Affine transformations

Brian Curless
CSE 457
Autumn 2010

Reading

Required:

- Angel 4.1, 4.6-4.10

Further reading:

- Angel, the rest of Chapter 4
- Foley, et al, Chapter 5.1-5.5.
- David F. Rogers and J. Alan Adams, *Mathematical Elements for Computer Graphics*, 2nd Ed., McGraw-Hill, New York, 1990, Chapter 2.

Geometric transformations

Geometric transformations will map points in one space to points in another: $(x', y', z') = \mathbf{f}(x, y, z)$.

These tranformations can be very simple, such as scaling each coordinate, or complex, such as non-linear twists and bends.

We'll focus on transformations that can be represented easily with matrix operations.

We'll start in 2D...

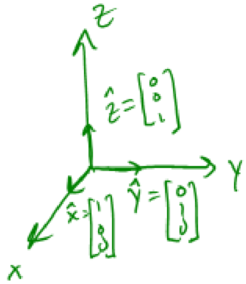
Vector representation

We can represent a **point**, **p** = (x,y), in the plane or p=(x,y,z) in 3D space

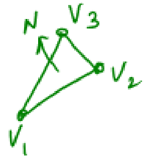
- as column vectors $\begin{bmatrix} x \\ y \end{bmatrix}$ $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$

- as row vectors $\begin{bmatrix} x & y \end{bmatrix}$ $\begin{bmatrix} x & y & z \end{bmatrix}$

Canonical axes



right-handed coord. systems



Vector length and dot products

$\hat{u}\cdot v = \|v\|\cos\theta$

$u=\begin{bmatrix}u_x\\u_y\\u_z\end{bmatrix}$ $v=\begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix}$

$\|u\| = \sqrt{u_x^2+u_y^2+u_z^2} = \sqrt{u\cdot u}$

$u\cdot v \equiv u_x v_x + u_y v_y + u_z v_z = v\cdot u$

$= u^T v = \begin{bmatrix}u_x & u_y & u_z\end{bmatrix}\begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix}$

$u\cdot v = \|u\|\,\|v\|\cos\theta$

$u\cdot v = 0 \iff \underline{\theta = 90, 270}$ or $\|u\|$ or $\|v\| = 0$

$\perp$ orthogonal

$\|u\| = \|v\| = 1 \Rightarrow u\cdot v = \cos\theta$

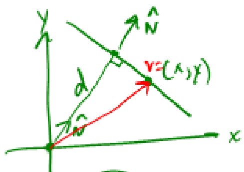
$\hat{u} = \dfrac{u}{\|u\|}$

direction

$u = \|u\|\cdot\hat{u}$

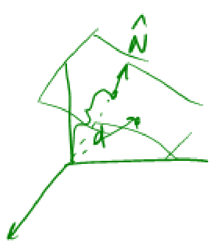
magnitude direction

The plane equation



$\hat{N}\cdot v = d$ eq. of a line or a plane

true in 3D as well



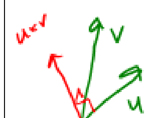
$ax + by + cz = \underset{\sim}{d}$

$a^2 + b^2 + c^2 = 1$

$\hat{N}\cdot\begin{bmatrix}x\\y\\z\end{bmatrix} = d$

$\hat{N} = \begin{bmatrix}a\\b\\c\end{bmatrix}$

Vector cross products



$u\times v = \left|\begin{matrix}\hat{x} & \hat{y} & \hat{z}\\ u_x & u_y & u_z\\ v_x & v_y & v_z\end{matrix}\right| = \hat{x}(u_y v_z - u_z v_y) + \hat{y}(u_z v_x - u_x v_z) + \hat{z}(u_x v_y - u_y v_x) = \begin{bmatrix}u_y v_z - u_z v_y\\ u_z v_x - u_x v_z\\ u_x v_y - u_y v_x\end{bmatrix}$

