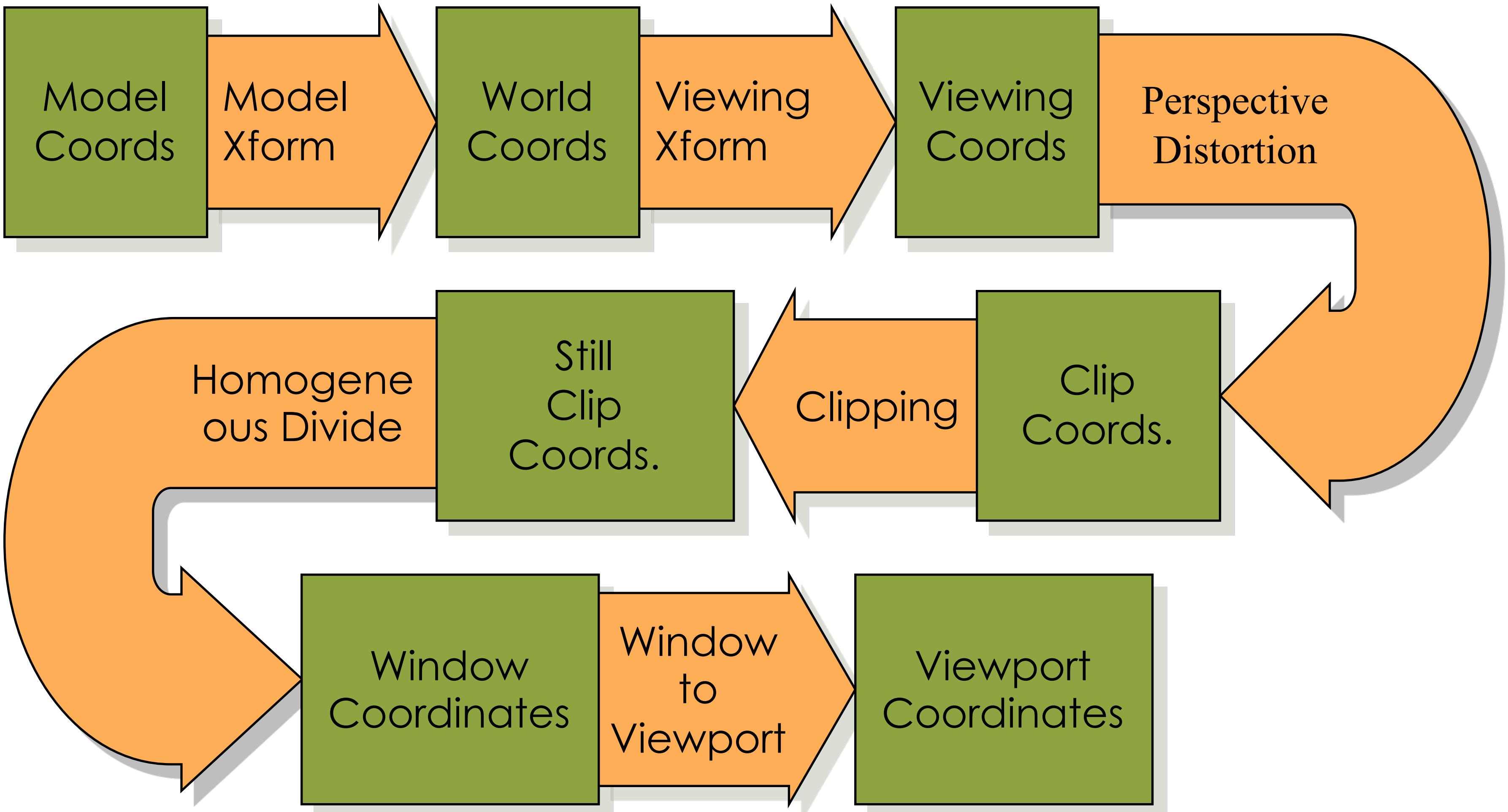
CS 418: Interactive Computer Graphics

Linear and Affine Transformations

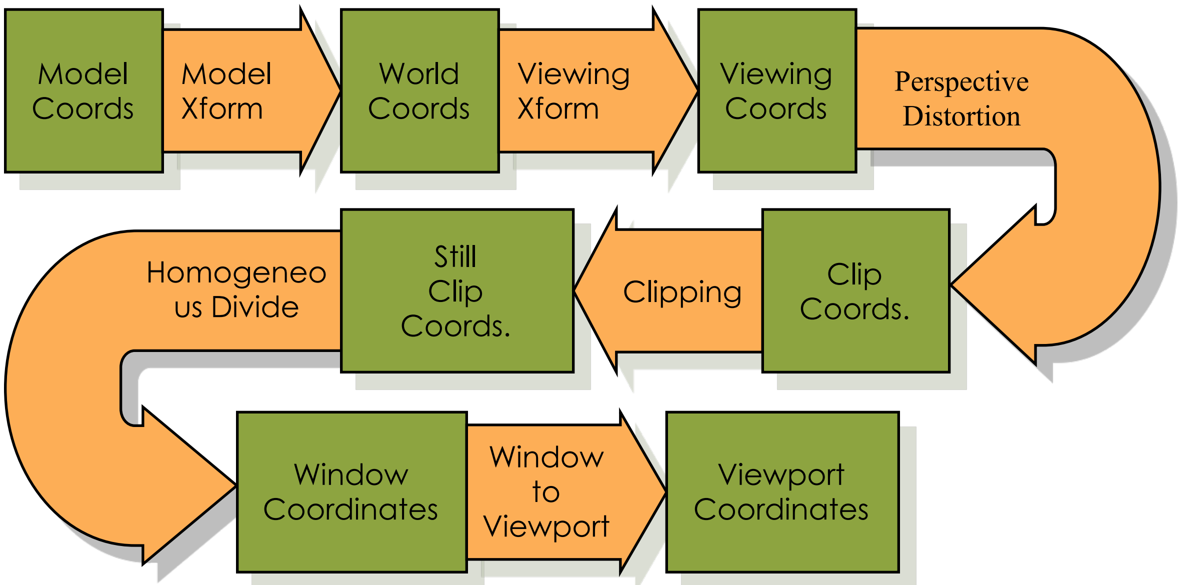
Dr. Eric Shaffer

Slides adapted from
Professor John Hart's CS 418 Slides

Rendering Pipeline: Coordinate Transformations



Rendering Pipeline: Coordinate Transformations



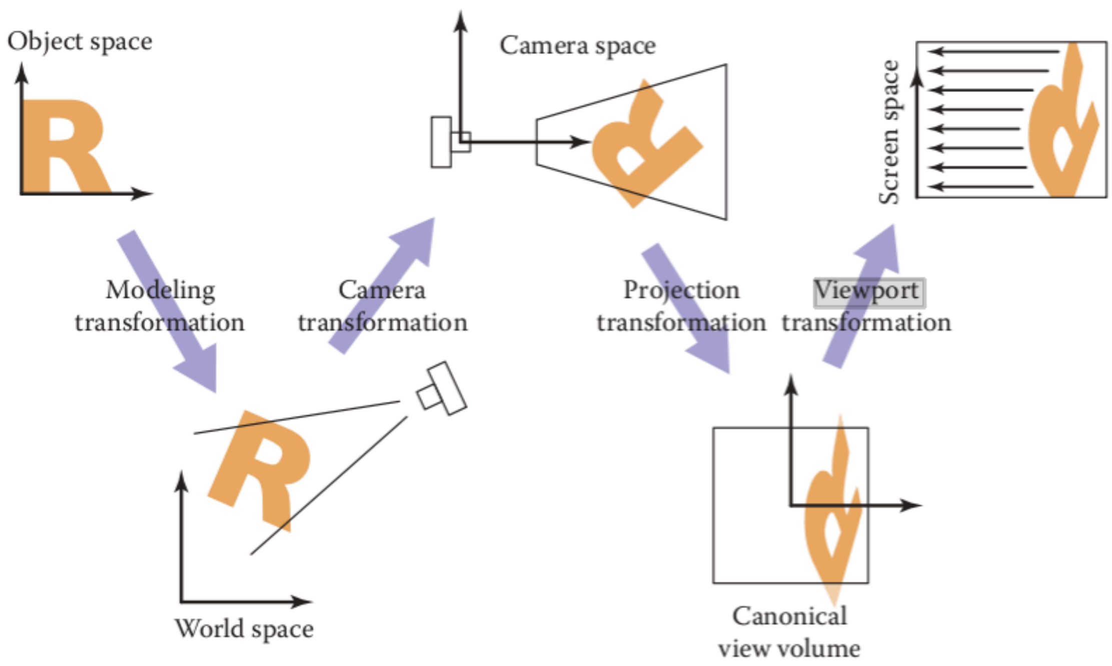
Not everyone uses the same terminology… in Unity, viewport coordinates are in the range [-1,1]

we'll use what you see listed here

Window Coordinates: 2D and in range [-1,1]
Viewport Coordinates: pixel coordinates (i.e. screen space coordinates)

What stages that you see here happen in the vertex shader?
When do we have NDC coordinates?

