EECS 487: Interactive Computer Graphics

Lecture 9:
- Lectures 1-8 recap
- 2D Modeling Transforms
- Affine Transforms

Math Tools

Mathematical tools in CG:
1. implicit line equation
2. half-spaces and in/out determination
3. parametric/barycentric coordinates and interpolation
4. dot product and projection and intersection
5. cross product to compute area and orientation
6. implicit plane equation
7. sampling and filtering (averaging)

> "Computer graphics: Mathematics made visible."
> — *James*

Along the way we learned:
1. line rasterization and coloring
2. triangle rasterization and coloring
3. clipping and visibility determination
4. anti-aliasing and compositing

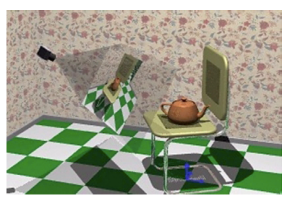
> "Much of graphics is just translating math directly into code. The cleaner the math, the cleaner the resulting code."
> — *Shirley*

The non-programmable parts of the pipeline

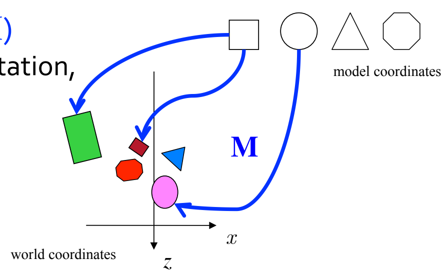
Where are We Headed Next?

Image synthesis is just like taking a picture

Application provides:
- scene description: objects, lights, camera
- different coordinate systems used:
  - model/object, world, eye/camera



Modeling Transform (**M**)
- change the position, orientation, deformation of objects to compose scene

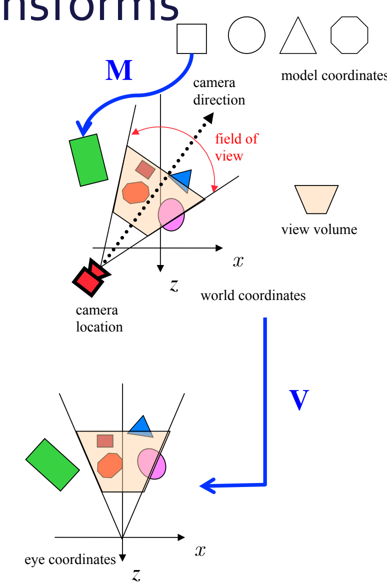


Gillies09

Model and View Transforms

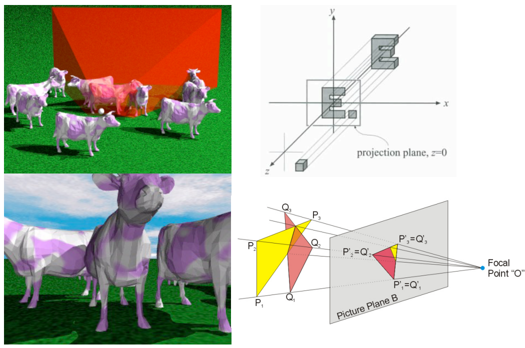
Viewing Transform (**V**)
- positioning the camera and its orientation
- where the user views the world from and how much of the world is visible

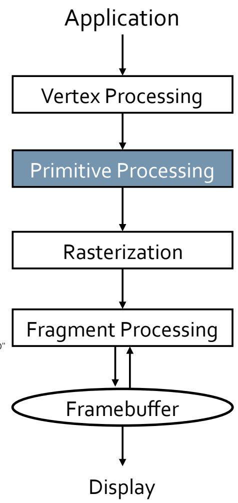
In OpenGL 2.1, Model and Viewing transforms encoded into a single matrix (**VM**)
- `GL_MODELVIEW_MATRIX`



Projection and Viewport Transforms

Projection Transform:
- framing the shot and zooming the lens





Viewport Transform:
- enlarging or reducing the size of the print

What is a Transformation?

Geometric transformations map points in one space to points in another: $\mathbf{p}' = f(\mathbf{p})$

Transformations provide a mechanism for manipulating geometric models and are essential pieces of graphics systems

- e.g., OpenGL/Direct3D and PostScript use them extensively

Transformations are used to:

- position objects in a scene (modeling)
- change the shape of objects
- create multiple copies of objects
- project for virtual cameras
- create animations
- etc.