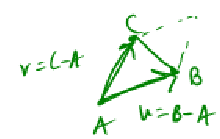
$\|u\times v\| = \|u\|\,\|v\|\,|\sin\theta|$

$u\times v = -v\times u$

$(u\times v)\cdot u = (u\times v)\cdot v = 0$

$u\times u = 0$

Area = $\|u\times v\|$



$\frac{1}{2}\|u\times v\| = \text{Area}(\Delta ABC)$

$$A \cdot A^{-1} = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$(AB)^{-1} \stackrel{?}{=} B^{-1}A^{-1}$$

$$(AB)^{-1}A \stackrel{?}{=} B^{-1}\cancel{A^{-1}A}$$

$$(AB)^{-1}AB \stackrel{?}{=} B^{-1}B$$

$$(AB)^{-1}(AB) \stackrel{\checkmark}{=} I$$

Representation, cont.

$$(AB)^T = B^T A^T$$

We can represent a **2-D transformation** M by a matrix

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

If **p** is a column vector, M goes on the left:

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

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

If **p** is a row vector, M^T goes on the right:

$$\mathbf{p}' = \mathbf{p}M^T$$

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

We will use **column vectors**.

Two-dimensional transformations

Here's all you get with a 2 x 2 transformation matrix M:

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

So:

$$x' = ax + by$$
$$y' = cx + dy$$

We will develop some intimacy with the elements a, b, c, d...

Identity

Suppose we choose $a=d=1$, $b=c=0$:

- Gives the **identity** matrix:

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

- Doesn't move the points at all

Scaling

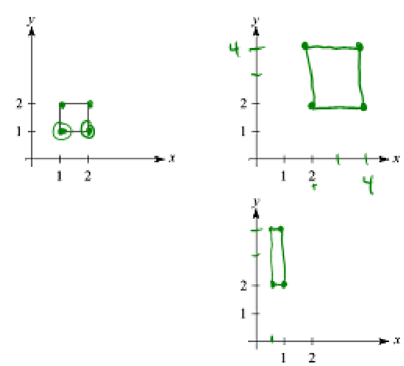
Suppose we set $b=c=0$, but let a and d take on any *positive* value:

- Gives a **scaling** matrix:

$$\begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Provides **differential (non-uniform) scaling** in x and y:

$$x' = ax$$
$$y' = dy$$



$$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

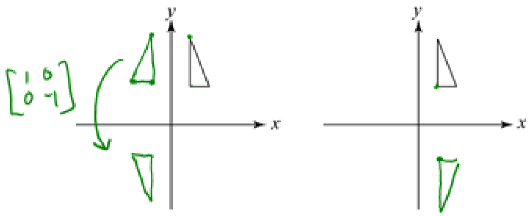
$$\begin{bmatrix} 1/2 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{1}{2}x \\ 2y \end{bmatrix}$$

Reflection (mirroring)

Suppose we keep $b=c=0$, but let either a or d go negative.

Examples:

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



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

Shear

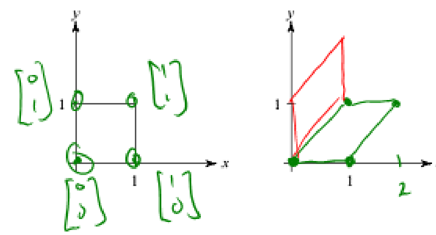
Now let's leave $a=d=1$ and experiment with b. . . .

The matrix

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

gives:

$$x' = x + by$$
$$y' = y$$



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

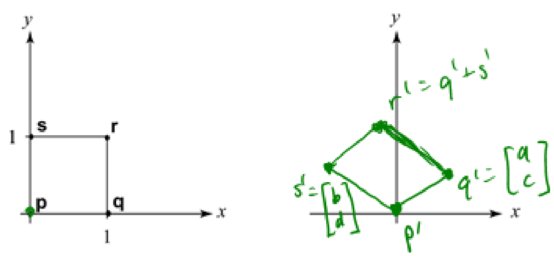
$$\begin{bmatrix}1&0\\1&1\end{bmatrix}$$