Another way of looking at it…

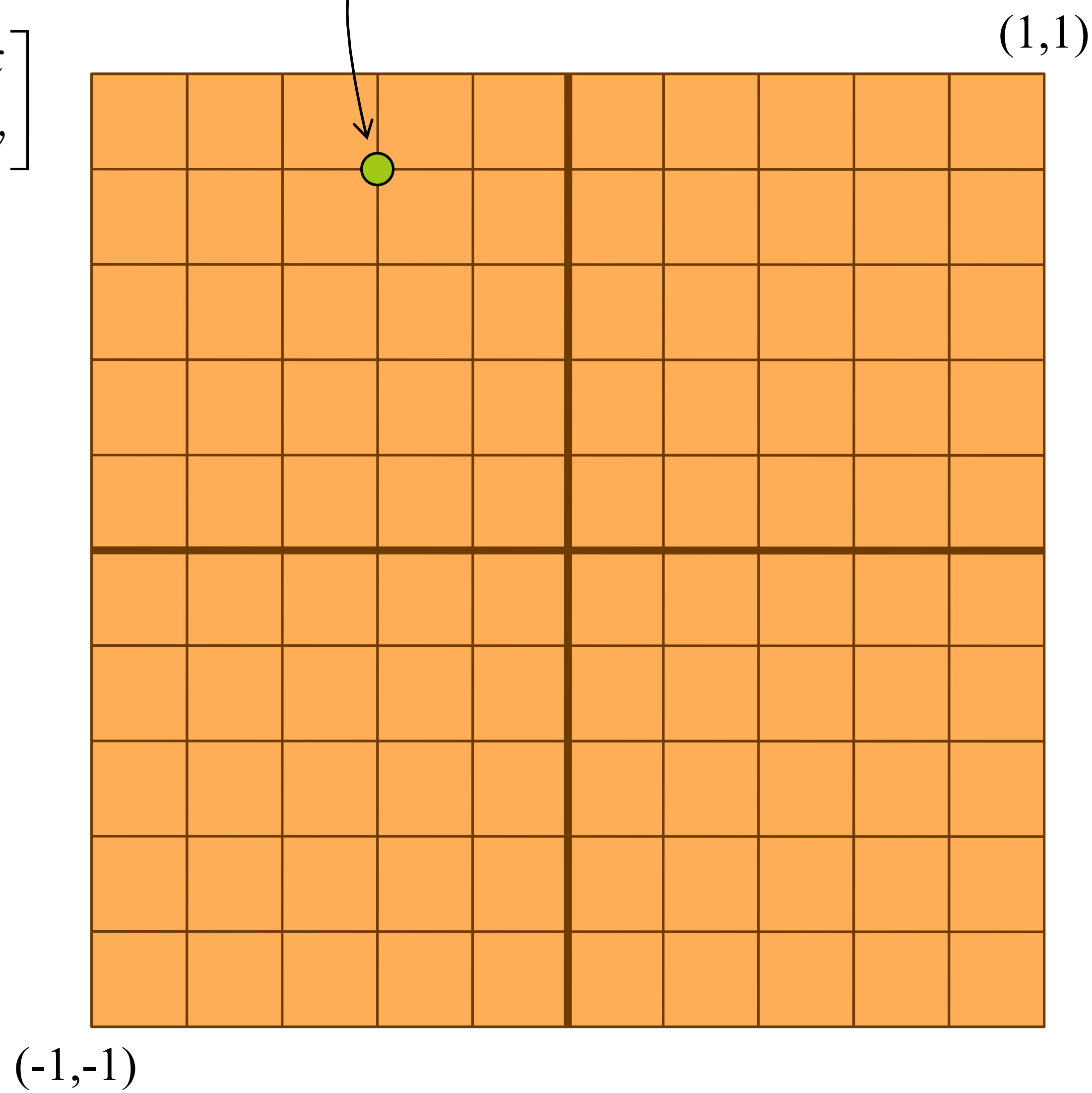


From Fundamentals of Computer Graphics by Marschner and Shirley

2-D Points

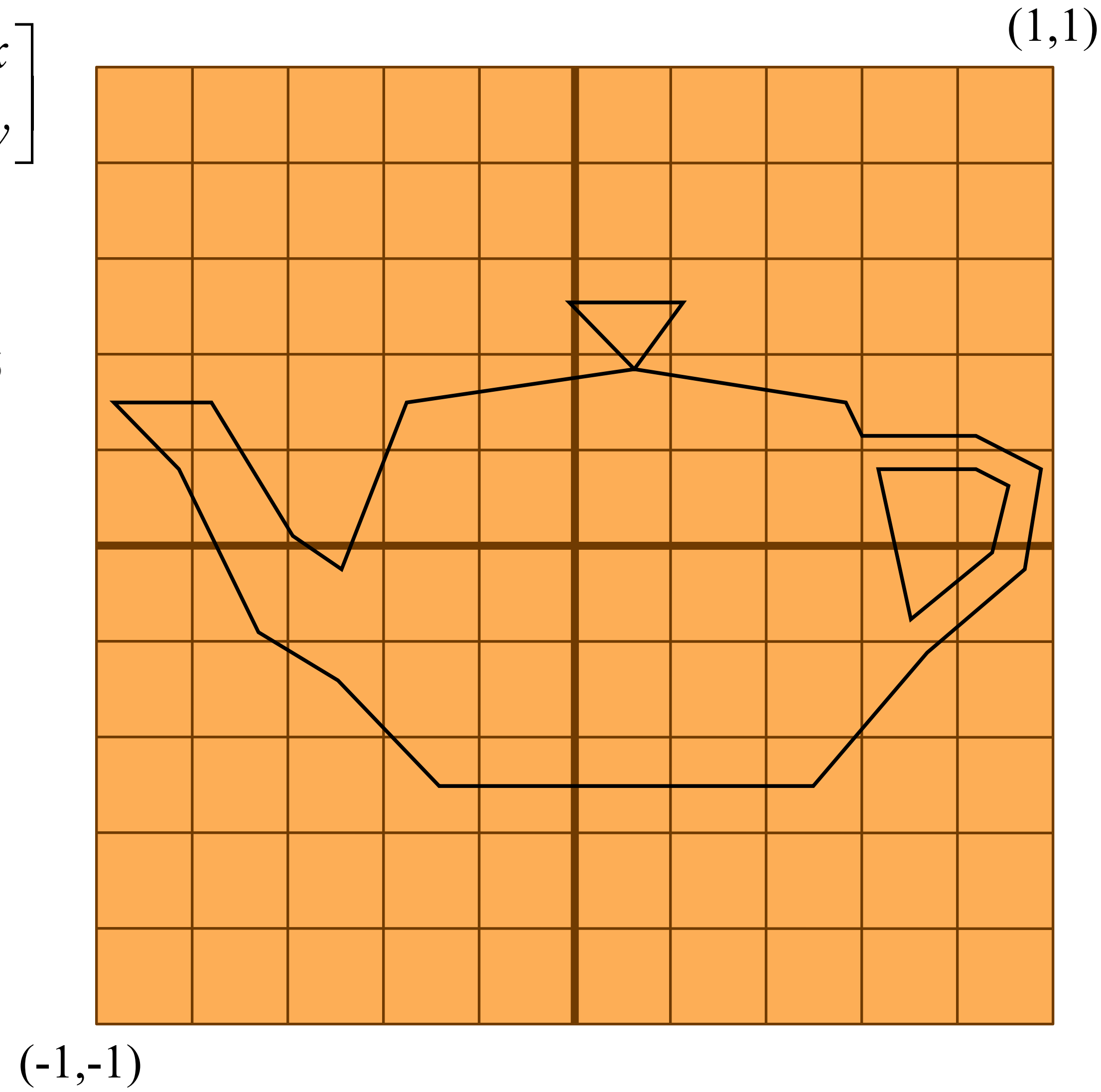
- Represents points and vertices as column vectors: $\begin{bmatrix} x \\ y \end{bmatrix}$