General (Free-form) Transformation

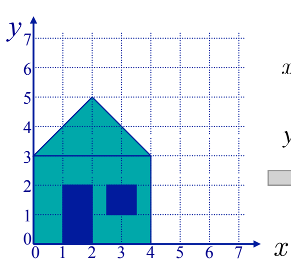
Sederberg & Parry 86

Does not preserve straight lines

Computationally involved

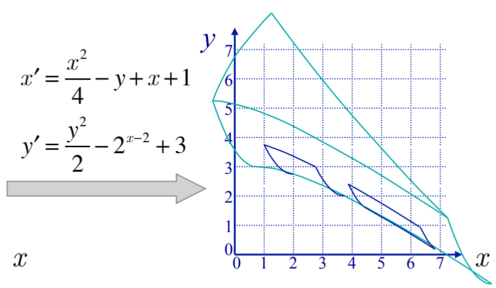
Not as pervasively used

Not covered . . . .



$$x' = \frac{x^2}{4} - y + x + 1$$

$$y' = \frac{y^2}{2} - 2^{x-2} + 3$$



Merrello8

Transformations We Cover

… and characteristics they preserve:

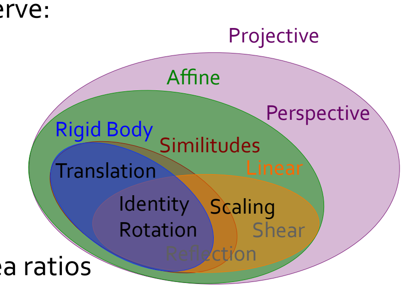
Rigid Body/Euclidean:
- angles, lengths, areas

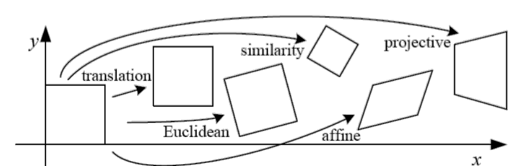
Linear:
- linear combination

Affine:
- parallel lines, length ratios, area ratios

Similitudes/similarity:
- angles, length ratios

Perspective:
- collinearity, cross-ratios

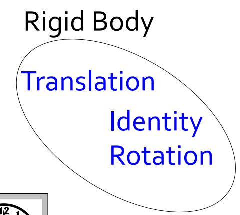




DeMenthon, Durando8

Rigid-Body/Euclidean Transforms

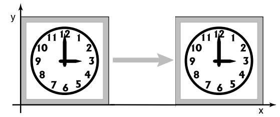
Preserves:
- absolute distances
- angles (size and sign)



Translation:

$x' = x + t_x$

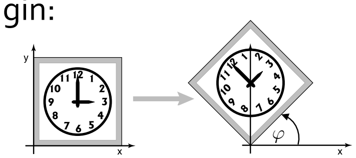
$y' = y + t_y$



Rotation about the origin:

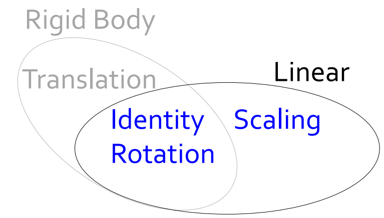
$x' = x \cos \varphi - y \sin \varphi$

$y' = x \sin \varphi + y \cos \varphi$



Shirley02
Durand08

Linear Transforms

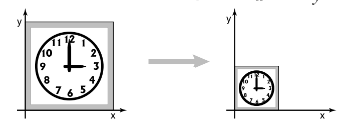


Scaling:

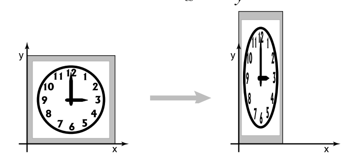
$x' = s_x x$

$y' = s_y y$

uniform/isotropic: $s_x = s_y$



non-uniform/anisotropic/
differential: $s_x \neq s_y$



Shirley02
Durand08

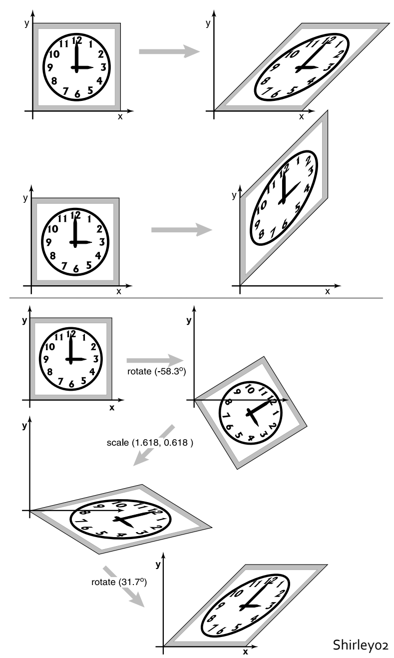
How About Shear?
Non-Primitive

Shear:

$x' = x + h_x y$

$y' = h_y x + y$

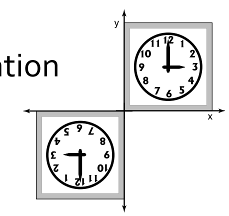
Non-primitive: it can be shown (by SVD) that shear is a combination of rotations and scale



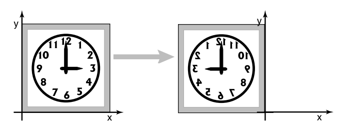
Shirley02

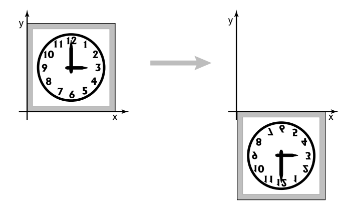
How About Reflection?
Non-Primitive

Reflection (even# flips): proper rotation



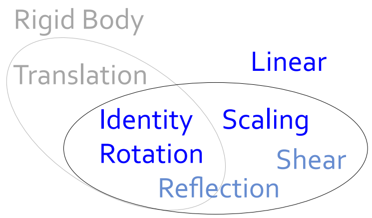
Reflection (odd# flips):
scaling with $s_x = -1$ or $s_y = -1$





Shirley02

Matrix Representation



Rotation about the origin:

$x' = x\cos\varphi - y\sin\varphi$

$y' = x\sin\varphi + y\cos\varphi$

Scaling:

$x' = s_x x$

$y' = s_y y$

More generally:

$x' = ax + by$

$y' = dx + ey$

A 2D linear transform can be represented by a 2×2 matrix:

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

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

To apply transformation to a point, multiply the column vector representing the point by the matrix

Durando8

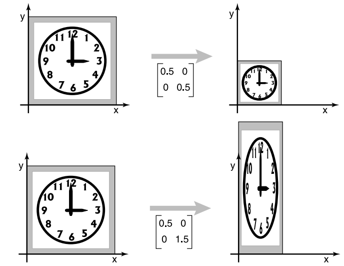
Linear Transform Matrices

Scaling: $x' = s_x x + 0y$

$y' = 0x + s_y y$

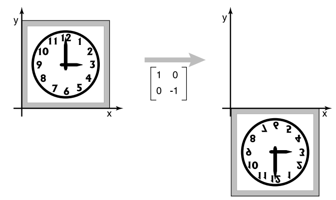
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{S}(s_x, s_y)\mathbf{p}$$

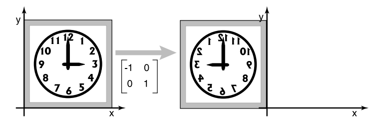


Reflection: flip vectors about an axis as if reflected in a mirror

Reflection about x



Reflection about y



Shirley02

Linear Transform Matrices

Shear: $x' = 1x + h_x y$

$y' = h_y x + 1y$

Shear transform in x:

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

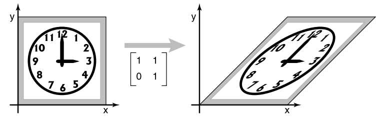
$$\mathbf{p}' = \mathbf{H}_x(h_x)\mathbf{p}$$

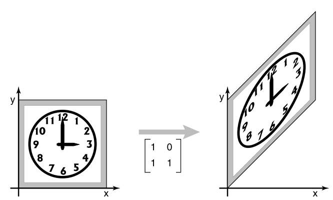
Shear transform in y:

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

$$\mathbf{p}' = \mathbf{H}_y(h_y)\mathbf{p}$$

Shear ::= an action or stress resulting from applied forces that causes, or tends to cause two contiguous parts of a body to slide relatively to each other





Shirley02

Linear Transform Matrices

Rotation about the origin:

$$\mathbf{p}\ \begin{cases} x = r\cos\theta \\ y = r\sin\theta \end{cases}$$

To rotate $\mathbf{p}$ by φ:

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

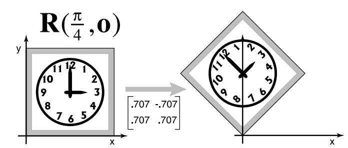
$$y' = r\sin(\theta+\varphi) = r\cos\theta\sin\varphi + r\sin\theta\cos\varphi$$

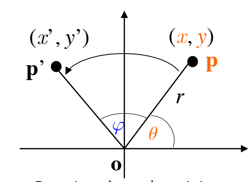
$$\mathbf{p}'\ \begin{cases} x' = x\cos\varphi - y\sin\varphi \\ y' = x\sin\varphi + y\cos\varphi \end{cases}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{R}(\varphi, \mathbf{o})\mathbf{p}$$

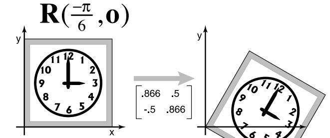
$\mathbf{R}(\frac{\pi}{4}, \mathbf{o})$





Rotation about the origin

$\mathbf{R}(\frac{-\pi}{6}, \mathbf{o})$

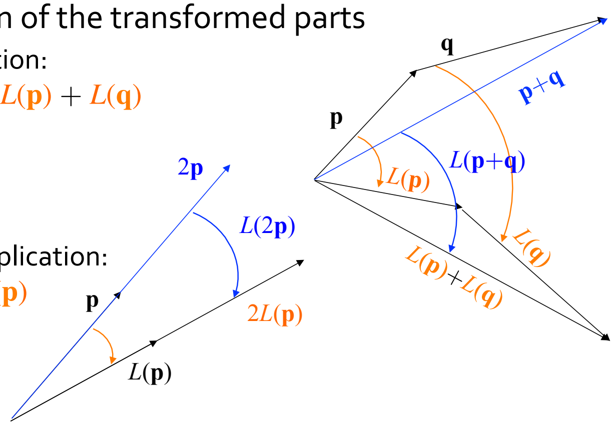


Shirley02

What Makes a Transformation Linear?

Preserves linear combination: transformation of a linear combination is the same as the linear combination of the transformed parts

- vector addition:
  $L(\mathbf{p}+\mathbf{q}) = L(\mathbf{p}) + L(\mathbf{q})$
- scalar multiplication:
  $L(a\mathbf{p}) = aL(\mathbf{p})$



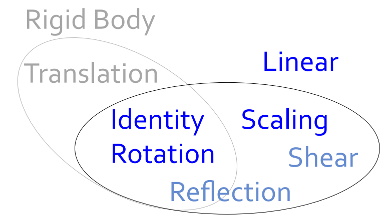
Durando8

What Makes a Transformation Linear?

Preserves linear combination:

- vector addition:
  $L(\mathbf{p}+\mathbf{q}) = L(\mathbf{p}) + L(\mathbf{q})$
- scalar multiplication:
  $L(a\mathbf{p}) = aL(\mathbf{p})$

$\Rightarrow$ a line remains a line:
$L(\alpha\,\mathbf{p} + (1-\alpha)(\mathbf{q}-\mathbf{p})) = \alpha L(\mathbf{p}) + (1-\alpha)(L(\mathbf{q}) - L(\mathbf{p}))$



To linear transform a polygon, it is sufficient to transform its vertices!