Effect on unit square

Let's see how a general 2 x 2 transformation M affects the unit square:

$$\begin{bmatrix}a&b\\c&d\end{bmatrix}\begin{bmatrix}\mathbf{p}&\mathbf{q}&\mathbf{r}&\mathbf{s}\end{bmatrix}=\begin{bmatrix}\mathbf{p'}&\mathbf{q'}&\mathbf{r'}&\mathbf{s'}\end{bmatrix}$$

$$\begin{bmatrix}a&b\\c&d\end{bmatrix}\begin{bmatrix}0&1&1&0\\0&0&1&1\end{bmatrix}=\begin{array}{c}\begin{bmatrix}0&a&a+b&b\\0&c&c+d&d\end{bmatrix}\\ p' \quad\; q' \quad\;\; r' \quad\;\; s'\end{array}$$



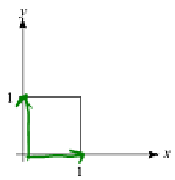
Effect on unit square, cont.

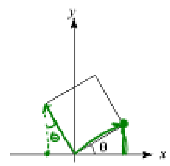
Observe:

- Origin invariant under M
- M can be determined just by knowing how the corners (1,0) and (0,1) are mapped
- a and d give x- and y-scaling
- b and c give x- and y-shearing

Rotation

From our observations of the effect on the unit square, it should be easy to write down a matrix for "rotation about the origin":





- $\begin{bmatrix}1\\0\end{bmatrix} \rightarrow \begin{bmatrix}\cos\theta\\ \sin\theta\end{bmatrix}$
- $\begin{bmatrix}0\\1\end{bmatrix} \rightarrow \begin{bmatrix}-\sin\theta\\ \cos\theta\end{bmatrix}$

Thus,

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

Degrees of freedom

For any transformation, we can count its **degrees of freedom** – the number of independent (though not necessarily unique) parameters needed to specify the transformation.

One way to count them is to add up all the apparently free variables and subtract the number of (independent) equations that constrain them.

How many degrees of freedom does an arbitrary 2X2 transformation have?

$$\begin{bmatrix}a & b\\ c & d\end{bmatrix} \quad 4$$

How many degrees of freedom does a 2D rotation have?

$$\begin{bmatrix}u & v\end{bmatrix}$$

$$\left.\begin{aligned} \|u\|=1 \rightarrow u\cdot u &= 1\\ v\cdot v &= 1\\ u\cdot v &= 0 \end{aligned}\right\} \text{3 constraints}$$

$$DOF = 4 - 3 = 1$$

Limitations of the 2 x 2 matrix

A 2 x 2 linear transformation matrix allows

- Scaling
- Rotation
- Reflection
- Shearing

Q: What important operation does that leave out?

translation

Homogeneous coordinates

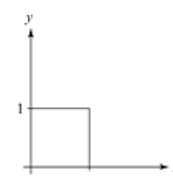
We can loft the problem up into 3-space, adding a third component to every point:

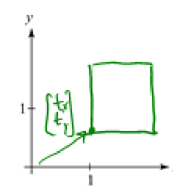
$$\begin{bmatrix}x\\y\end{bmatrix} \rightarrow \begin{bmatrix}x\\y\\1\end{bmatrix}$$

Adding the third "*w*" component puts us in **homogenous coordinates**.

Then, transform with a 3 x 3 matrix:

$$\begin{bmatrix}x'\\y'\\w'\end{bmatrix} = T(\mathbf{t})\begin{bmatrix}x\\y\\1\end{bmatrix} = \begin{bmatrix}1 & 0 & t_x\\ 0 & 1 & t_y\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\y\\1\end{bmatrix} = \begin{bmatrix}x+t_x\\ y+t_y\\ 1\end{bmatrix}$$





$$\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 1/2\\ 0 & 0 & 1\end{bmatrix}$$

. . . gives **translation**!

