CSE 167:
Introduction to Computer Graphics
Lecture #2: Coordinate Transformations

Jürgen P. Schulze, Ph.D.
University of California, San Diego
Fall Quarter 2013

Announcements

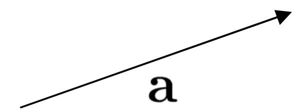
- Homework #1 due Friday Oct 4th at 1:30pm; presentation in CS basement lab 260
- Tip: Don't save anything on the C: drive of the lab PCs in Windows. You will lose it when you log out!

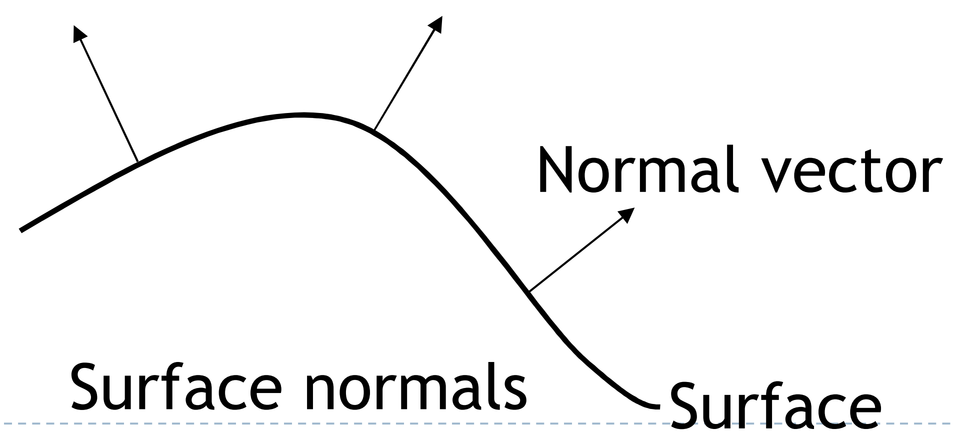
Lecture Overview

- Vectors and Matrices
- Linear Transformations
- Homogeneous Coordinates
- Affine Transformations

Vectors

- Direction and length in 3D
- Vectors can describe
  - Difference between two 3D points
  - Speed of an object
  - Surface normals (directions perpendicular to surfaces)





Vector arithmetic using coordinates

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

$$\mathbf{a} + \mathbf{b} = \begin{bmatrix} a_x + b_x \\ a_y + b_y \\ a_z + b_z \end{bmatrix} \qquad \mathbf{a} - \mathbf{b} = \begin{bmatrix} a_x - b_x \\ a_y - b_y \\ a_z - b_z \end{bmatrix}$$

$$-\mathbf{a} = \begin{bmatrix} -a_x \\ -a_y \\ -a_z \end{bmatrix} \qquad s\mathbf{a} = \begin{bmatrix} sa_x \\ sa_y \\ sa_z \end{bmatrix}$$

where s is a scalar

Vector Magnitude

- The magnitude (length) of a vector is:

$$|\mathbf{v}|^2 = v_x^2 + v_y^2 + v_z^2$$

$$|\mathbf{v}| = \sqrt{v_x^2 + v_y^2 + v_z^2}$$

- A vector with length of 1.0 is called *unit vector*
- We can also *normalize* a vector to make it a unit vector

$$\frac{\mathbf{v}}{|\mathbf{v}|}$$

- Unit vectors are often used as **surface normals**

Dot Product

$$\mathbf{a} \cdot \mathbf{b} = \sum a_i b_i$$

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z$$

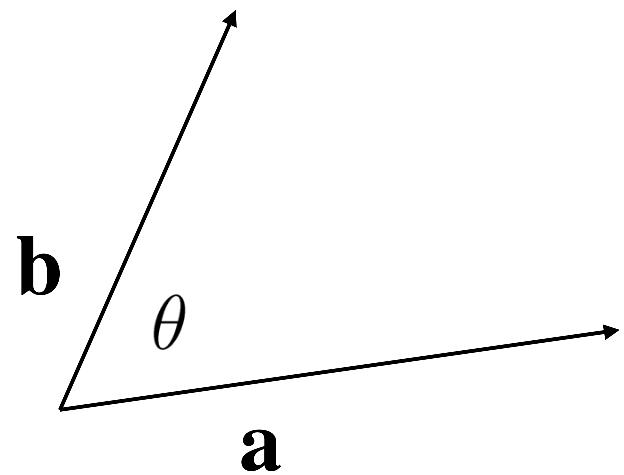
$$\mathbf{a} \cdot \mathbf{b} = |a||b| \cos \theta$$