$\begin{bmatrix} -.4 \\ .8 \end{bmatrix}$



2-D Points

- Represents points and vertices as column vectors: $\begin{bmatrix} x \\ y \end{bmatrix}$
- Transform polygonal object by transforming its vertices

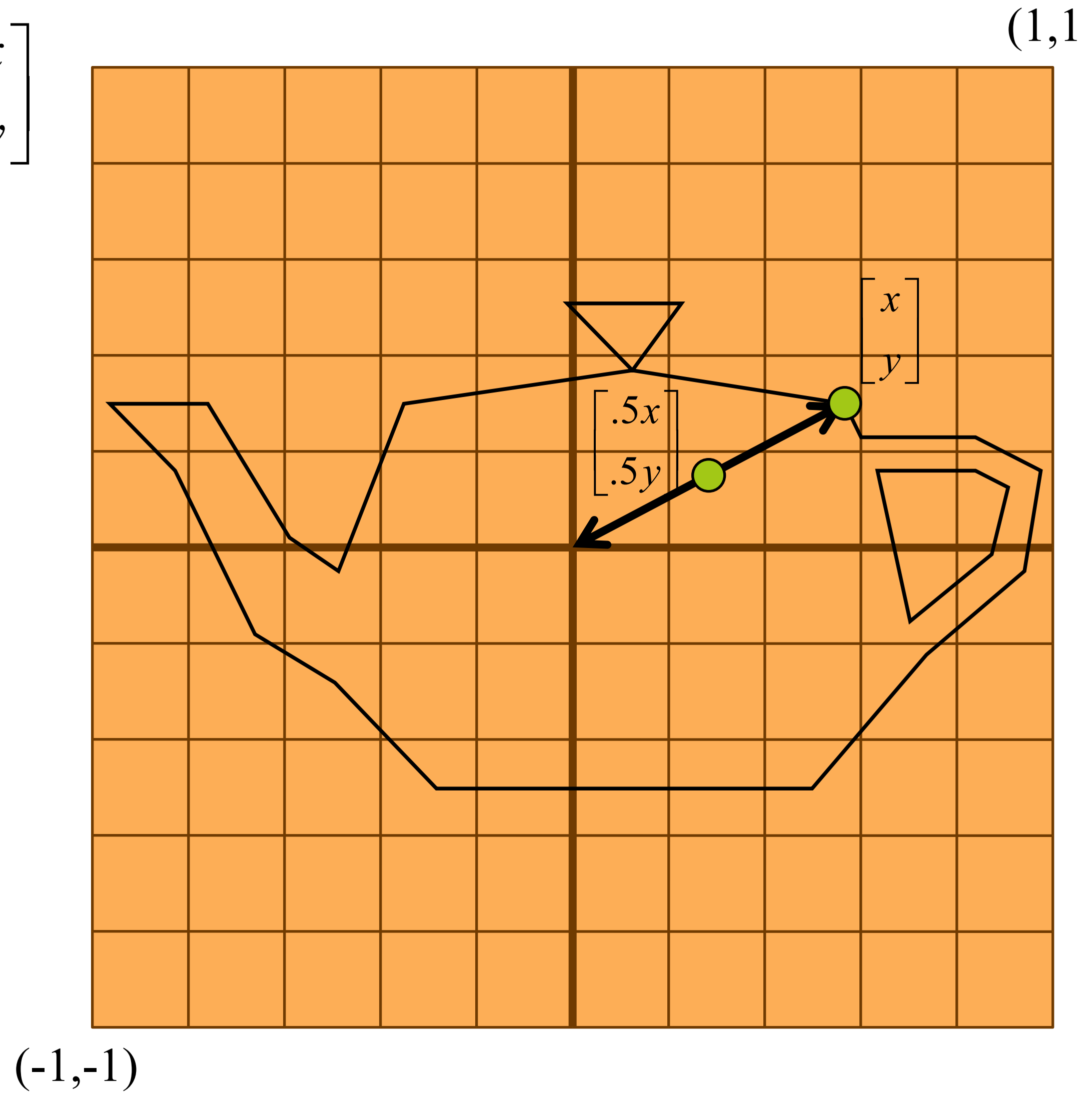


2-D Points

- Represents points and vertices as column vectors: $\begin{bmatrix} x \\ y \end{bmatrix}$
- Transform polygonal object by transforming its vertices
- Scale by matrix multiplication

$$\begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ ay \end{bmatrix}$$

$$\begin{bmatrix} .5 & 0 \\ 0 & .5 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$



