" indexing
  - Conversion from Java to OpenGL convention hidden somewhere in basecode!

$$\begin{bmatrix} m[0] & m[4] & m[8] & m[12] \\ m[1] & m[5] & m[9] & m[13] \\ m[2] & m[6] & m[10] & m[14] \\ m[3] & m[7] & m[11] & m[15] \end{bmatrix}$$

$$\begin{bmatrix} m(0,0) & m(0,1) & m(0,2) & m(0,3) \\ m(1,0) & m(1,1) & m(1,2) & m(1,3) \\ m(2,0) & m(2,1) & m(2,2) & m(2,3) \\ m(3,0) & m(3,1) & m(3,2) & m(3,3) \end{bmatrix}$$

Today

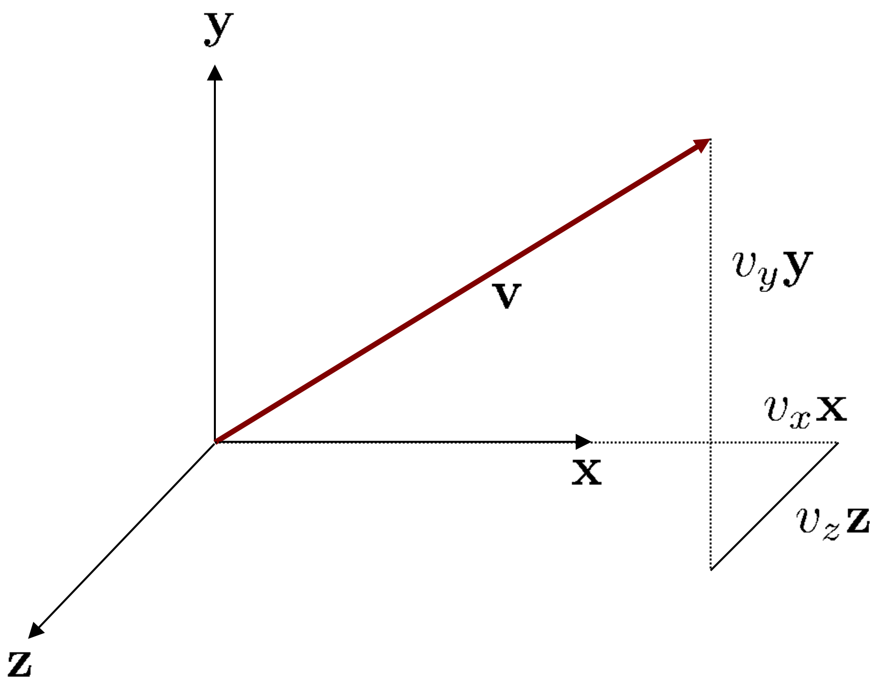
Transformations & matrices

- Introduction
- Matrices
- Homogeneous coordinates
- Affine transformations
- Concatenating transformations
- Change of coordinates
- Common coordinate systems

Vectors & coordinate systems

- Vectors defined by orientation, length
- Describe using three basis vectors

$$\mathbf{x}, \mathbf{y}, \mathbf{z}$$



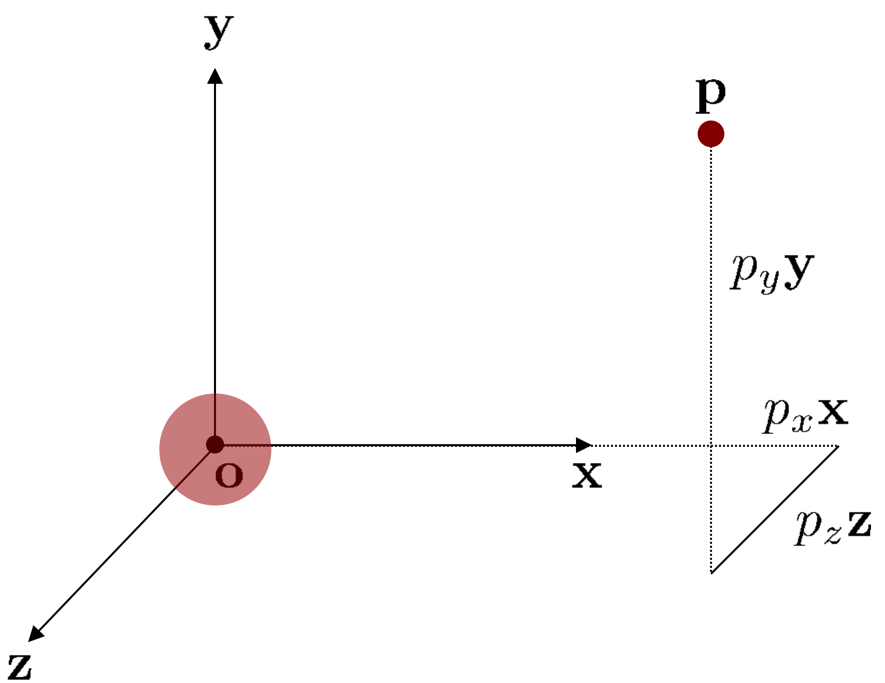
$$\mathbf{v} = v_x\mathbf{x} + v_y\mathbf{y} + v_z\mathbf{z}$$

Points in 3D

- How do we represent 3D points?
- Are three basis vectors enough to define the location of a point?