Affine transformations

The addition of translation to linear transformations gives us **affine transformations**.

In matrix form, 2D affine transformations always look like this:

$$M = \begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix} = \left[\begin{array}{cc|c} \multicolumn{2}{c|}{A} & \mathbf{t} \\ \hline 0 & 0 & 1 \end{array}\right]$$

2D affine transformations always have a bottom row of [0 0 1].

An "affine point" is a "linear point" with an added w-coordinate which is always 1:

$$\mathbf{p}_{\text{aff}} = \begin{bmatrix} \mathbf{p}_{\text{lin}} \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Applying an affine transformation gives another affine point:

$$M\mathbf{p}_{\text{aff}} = \begin{bmatrix} A\mathbf{p}_{\text{lin}} + \mathbf{t} \\ 1 \end{bmatrix}$$

Rotation about arbitrary points

Until now, we have only considered rotation about the origin.

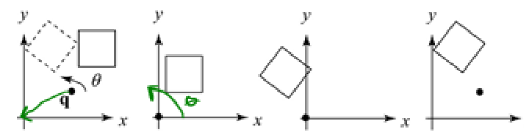
With homogeneous coordinates, you can specify a rotation, θ about any point $\mathbf{q} = [q_x\ q_y\ 1]^T$ with a matrix:

$T(v)$

$R(\theta)$

$$T(q) = \begin{bmatrix} 1 & 0 & q_x \\ 0 & 1 & q_y \\ 0 & 0 & 1 \end{bmatrix}$$

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



$M \neq$ $T(-q)$ $R(\theta)$ $T(q)$

1. Translate **q** to origin
2. Rotate
3. Translate back

$$M = T(q)\,R(\theta)\,T(-q)$$

<u>Note</u>: Transformation order is important!!

Points and vectors

Vectors have an additional coordinate of w=0. Thus, a change of origin has no effect on vectors.

$q \to p$

$$p - q \to \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} - \begin{bmatrix} q_x \\ q_y \\ 1 \end{bmatrix} = \begin{bmatrix} \cdots \\ \cdots \\ 0 \end{bmatrix}$$

Q: What happens if we multiply a vector by an affine matrix?

$$\begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_x \\ u_y \\ 0 \end{bmatrix} = \begin{bmatrix} au_x + bu_y \\ cu_x + du_y \\ 0 \end{bmatrix}$$

These representations reflect some of the rules of affine operations on points and vectors:

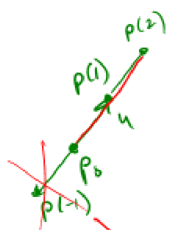
vector + vector → vector
scalar · vector → vector
point - point → vector
point + vector → point
point + point → chaos

points → point: $\sum \alpha_i P_i \to Q$, $\sum \alpha_i = 1$

points → vector: $\sum \alpha_i P_i \to V$, $\sum \alpha_i = 0$

One useful combination of affine operations is:

$$\mathbf{p}(t) = \mathbf{p}_o + t\mathbf{u}$$



Q: What does this describe?

$t \in [-\infty, \infty] \to$ line

$t \in [0, \infty] \to$ half line, ray

Basic 3-D transformations: scaling

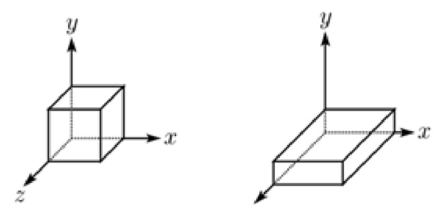
Some of the 3-D affine transformations are just like the 2-D ones.

In this case, the bottom row is always [0 0 0 1].