Angle Between Two Vectors

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}|\cos\theta$$

$$\cos\theta = \left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|} \right)$$

$$\theta = \cos^{-1}\left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|} \right)$$



Cross Product

$\mathbf{a} \times \mathbf{b}$ is a vector *perpendicular* to both $\mathbf{a}$ and $\mathbf{b}$, in the direction defined by the right hand rule

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

$|\mathbf{a} \times \mathbf{b}| =$ area of parallelogram $\mathbf{ab}$

$|\mathbf{a} \times \mathbf{b}| = 0$ if $\mathbf{a}$ and $\mathbf{b}$ are parallel (or one or both degenerate)

Cross product

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{bmatrix}$$

Sample Vector Class in C++

```
class Vector3 {
public:
    float x,y,z;
    Vector3()                                   {x=0.0; y=0.0; z=0.0;}
    Vector3(float x0,float y0,float z0)         {x=x0; y=y0; z=z0;}
    void set(float x0,float y0,float z0)        {x=x0; y=y0; z=z0;}
    void add(Vector3 &a)                        {x+=a.x; y+=a.y; z+=a.z;}
    void add(Vector3 &a,Vector3 &b)             {x=a.x+b.x; y=a.y+b.y; z=a.z+b.z;}
    void subtract(Vector3 &a)                   {x-=a.x; y-=a.y; z-=a.z;}
    void subtract(Vector3 &a,Vector3 &b)        {x=a.x-b.x; y=a.y-b.y; z=a.z-b.z;}
    void negate()                               {x=-x; y=-y; z=-z;}
    void negate(Vector3 &a)                     {x=-a.x; y=-a.y; z=-a.z;}
    void scale(float s)                         {x*=s; y*=s; z*=s;}
    void scale(float s,Vector3 &a)              {x=s*a.x; y=s*a.y; z=s*a.z;}
    float dot(Vector3 &a)                       {return x*a.x+y*a.y+z*a.z;}
    void cross(Vector3 &a,Vector3 &b)           {x=a.y*b.z-a.z*b.y; y=a.z*b.x-a.x*b.z; z=a.x*b.y-a.y*b.x;}
    float magnitude()                           {return sqrt(x*x+y*y+z*z);}
    void normalize()                            {scale(1.0/magnitude());}
};
```

Matrices

- 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 **m = n**
- In graphics often **m = n = 3; m = 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

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

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

Matrix-Vector Multiplication

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

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

Identity Matrix

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

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

Matrix Inverse

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

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

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

OpenGL Matrices

- Vectors are column vectors
- "Column major" ordering
- Matrix elements stored in array of floats

  float M[16];
- Corresponding matrix elements:

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

Lecture Overview

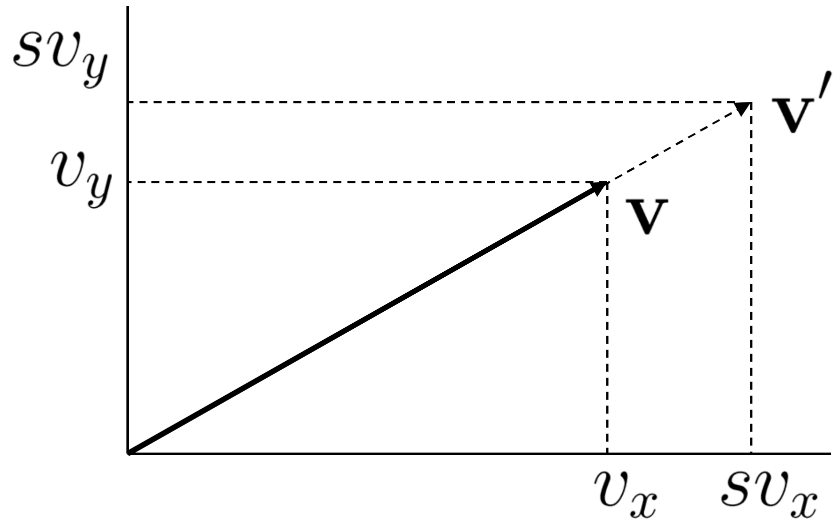
- Vectors and Matrices
- Linear Transformations
- Homogeneous Coordinates
- Affine Transformations

Linear Transformations

- Scaling, shearing, rotation, reflection of vectors, and combinations thereof
- Implemented using matrix multiplications