Points in 3D

- Describe using three basis vectors and reference point, origin



$$\mathbf{p} = p_x \mathbf{x} + p_y \mathbf{y} + p_z \mathbf{z} + \mathbf{o}$$

Vectors vs. points

- Vectors

$$\mathbf{v} = v_x\mathbf{x} + v_y\mathbf{y} + v_z\mathbf{z} + \boxed{0 \cdot \mathbf{o}} \qquad \begin{bmatrix} v_x \\ v_y \\ v_z \\ \boxed{0} \end{bmatrix}$$

- Points

$$\mathbf{p} = p_x\mathbf{x} + p_y\mathbf{y} + p_z\mathbf{z} + \boxed{1 \cdot \mathbf{o}} \qquad \begin{bmatrix} p_x \\ p_y \\ p_z \\ \boxed{1} \end{bmatrix}$$

- Representation of vectors and points using 4th coordinate is called homogeneous coordinates

Homogeneous coordinates

- Represent an affine space
  http://en.wikipedia.org/wiki/Affine_space
- Intuitive definition
  - Affine spaces consist of a vector space and a set of points
  - There is a subtraction operation that takes two points and returns a vector
  - Axiom I: for any point $\mathbf{a}$ and vector $\mathbf{v}$, there exists point $\mathbf{b}$, such that $(\mathbf{b}-\mathbf{a}) = \mathbf{v}$
  - Axiom II: for any points $\mathbf{a}, \mathbf{b}, \mathbf{c}$ we have $(\mathbf{b}-\mathbf{a})+(\mathbf{c}-\mathbf{b}) = \mathbf{c}-\mathbf{a}$

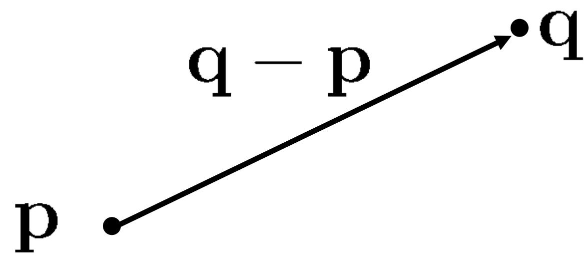
Affine space

Vector space,
http://en.wikipedia.org/wiki/Vector_space
- [xyz] coordinates
- represents vectors

Affine space
http://en.wikipedia.org/wiki/Affine_space
- [xyz1], [xyz0] homogeneous coordinates
- distinguishes points and vectors

Homogeneous coordinates

- Subtraction of two points yields a vector



- Using homogeneous coordinates

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \qquad \mathbf{q} = \begin{bmatrix} q_x \\ q_y \\ q_z \\ 1 \end{bmatrix} \qquad \mathbf{q} - \mathbf{p} = \begin{bmatrix} q_x - p_x \\ q_y - p_y \\ q_z - p_z \\ 0 \end{bmatrix}$$

Transformations & matrices

- Introduction
- Matrices
- Homogeneous coordinates
- Affine transformations
- Concatenating transformations
- Change of coordinates
- Common coordinate systems

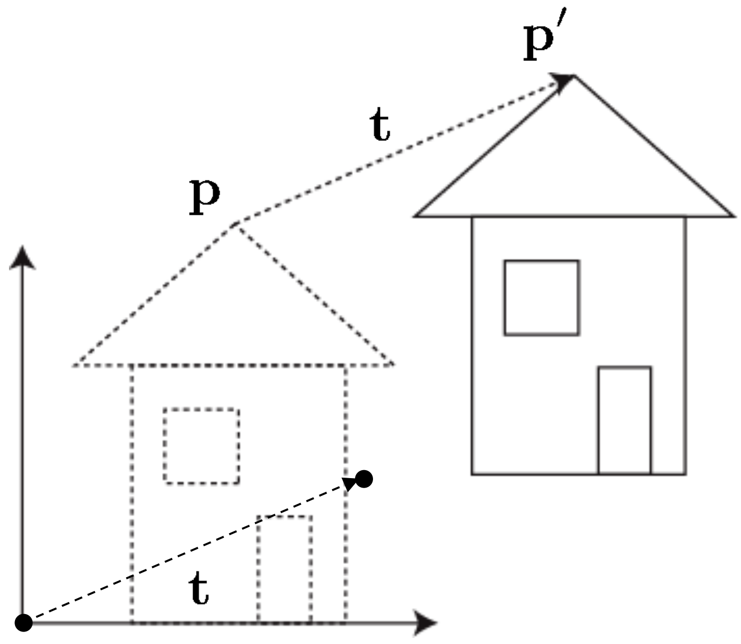
Affine transformations

- Transformation, or mapping: function that maps each 3D point to a new 3D point „f: $\mathbf{R}^3$ -> $\mathbf{R}^3$“
- Affine transformations: class of transformations to position 3D objects in space
- Affine transformations include
  - Rigid transformations
    - Rotation
    - Translation
  - Non-rigid transformations
    - Scaling
    - Shearing

Affine transformations

- Definition: mappings that preserve colinearity and ratios of distances
  http://en.wikipedia.org/wiki/Affine_transformation
  - Straight lines are preserved
  - Parallel lines are preserved
- Linear transformations + translation
- Nice: All desired transformations (translation, rotation) implemented using homogeneous coordinates and matrix-vector multiplication

