University of British Columbia
CPSC 314 Computer Graphics
Jan-Apr 2016

Tamara Munzner

Transformations 2

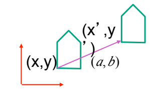
http://www.ugrad.cs.ubc.ca/~cs314/Vjan2016

Readings for Transformations 1-5

- Shirley/Marschner
  - Ch 6: Transformation Matrices
    - *except* 6.1.6, 6.3.1
  - Sect 12.2 Scene Graphs
- Gortler
  - Ch 2: Linear, Sec 2.5-2.6
  - Ch 3: Affine
  - Ch 4: Respect
  - Ch 5: Frames in Graphics, 5.3-5.4

2D Transformations

2D Translation



vector addition

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

matrix multiplication

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

matrix multiplication

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

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

translation multiplication matrix??

Linear Transformations

- linear transformations are combinations of
  - shear
  - scale
  - rotate
  - reflect

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

- 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

Challenge

- matrix multiplication
  - for everything except translation
  - can we just do everything with multiplication?
    - then could just do composition, no special cases

Homogeneous Coordinates

- represent 2D coordinates (x,y) with 3-vector (x,y,1)
  - use 3x3 matrices for 2D transformations

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

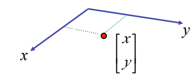
$$\mathbf{T}ranslation=\begin{bmatrix}1 & 0 & T_x\\0 & 1 & T_y\\0 & 0 & 1\end{bmatrix}$$

- use rightmost column

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

Homogeneous Coordinates Geometrically

- point in 2D cartesian

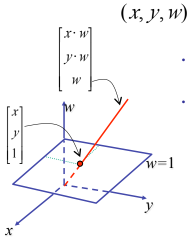


$$\begin{bmatrix}x\\y\end{bmatrix}$$

Homogeneous Coordinates Geometrically

homogeneous **cartesian**

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

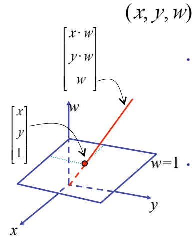


- point in 2D cartesian + weight w = point P in 3D homog. coords
- multiples of (x,y,w)
  - form a line L in 3D
  - all homogeneous points on L represent same 2D cartesian point
  - example: (2,2,1) = (4,4,2) = (1,1,0.5)

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

Affine Transformations

- affine transforms are combinations of
  - linear transformations
  - translations

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

- properties of affine transformations
  - origin does not necessarily map to origin
  - lines map to lines
  - parallel lines remain parallel
  - ratios are preserved
  - closed under composition

Homogeneous Coordinates Summary

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