Scaling

- Uniform scaling matrix in 2D

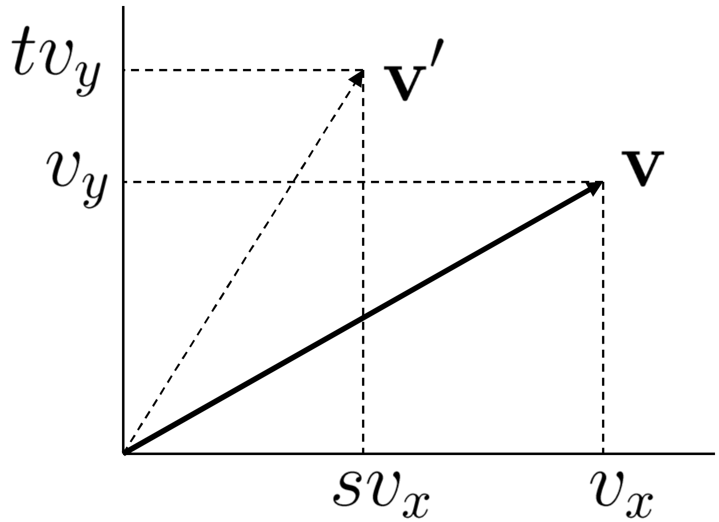


- Analogous in 3D

$$\begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \mathbf{v} = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} v'_x \\ v'_y \end{bmatrix} = \mathbf{v}'$$

Scaling

- Nonuniform scaling matrix in 2D

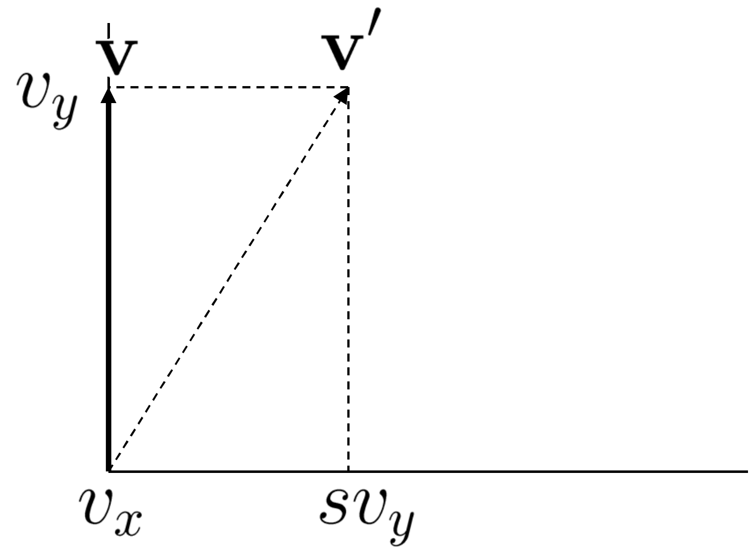


- Analogous in 3D

$$\begin{bmatrix} s & 0 \\ 0 & t \end{bmatrix} \mathbf{v} = \begin{bmatrix} s & 0 \\ 0 & t \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} v'_x \\ v'_y \end{bmatrix} = \mathbf{v}'$$

Shearing

- Shearing along x-axis in 2D



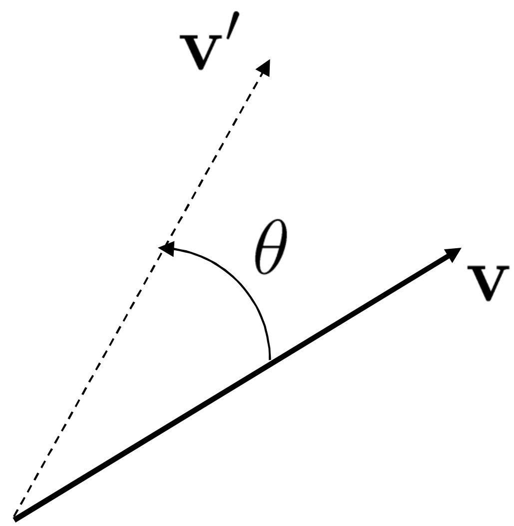
- Analogous for y-axis, in 3D

$$\mathbf{v}' = \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix} \mathbf{v}$$

Rotation in 2D

- Convention: positive angle rotates counterclockwise
- Rotation matrix

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



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

Rotation in 3D

Rotation around coordinate axes

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

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

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

Rotation in 3D

- Concatenation of rotations around x, y, z axes

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

- $\theta_x, \theta_y, \theta_z$ are called Euler angles
- Result depends on matrix order!