Translation



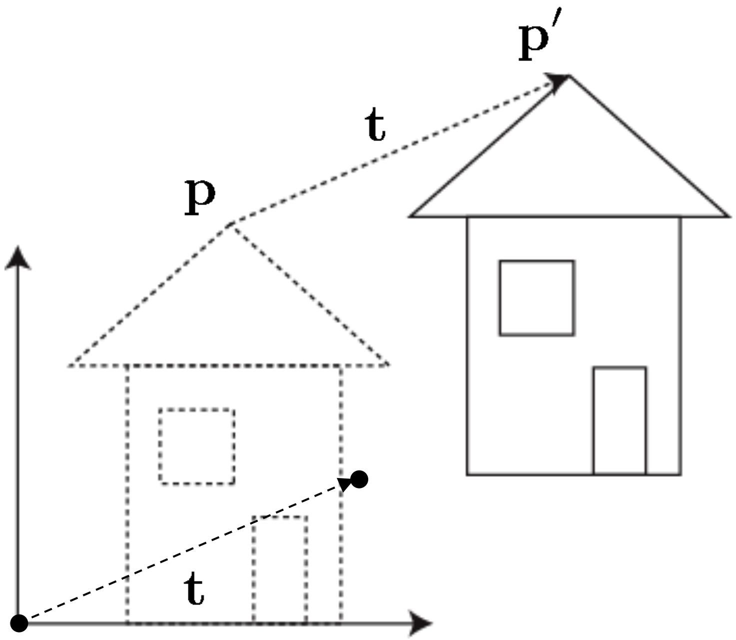
Point

Vector

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \\ 0 \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{p} + \mathbf{t} = \begin{bmatrix} p_x + t_x \\ p_y + t_y \\ p_z + t_z \\ 1 \end{bmatrix}$$

Matrix formulation



Point

Vector

$$\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \qquad \mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \\ 0 \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{p} + \mathbf{t} = \begin{bmatrix} p_x + t_x \\ p_y + t_y \\ p_z + t_z \\ 1 \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{bmatrix}}_{\mathbf{p}'} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}(\mathbf{t})} \underbrace{\begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}}_{\mathbf{p}}$$

$$\mathbf{p}' = \mathbf{T}(\mathbf{t})\mathbf{p}$$

- Inverse translation

$$\mathbf{T}(\mathbf{t})^{-1} = \mathbf{T}(-\mathbf{t})$$

$$\mathbf{T}(\mathbf{t}) = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}(-\mathbf{t}) = \begin{bmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Verify that

$$\mathbf{T}(-\mathbf{t})\mathbf{T}(\mathbf{t}) = \mathbf{T}(\mathbf{t})\mathbf{T}(-\mathbf{t}) = \mathbf{I}$$

Note

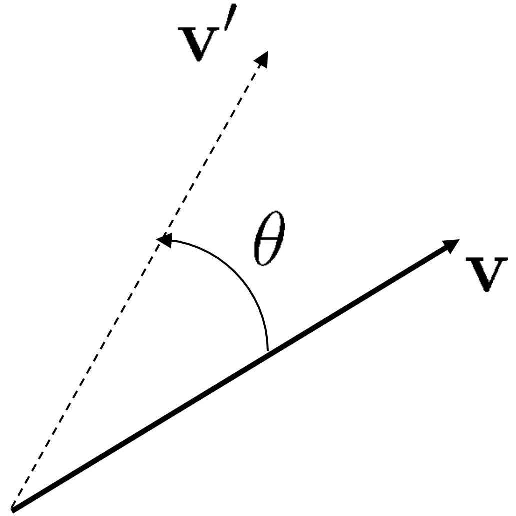
- What happens when you translate a vector?

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix} = ?$$

First: rotating a vector in 2D

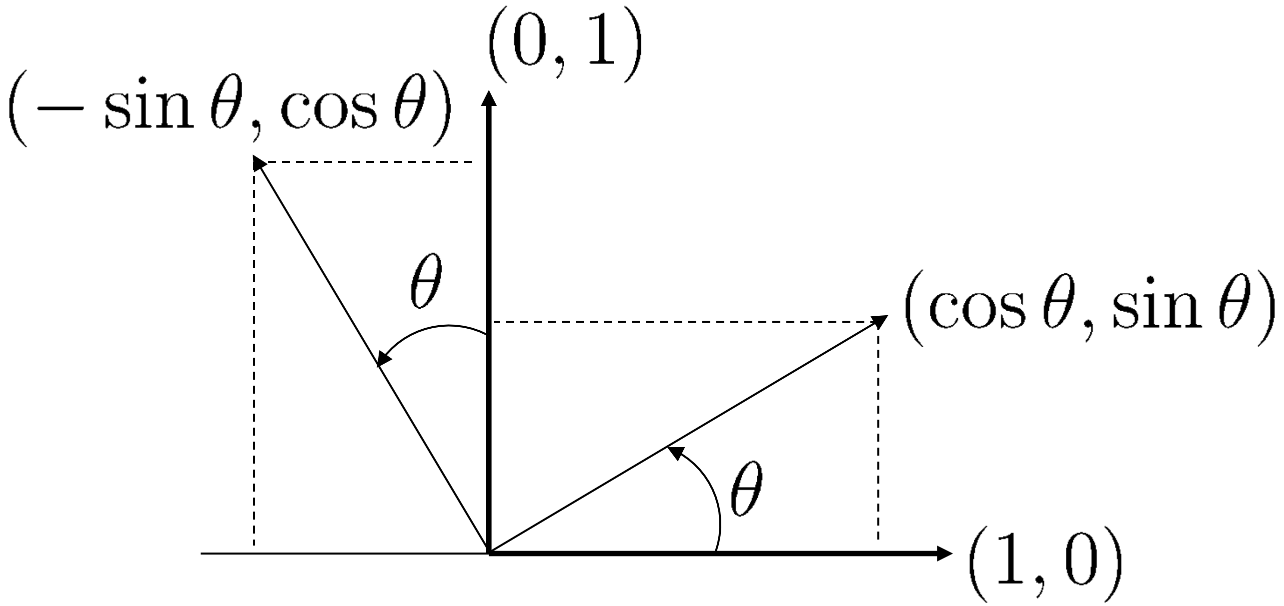
- Convention: positive angle rotates counterclockwise
- Express using rotation matrix