2-D Points

$$\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -.4 \end{bmatrix}$$

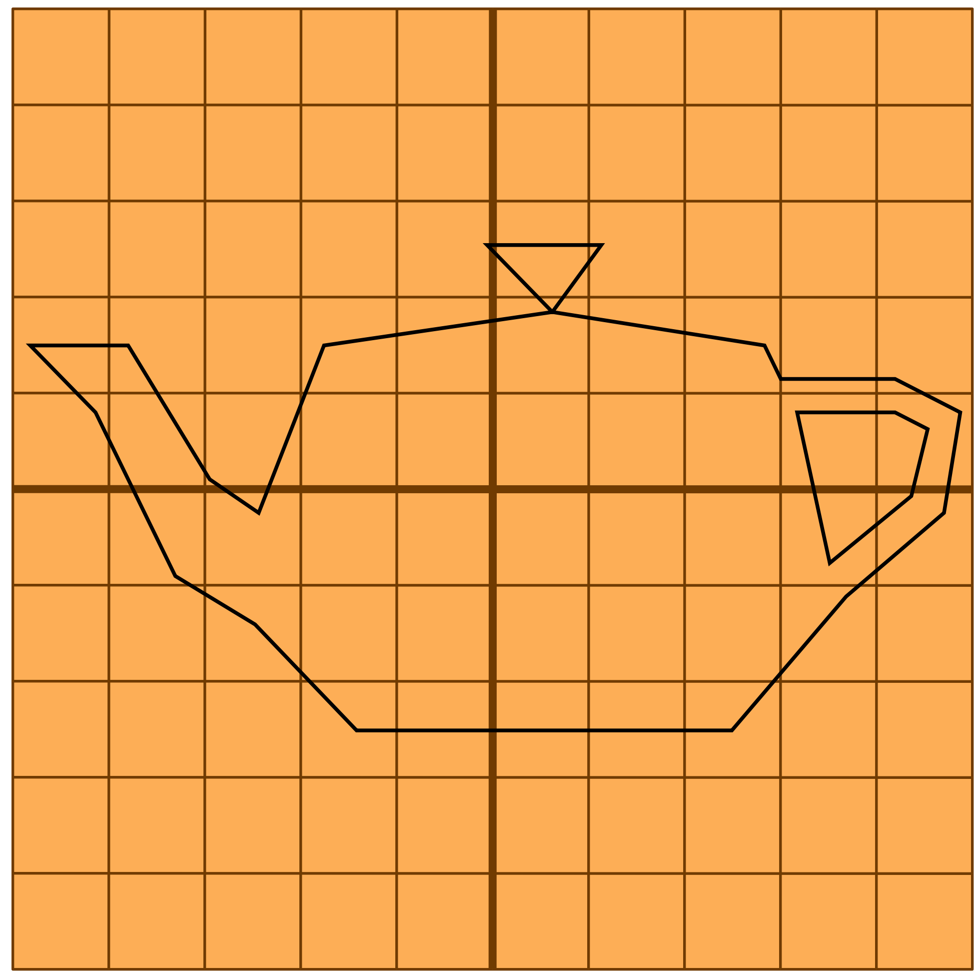
- Represents points and vertices as column vectors: $\begin{bmatrix} x \\ y \end{bmatrix}$
- Transform polygonal object by transforming its vertices
- Scale by matrix multiplication

$$\begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ ay \end{bmatrix}$$

- Translation via vector sum

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

(1,1)



(-1,-1)

2-D Points

$$\begin{bmatrix} .5 & 0 \\ 0 & .5 \end{bmatrix}\left(\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -.4 \end{bmatrix}\right)$$

$$\begin{bmatrix} .5 & 0 \\ 0 & .5 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -.4 \end{bmatrix}$$

- Represent points and vertices as column vectors: $\begin{bmatrix} x \\ y \end{bmatrix}$
- Transform polygonal object by transforming its vertices
- Scale by matrix multiplication

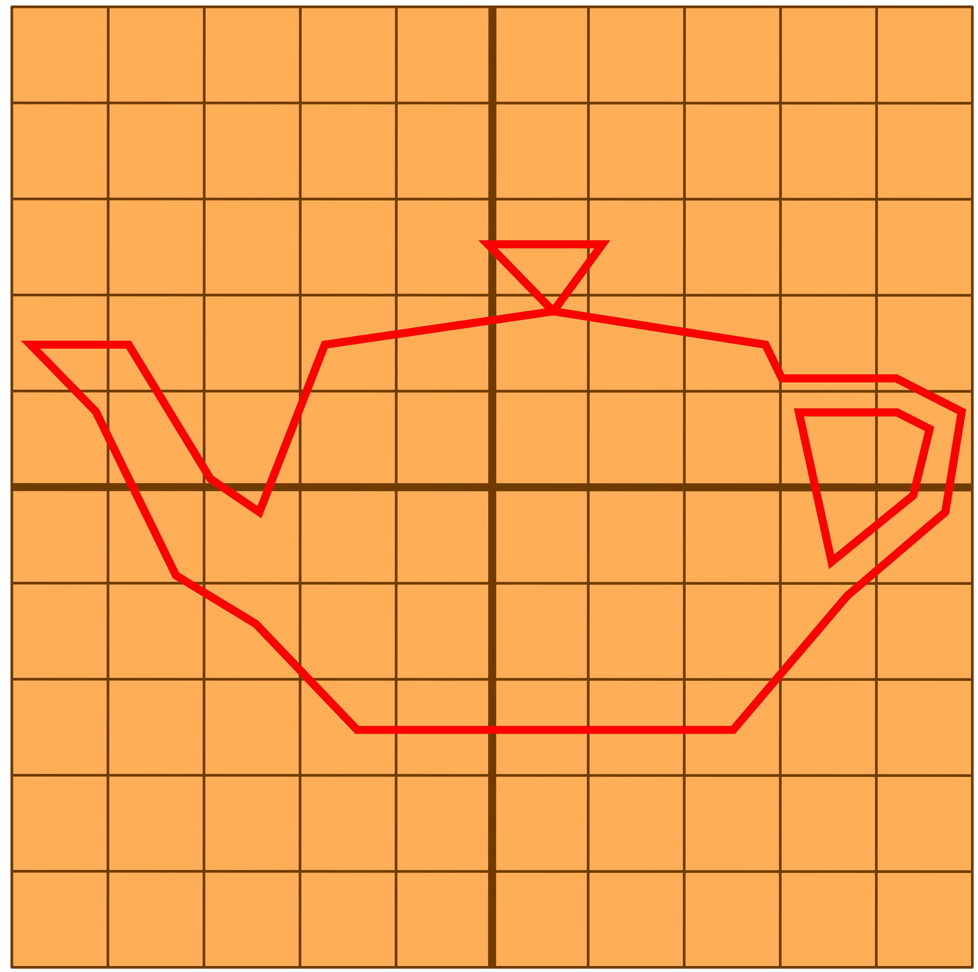
$$\begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} ax \\ ay \end{bmatrix}$$

- Translation via vector sum

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

- Order is important
  - Translate then scale
  - Scale then translate

(1,1)



(-1,-1)