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

Rotation in 3D

Around arbitrary axis

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

- Rotation axis **a**
  - **a** must be a unit vector: $|\mathbf{a}| = 1$
- Right-hand rule applies for direction of rotation
  - Counterclockwise rotation

Lecture Overview

- Vectors and Matrices
- Linear Transformations
- Homogeneous Coordinates
- Affine Transformations

Homogeneous Coordinates

- Generalization: homogeneous point

$$\mathbf{p}_h = wp_x\mathbf{x} + wp_y\mathbf{y} + wp_z\mathbf{z} + w\mathbf{o} \qquad \begin{bmatrix} wp_x \\ wp_y \\ wp_z \\ w \end{bmatrix}$$

- Homogeneous coordinate
- Corresponding 3D point: divide by homogeneous coordinate w

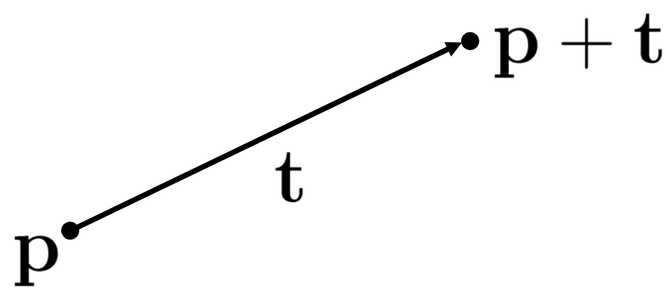
$$\mathbf{p} = p_x\mathbf{x} + p_y\mathbf{y} + p_z\mathbf{z} + \mathbf{o} \qquad \begin{bmatrix} wp_x/w \\ wp_y/w \\ wp_z/w \\ w/w \end{bmatrix}$$

Homogeneous coordinates

- Usually for 3D points you choose $w = 1$
- For 3D vectors $w = 0$
- Benefit: same representation for vectors and points

Translation

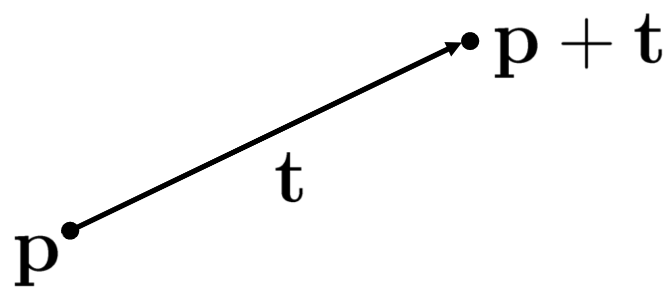
Using homogeneous coordinates



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

Translation

Using homogeneous coordinates



Matrix notation

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

Translation matrix

Transformations

- Add 4th row/column to 3 x 3 transformation matrices
- Example: rotation

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

$$\begin{bmatrix} & \mathbf{R}(\mathbf{a}, \theta) & & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Transformations

Concatenation of transformations:

- Arbitrary transformations (scale, shear, rotation, translation) $\mathbf{M}_3, \mathbf{M}_2, \mathbf{M}_1 \in \mathbf{R}^{4\times 4}$
- Build "chains" of transformations $\mathbf{p}'_h = \mathbf{M}_3\mathbf{M}_2\mathbf{M}_1\mathbf{p}_h$
- Result depends on order

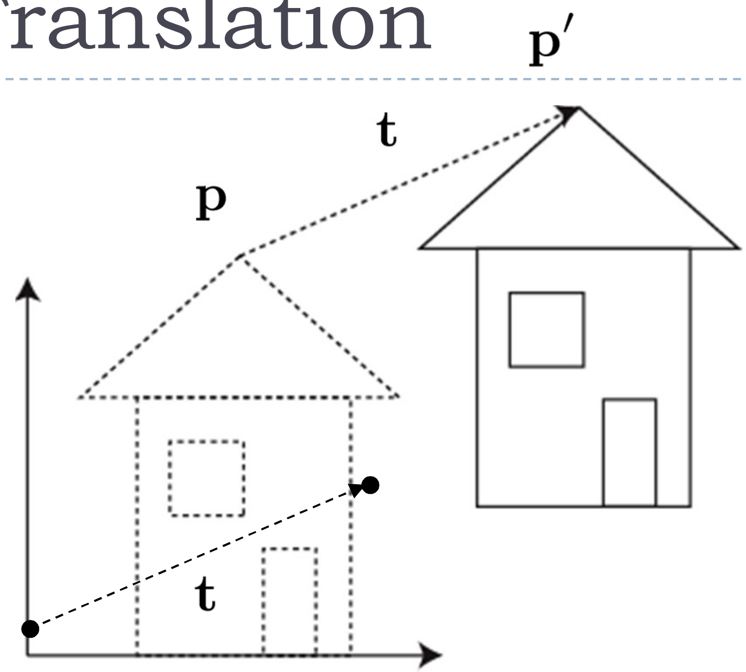
Lecture Overview

- Vectors and Matrices
- Linear Transformations
- Homogeneous Coordinates
- Affine Transformations