$$\mathbf{R}(\theta)$$

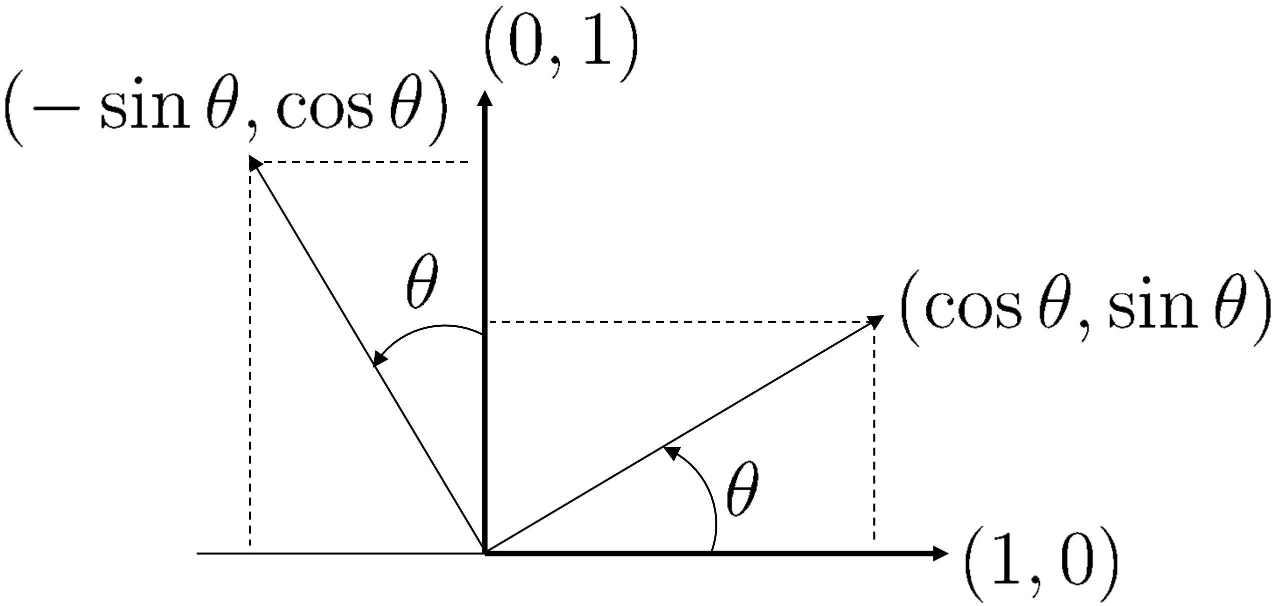


$$\mathbf{v}' = \mathbf{R}(\theta)\mathbf{v}$$

Rotating a vector in 2D



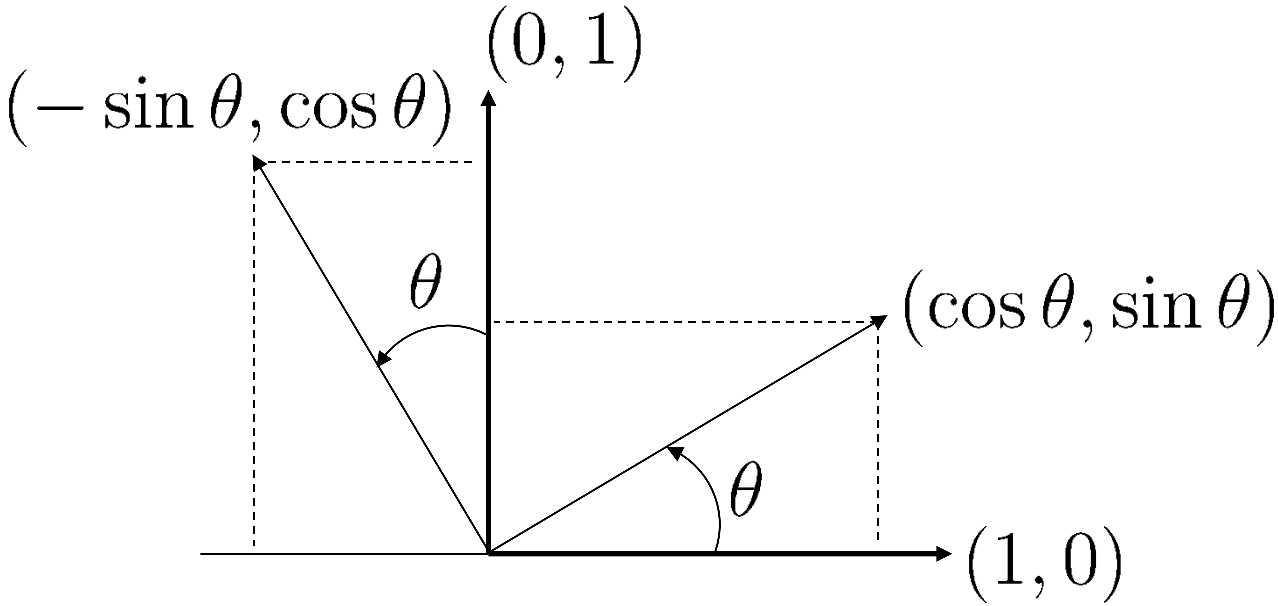
Rotating a vector in 2D



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

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

Rotating a vector in 2D



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

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

Rotation around z-axis

- z-coordinate does not change

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

$$v^0 = R_z(\mu)v$$

- What is the matrix for

$$\theta = 0, \theta = 90, \theta = 180$$

$$\mathbf{R}_z(\theta)\mathbf{v} = \begin{bmatrix} \cos(\theta)v_x - \sin(\theta)v_y \\ \sin(\theta)v_x + \cos(\theta)v_y \\ v_z \\ 1 \end{bmatrix}$$

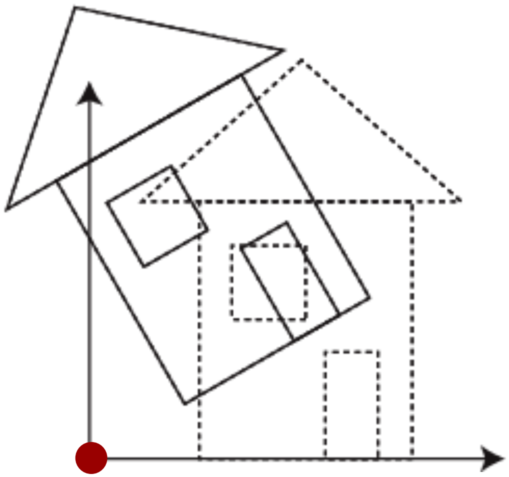
Other coordinate axes

- Same matrix to rotate points and vectors
- Points are rotated around origin

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

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

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

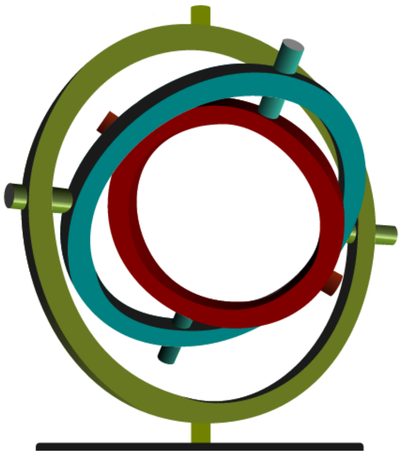


Rotation in 3D

- Concatenate rotations around *x,y,z* axes to obtain rotation around arbitrary axes through origin

$$\mathbf{R}_{x,y,z}(\theta_x, \theta_y, \theta_z) = \mathbf{R}_x(\theta_x)\mathbf{R}_y(\theta_y)\mathbf{R}_z(\theta_z)$$