Homogeneous Coordinates

$$\begin{bmatrix} .5 & 0 \\ 0 & .5 \end{bmatrix}\left(\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -.4 \end{bmatrix}\right)$$

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

- Translation by vector sum is cumbersome
- Add a extra coordinate
  - Called the homogeneous coordinate
  - For now, set to one
- Translation now expressed as a matrix
- Now we can compose scales and translations into a single matrix by matrix multiplication

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

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -.4 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} .5 & 0 & 0 \\ 0 & .5 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} .5 & 0 & 0 \\ 0 & .5 & -.4 \\ 0 & 0 & 1 \end{bmatrix}$$

Order Dependence

$$\begin{bmatrix} .5 & 0 \\ 0 & .5 \end{bmatrix}\left(\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -.4 \end{bmatrix}\right)$$

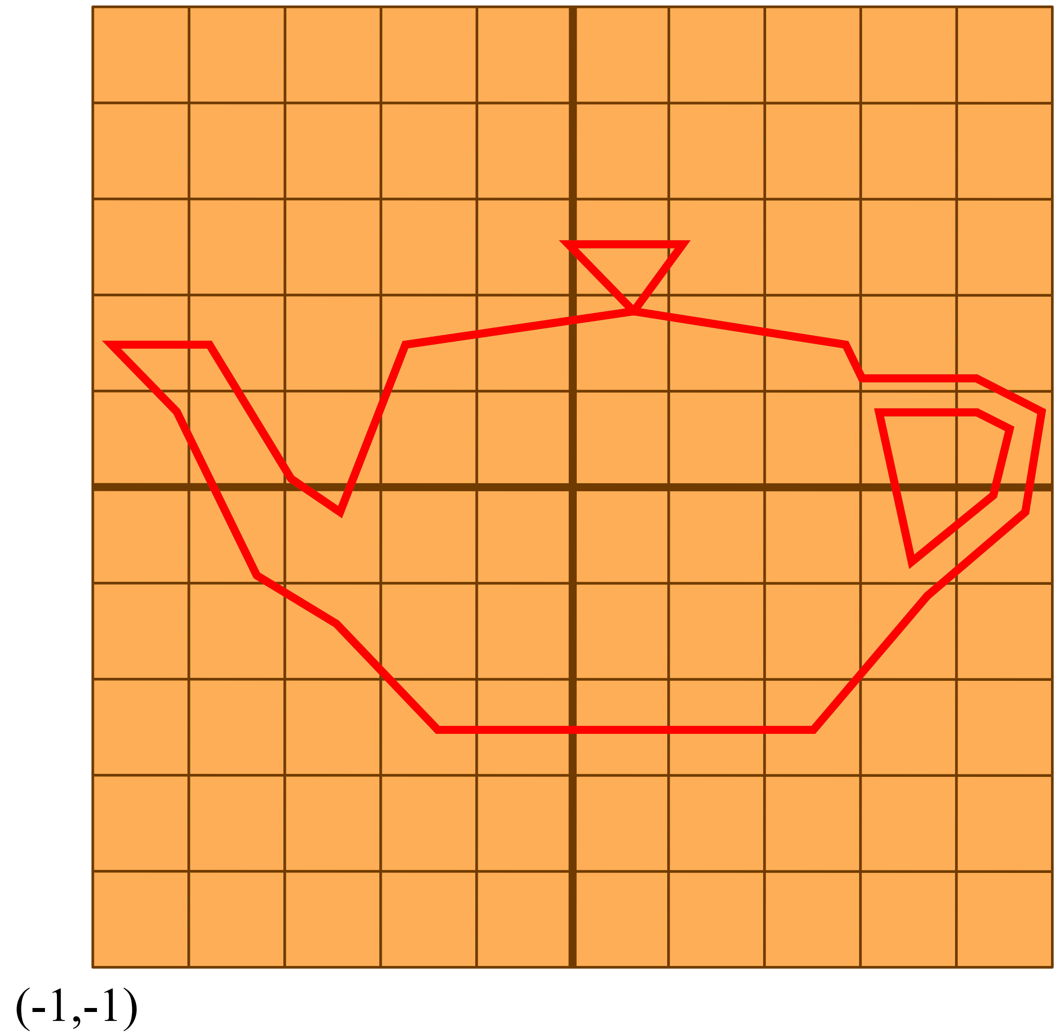
$$\begin{bmatrix} .5 & 0 & 0 \\ 0 & .5 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -.4 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} .5 & 0 & 0 \\ 0 & .5 & -.2 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} .5 & 0 \\ 0 & .5 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} 0 \\ -.4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -.4 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} .5 & 0 & 0 \\ 0 & .5 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} .5 & 0 & 0 \\ 0 & .5 & -.4 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



Window-to-Viewport

- First translate lower-left corner to (0,0)