Why is translation not a linear transform?

$$T(a\mathbf{p}) = a\mathbf{p} + \mathbf{t} \neq aT(\mathbf{p}) = a(\mathbf{p}+\mathbf{t}) = a\mathbf{p} + a\mathbf{t}$$

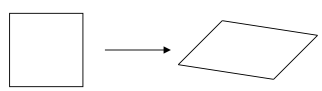
Durando8

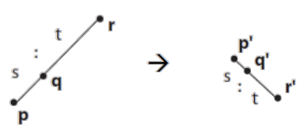
Affine Transforms

Preserves:

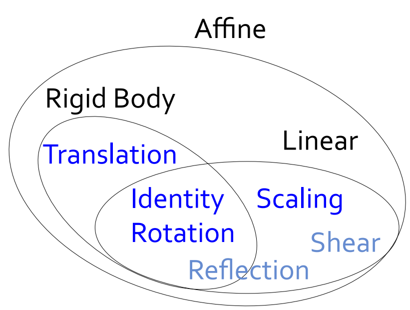
- (straight) lines
- parallel lines
- affine combination

$\Rightarrow$ preserve distance ratios (midpoints map to midpoints)





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



Curlesso8
Durando8

Matrix Representation of Affine Transforms

Translation:

$$x' = x + t_x$$
$$y' = y + t_y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{p} + \mathbf{t}$$

In general, an affine transform can be represented as:

$$x' = ax + by + t_x$$
$$y' = dx + ey + t_y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ d & e \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{M}\mathbf{p} + \mathbf{t} = A(\mathbf{p})$$

linear transforms plus translation

Affine Transforms

Is translation an affine transform?

Does translation preserve affine combination?

Let $\mathbf{p} = \sum_{i=0}^{n} a_i \mathbf{p}_i$ and $\sum_{i=0}^{n} a_i = 1$

Affine transform:

$A(\mathbf{p}) = \mathbf{Mp} + \mathbf{t}$

$$A\left(\sum_{i=0}^{n} a_i \mathbf{p}_i\right) \stackrel{?}{=} \sum_{i=0}^{n} a_i A(\mathbf{p}_i)$$

$$= \sum_{i=0}^{n} a_i (\mathbf{Mp}_i + \mathbf{t})$$

$$= \sum_{i=0}^{n} a_i \mathbf{Mp}_i + \sum_{i=0}^{n} a_i \mathbf{t}$$

$$= \mathbf{M}\left(\sum_{i=0}^{n} a_i \mathbf{p}_i\right) + \mathbf{t}$$

$$= \mathbf{Mp} + \mathbf{t} = A(\mathbf{p})$$

by definition of affine combination

by definition of linear transform

?

⇒ barycentric coordinates are preserved

⇒ to affine transform a polygon, it is still sufficient to transform only its vertices

Composition of Transformations: Linear Transforms

Linear transforms are associative $(\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C})$

Linear transforms are closed under composition

⇒ composition of linear transforms can be represented as multiplication of the transformation matrices, e.g., rotate then scale:

$$\mathbf{p}' = \mathbf{S}(s_x, s_y)\mathbf{R}(\varphi, \mathbf{o})\mathbf{p}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} \cos\varphi & -\sin\varphi \\ \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x \cos\varphi & -s_x \sin\varphi \\ s_y \sin\varphi & s_y \cos\varphi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Why is being closed under composition important?

- why do we want to combine/compose transforms?

Composition of Linear Transforms

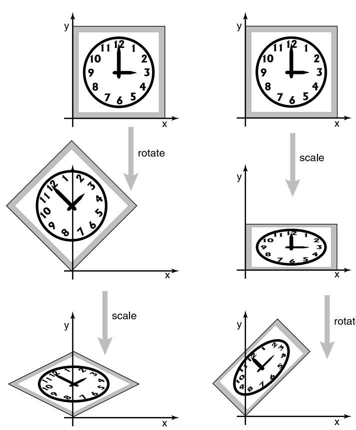
Matrix multiplication applied right to left

Matrix multiplications are generally not commutative:

though the following are:

SS, **RR** (2D only),

SR (isotropic **S** only)



Shirley02

Composition of Affine Transforms?

In general, an affine transform can be represented as:

$$x' = ax + by + t_x$$

$$y' = dx + ey + t_y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ d & e \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\mathbf{p}' = \mathbf{Mp} + \mathbf{t} = A(\mathbf{p})$$

Not in $\mathbf{p}' = \mathbf{Mp}$ form!

⇒ combining transformations is no longer a simple matrix multiplication!

We would like to use only matrices to represent all affine transforms:

$\mathbf{p}' \stackrel{?}{=} \mathbf{Mp}$