- $\theta_x, \theta_y, \theta_z$ are called Euler angles
  http://en.wikipedia.org/wiki/Euler_angles
- Disadvantage: result depends on order!



Gimbal
https://en.wikipedia.org/wiki/Gimbal

$$\mathbf{R}_x(\theta_x)\mathbf{R}_y(\theta_y)\mathbf{R}_z(\theta_z) \neq \mathbf{R}_z(\theta_z)\mathbf{R}_y(\theta_y)\mathbf{R}_x(\theta_x)$$

Rotation around arbitrary axis

- Still: origin does not change
- Counterclockwise rotation
- Angle θ, unit axis $\mathbf{a}$
-

$$c_\theta = \cos\theta, s_\theta = \sin\theta$$

$$\mathbf{R}(\mathbf{a},\theta) = \begin{bmatrix} a_x^2 + c_\theta(1-a_x^2) & a_x a_y(1-c_\theta) - a_z s_\theta & a_x a_z(1-c_\theta) + a_y s_\theta & 0 \\ a_x a_y(1-c_\theta) + a_z s_\theta & a_y^2 + c_\theta(1-a_y^2) & a_y a_z(1-c_\theta) - a_x s_\theta & 0 \\ a_x a_z(1-c_\theta) - a_y s_\theta & a_y a_z(1-c_\theta) + a_x s_\theta & a_z^2 + c_\theta(1-a_z^2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Summary

- Different ways to describe rotations mathematically
  - Sequence of rotations around three axes (Euler angles)
  - Rotation around arbitrary angles (axis-angle representation)
  - Other options exist (quaternions, etc.)
- Rotations preserve
  - Angles
  - Lengths
  - Handedness of coordinate system
- Rigid transforms
  - Rotations and translations

Rotation matrices

- Orthonormal
  - Rows, columns are unit length and orthogonal
- Inverse of rotation matrix?