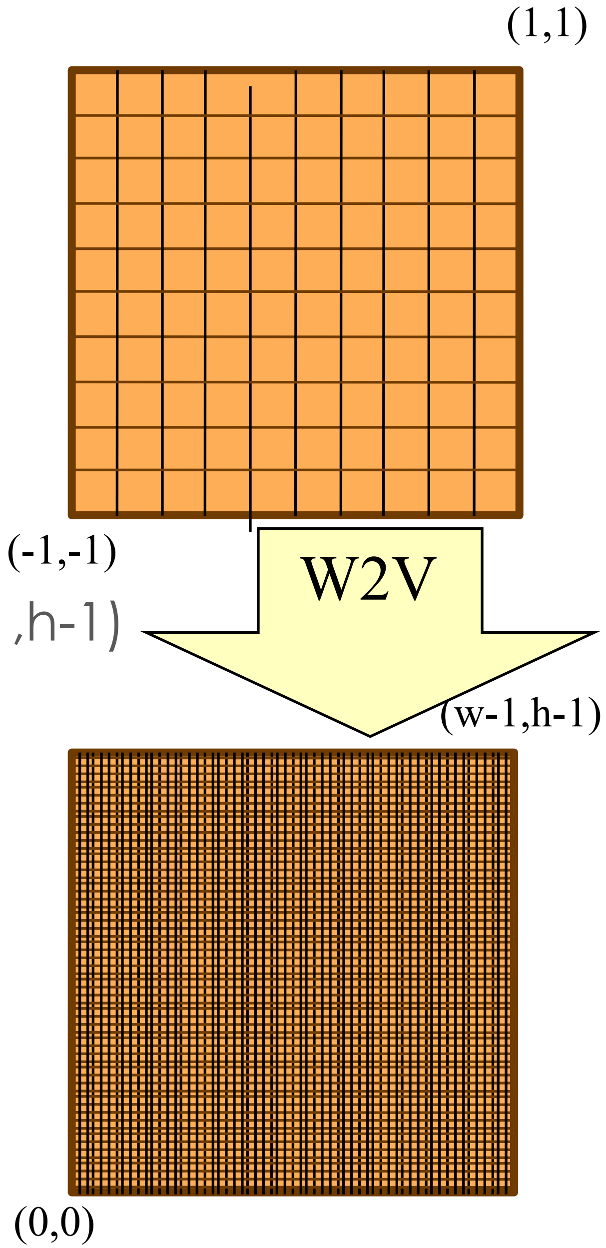
$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Then scale upper-right corner from (2,2) to (w-1,h-1)

$$\begin{bmatrix} \frac{w-1}{2} & 0 & 0 \\ 0 & \frac{h-1}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

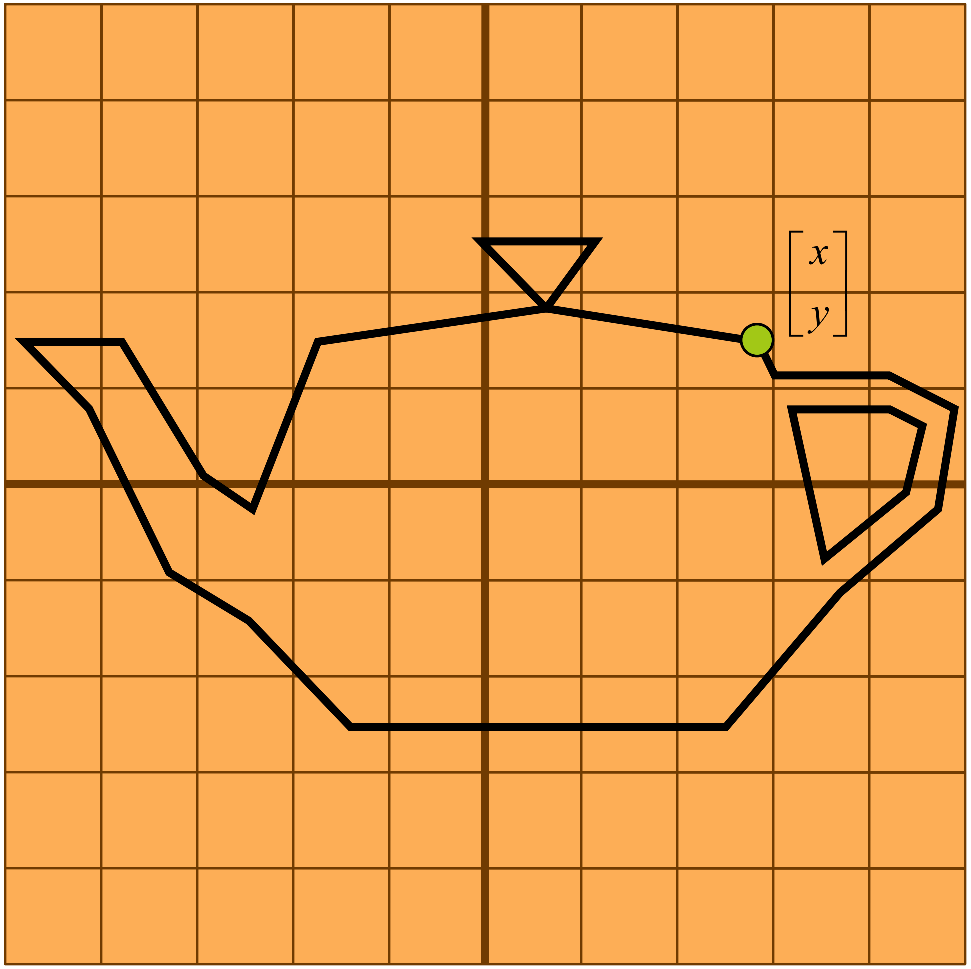
- To get

$$\begin{bmatrix} \frac{w-1}{2} & 0 & \frac{w-1}{2} \\ 0 & \frac{h-1}{2} & \frac{h-1}{2} \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