For example, scaling:

$$S^{-1}(s_x, s_y, s_z) = S\left(\frac{1}{s_x}, \frac{1}{s_y}, \frac{1}{s_z}\right)$$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

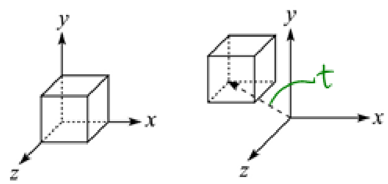


Translation in 3D

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

t

$T^{-1}(t_x, t_y, t_z)$
$= T(-t_x, -t_y, t_z)$



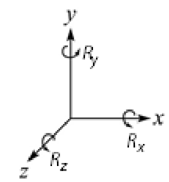
Rotation in 3D

Rotation now has more possibilities in 3D:

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 & 0 \\ \sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Use right hand rule

Rotation in 3D (cont'd)

$R^T R = ?$

$$\begin{bmatrix} \cdots u^T \cdots \\ \cdots v^T \cdots \\ \cdots w^T \cdots \end{bmatrix} \begin{bmatrix} u & v & w \end{bmatrix} = \begin{bmatrix} u\cdot u & u\cdot v & u\cdot w \\ v\cdot u & v\cdot v & v\cdot w \\ w\cdot u & w\cdot v & w\cdot w \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I$$

$\boxed{R^{-1} = R^T}$

How many degrees of freedom are there in an arbitrary rotation?

$A_{3\times3}$ 9 variables in A

$\begin{bmatrix} u & v & w \end{bmatrix}$

$u\cdot u = 1$ $\quad u\cdot v = 0$
$v\cdot v = 1$ $\quad v\cdot w = 0$
$w\cdot w = 1$ $\quad u\cdot w = 0$

6 constraints

$DOFs = 9 - 6 = 3$

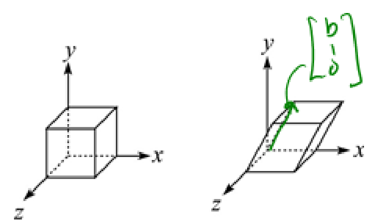
How else might you specify a 3D rotation?

θ - 1 var $\quad \|v\| = 1 \quad$ 4 vars - 1 constr = 3
v - 3 var

Shearing in 3D

Shearing is also more complicated. Here is one example:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & b & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

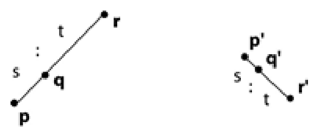


We call this a shear with respect to the x-z plane.

Properties of affine transformations

Here are some useful properties of affine transformations:

- Lines map to lines
- Parallel lines remain parallel
- Midpoints map to midpoints (in fact, ratios are always preserved)



$$\text{ratio} = \frac{\|\mathbf{pq}\|}{\|\mathbf{qr}\|} = \frac{s}{t} = \frac{\|\mathbf{p'q'}\|}{\|\mathbf{q'r'}\|}$$

Affine transformations in OpenGL

OpenGL maintains a "modelview" matrix that holds the current transformation **M.**

The modelview matrix is applied to points (usually vertices of polygons) before drawing.

It is modified by commands including:

- `glLoadIdentity()` $\mathbf{M} \leftarrow \mathbf{I}$
  - set **M** to identity
- `glTranslatef(t_x, t_y, t_z)` $\mathbf{M} \leftarrow \mathbf{MT}$
  - translate by (t_x, t_y, t_z)
- `glRotatef(θ, x, y, z)` $\mathbf{M} \leftarrow \mathbf{MR}$
  - rotate by angle θ about axis (x, y, z)
- `glScalef(s_x, s_y, s_z)` $\mathbf{M} \leftarrow \mathbf{MS}$
  - scale by (s_x, s_y, s_z)

Note that OpenGL adds transformations by *postmultiplication* of the modelview matrix.

Summary

What to take away from this lecture:

- All the names in boldface.
- How points and transformations are represented.
- How to compute lengths, dot products, and cross products of vectors, and what their geometrical meanings are.
- What all the elements of a 2 x 2 transformation matrix do and how these generalize to 3 x 3 transformations.
- What homogeneous coordinates are and how they work for affine transformations.
- How to concatenate transformations.
- The mathematical properties of affine transformations.