Rotation matrices

- Orthonormal
  - Rows, columns are unit length and orthogonal
- Inverse of rotation matrix?
  - Its transpose

$$\mathbf{R}(\mathbf{a}, \theta)^{-1} = \mathbf{R}(\mathbf{a}, \theta)^T$$

Rotations

- Given a rotation matrix $\mathbf{R}(\mathbf{a}, \theta)$
- How do we obtain $\mathbf{R}(\mathbf{a}, -\theta)$?

Rotations

- Given a rotation matrix $\mathbf{R}(\mathbf{a}, \theta)$
- How do we obtain $\mathbf{R}(\mathbf{a}, -\theta)$?

$$\mathbf{R}(\mathbf{a}, -\theta) = \mathbf{R}(\mathbf{a}, \theta)^{-1} = \mathbf{R}(\mathbf{a}, \theta)^{T}$$

Rotations

- Given a rotation matrix $\mathbf{R}(\mathbf{a}, \theta)$
- How do we obtain $\mathbf{R}(\mathbf{a}, -\theta)$?

$$\mathbf{R}(\mathbf{a}, -\theta) = \mathbf{R}(\mathbf{a}, \theta)^{-1} = \mathbf{R}(\mathbf{a}, \theta)^T$$

- How do we obtain $\mathbf{R}(\mathbf{a}, 2\theta), \mathbf{R}(\mathbf{a}, 3\theta)$ …?

Rotations

- Given a rotation matrix $\mathbf{R}(\mathbf{a}, \theta)$
- How do we obtain $\mathbf{R}(\mathbf{a}, -\theta)$?

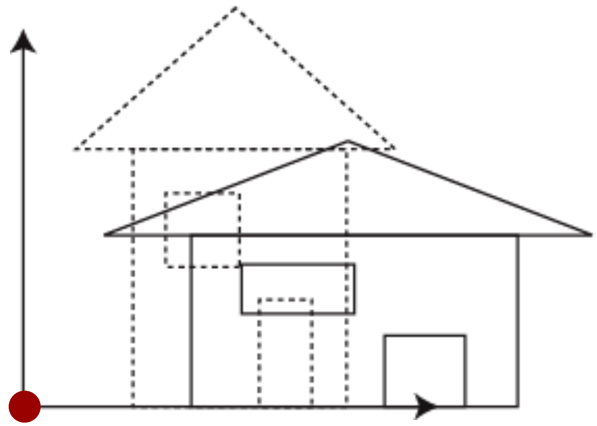
$$\mathbf{R}(\mathbf{a}, -\theta) = \mathbf{R}(\mathbf{a}, \theta)^{-1} = \mathbf{R}(\mathbf{a}, \theta)^T$$

- How do we obtain $\mathbf{R}(\mathbf{a}, 2\theta), \mathbf{R}(\mathbf{a}, 3\theta)$ …?

$$\mathbf{R}(\mathbf{a}, 2\theta) = \mathbf{R}(\mathbf{a}, \theta)^2 = \mathbf{R}(\mathbf{a}, \theta)\mathbf{R}(\mathbf{a}, \theta)$$

$$\mathbf{R}(\mathbf{a}, 3\theta) = \mathbf{R}(\mathbf{a}, \theta)^3 = \mathbf{R}(\mathbf{a}, \theta)\mathbf{R}(\mathbf{a}, \theta)\mathbf{R}(\mathbf{a}, \theta)$$

- Origin does not change



$$\mathbf{S}(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Scaling

- Inverse scaling?

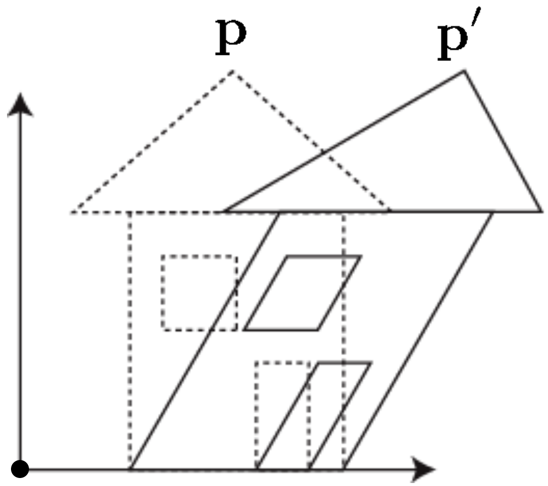
$$\mathbf{S}(s_x, s_y, s_z)^{-1} =$$

Scaling

- Inverse scaling?

$$\mathbf{S}(s_x, s_y, s_z)^{-1} = \mathbf{S}(1/s_x, 1/s_y, 1/s_z)$$

Shear



$$\mathbf{p}' = \begin{bmatrix} 1 & z \\ 0 & 1 \end{bmatrix} \mathbf{p}$$

- Pure shear if only one parameter is non-zero
- Cartoon-like effects

$$\mathbf{Z}(z_1 \ldots z_6) = \begin{bmatrix} 1 & z_1 & z_2 & 0 \\ z_3 & 1 & z_4 & 0 \\ z_5 & z_6 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Summary affine transformations

- Linear transformations (rotation, scale, shear, reflection) + translation

Vector space,

http://en.wikipedia.org/wiki/Vector_space

- vectors as [xyz] coordinates
- represents vectors
- linear transformations

Affine space

http://en.wikipedia.org/wiki/Affine_space

- points and vectors as [xyz1], [xyz0] homogeneous coordinates
- distinguishes points and vectors
- linear tranforms and translation