Affine transformations

- Generalization of linear transformations
  - Scale, shear, rotation, reflection (linear)
  - *Translation*
- Preserve straight lines, parallel lines
- Implementation using 4x4 matrices and homogeneous coordinates

Translation



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

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

$$\begin{bmatrix} p'_x \\ p'_y \\ p'_z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

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

Translation

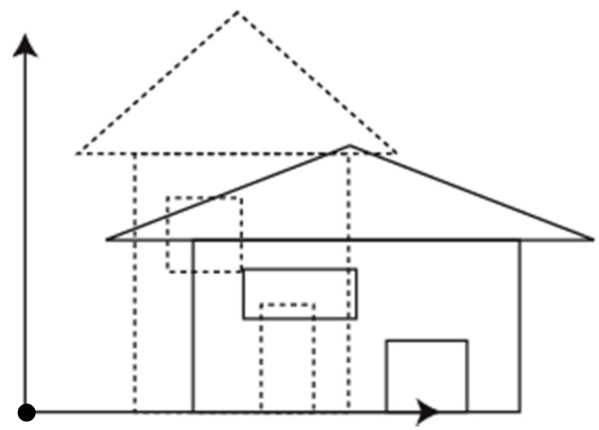
- Inverse translation

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

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

Scaling

- Origin does not change



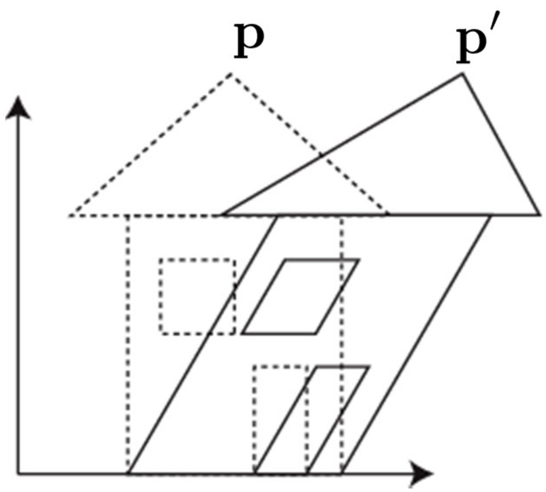
$$\mathbf{S}(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Scaling

- Inverse of scale:

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

Shear



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

- Pure shear if only one parameter is non-zero

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

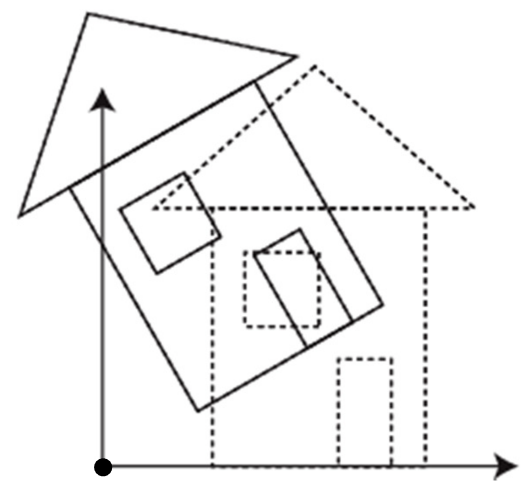
Rotation around coordinate axis

- Origin does not change

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

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

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



Rotation around arbitrary axis

- Origin does not change
- Angle θ, unit axis **a**
- $c_\theta = \cos\theta, s_\theta = \sin\theta$

$$\mathbf{R}(\mathbf{a},\theta) = \begin{bmatrix} a_x^2 + c_\theta(1-a_x^2) & a_xa_y(1-c_\theta) - a_zs_\theta & a_xa_z(1-c_\theta) + a_ys_\theta & 0 \\ a_xa_y(1-c_\theta) + a_zs_\theta & a_y^2 + c_\theta(1-a_y^2) & a_ya_z(1-c_\theta) - a_xs_\theta & 0 \\ a_xa_z(1-c_\theta) - a_ys_\theta & a_ya_z(1-c_\theta) + a_xs_\theta & a_z^2 + c_\theta(1-a_z^2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotation matrices

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

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

Videos

- Linear Algebra - Affine Matrix Transformations With Shapes
  - http://www.youtube.com/watch?v=4INR_27DM4U
- Online Graphics Basic Math: Matrices
  - http://www.youtube.com/watch?v=RkX2U16QyY8