Summary affine transformations

- Implemented using 4x4 matrices, homogeneous coordinates
  - Last row of 4x4 matrix is always $[0\ 0\ 0\ 1]$
- Any such matrix represents an affine transformation in 3D
- Factorization into scale, shear, rotation, etc. is always possible, but non-trivial
  - Polar decomposition
    http://en.wikipedia.org/wiki/Polar_decomposition

Today

Transformations & matrices

- Introduction
- Matrices
- Homogeneous coordinates
- Affine transformations
- Concatenating transformations
- Change of coordinates
- Common coordinate systems

Concatenating transformations

- Build "chains" of transformations

$$\mathbf{M}_3, \mathbf{M}_2, \mathbf{M}_1 \in \mathbf{R}^{4\times 4}$$

- Apply M_1 followed by M_2 followed by M_3

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

- Overall transformation is an affine transformation

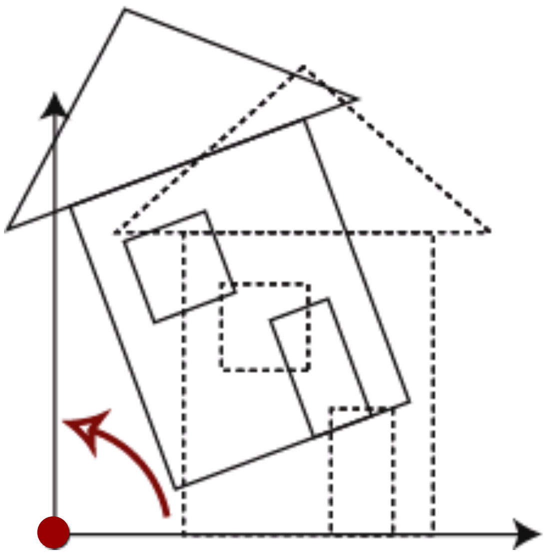
$$\mathbf{p}' = \mathbf{M}_3\mathbf{M}_2\mathbf{M}_1\mathbf{p} = \mathbf{M}\mathbf{p}$$

- Multiplication on the left

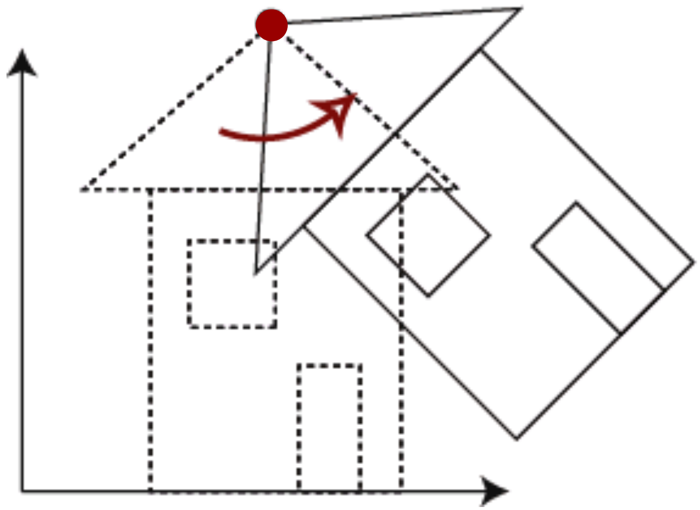
Concatenating transformations

- Result depends on order because matrix multiplication not commutative
- Thought experiment
  - Translation followed by rotation vs. rotation followed by translation

Rotating with pivot

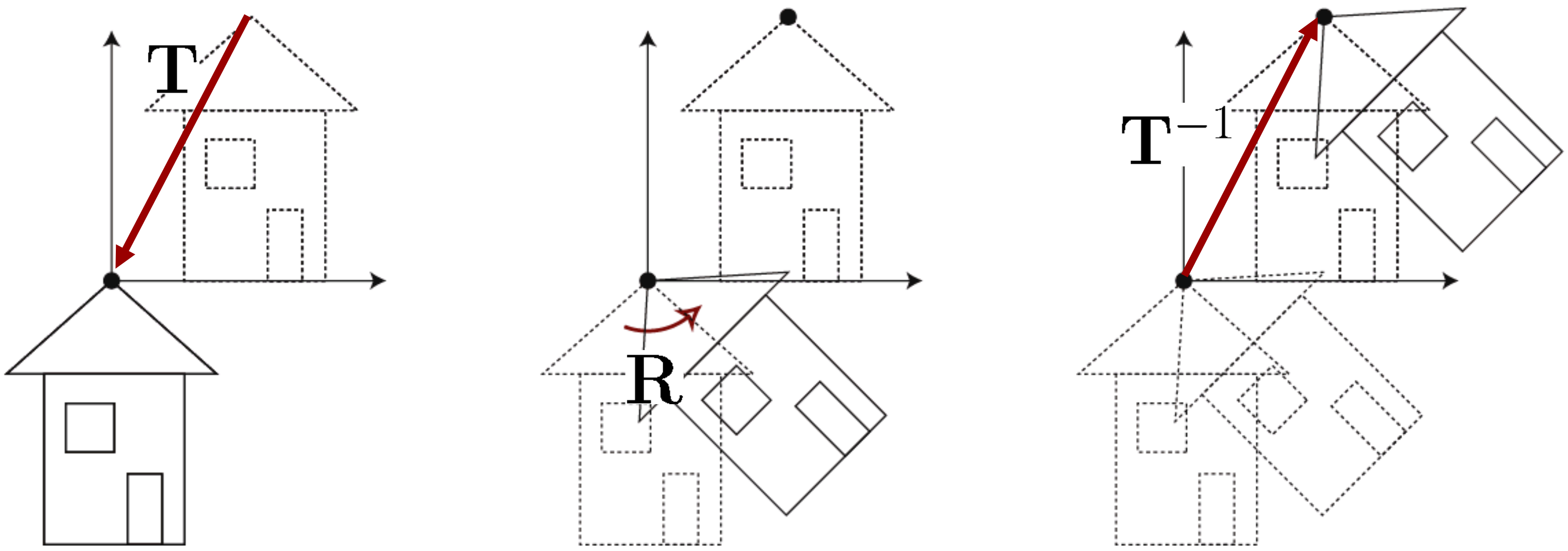


Rotation around origin



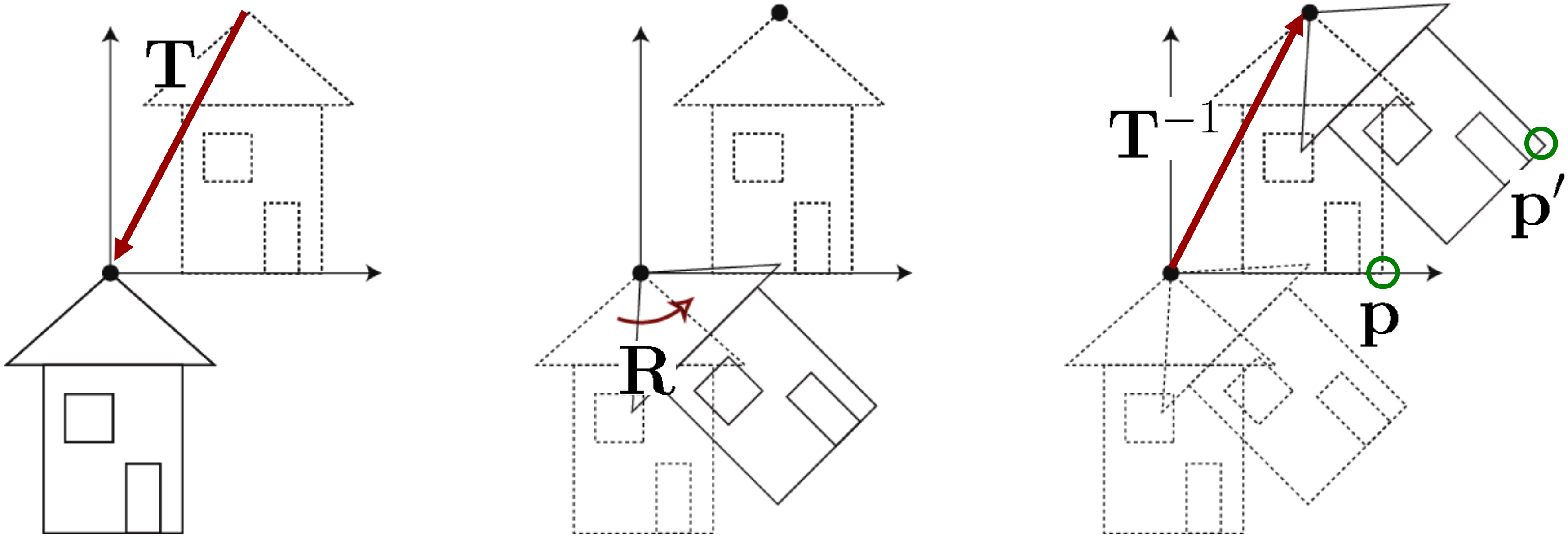
Rotation with pivot

Rotating with pivot



1. Translation $\mathbf{T}$ 2. Rotation $\mathbf{R}$ 3. Translation $\mathbf{T}^{-1}$

Rotating with pivot



1. Translation $\mathbf{T}$ 2. Rotation $\mathbf{R}$ 3. Translation $\mathbf{T}^{-1}$

$$\mathbf{p}' = \mathbf{T}^{-1}\mathbf{R}\mathbf{T}\mathbf{p}$$

Concatenating transformations

- Arbitrary sequence of transformations

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

$$\mathbf{M}_{total} = \mathbf{M}_3\mathbf{M}_2\mathbf{M}_1$$

$$\mathbf{p}' = \mathbf{M}_{total}\mathbf{p}$$

- Note: associativity

$$\mathbf{M}_{total} = (\mathbf{M}_3\mathbf{M}_2)\mathbf{M}_1 = \mathbf{M}_3(\mathbf{M}_2\mathbf{M}_1)$$

So either is valid

T=M3.multiply(M2); Mtotal=T.multiply(M1)

or

T=M2.multiply(M1); Mtotal=M3.multiply(T)

Transformation: summary

- Transformations are used for modeling
- Classes of transformation: rigid and affine
- Why we use homo. coordinates and matrices
- How to do matrix mults, inversion, transpose
- Homogenous coordinates, vectors vs. points
- Properties of affine transformations
- Transforms: translation, scale, rotation, shear
  - Only starting with 3D rotations – don't be concerned
- Order of transformations
  - They don't commute, but are associative
  - Translate to origin for scaling, rotation