2-D Rotation

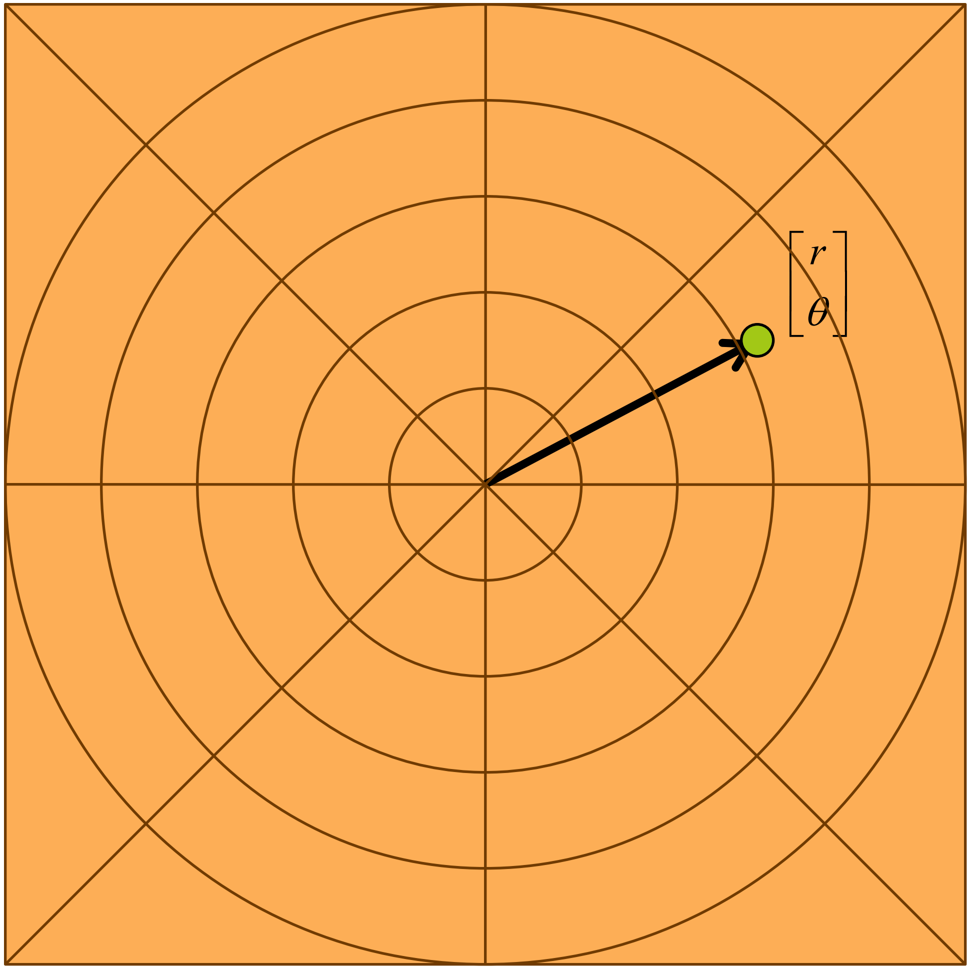
- Pick a point (x,y)



2-D Rotation

- Pick a point (x,y)
- Assume polar coords

  $x = r\cos\theta,\ y = r\sin\theta$

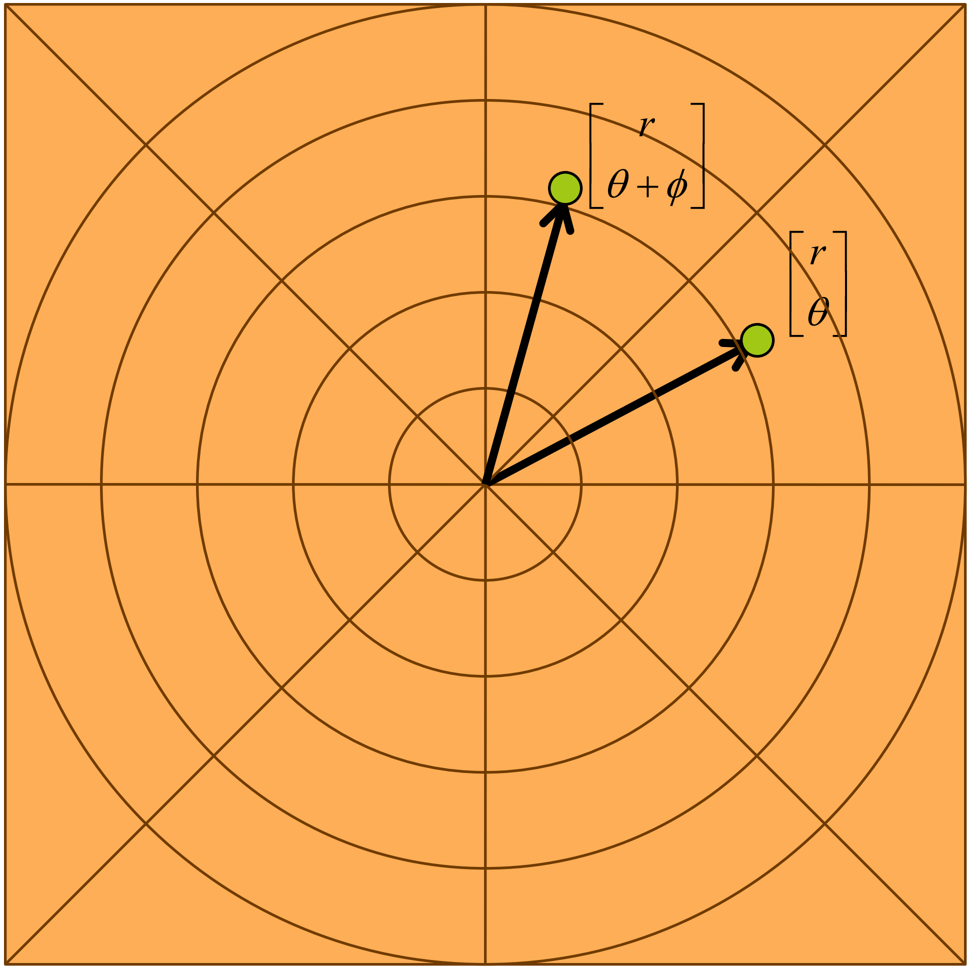


2-D Rotation

- Pick a point (x,y)
- Assume polar coords

  $x = r \cos \theta$, $y = r \sin \theta$
- Rotate about origin by ϕ

  $x' = r \cos \theta + \phi$, $y' = r \sin \theta + \phi$



2-D Rotation

- Pick a point (x,y)
- Assume polar coords

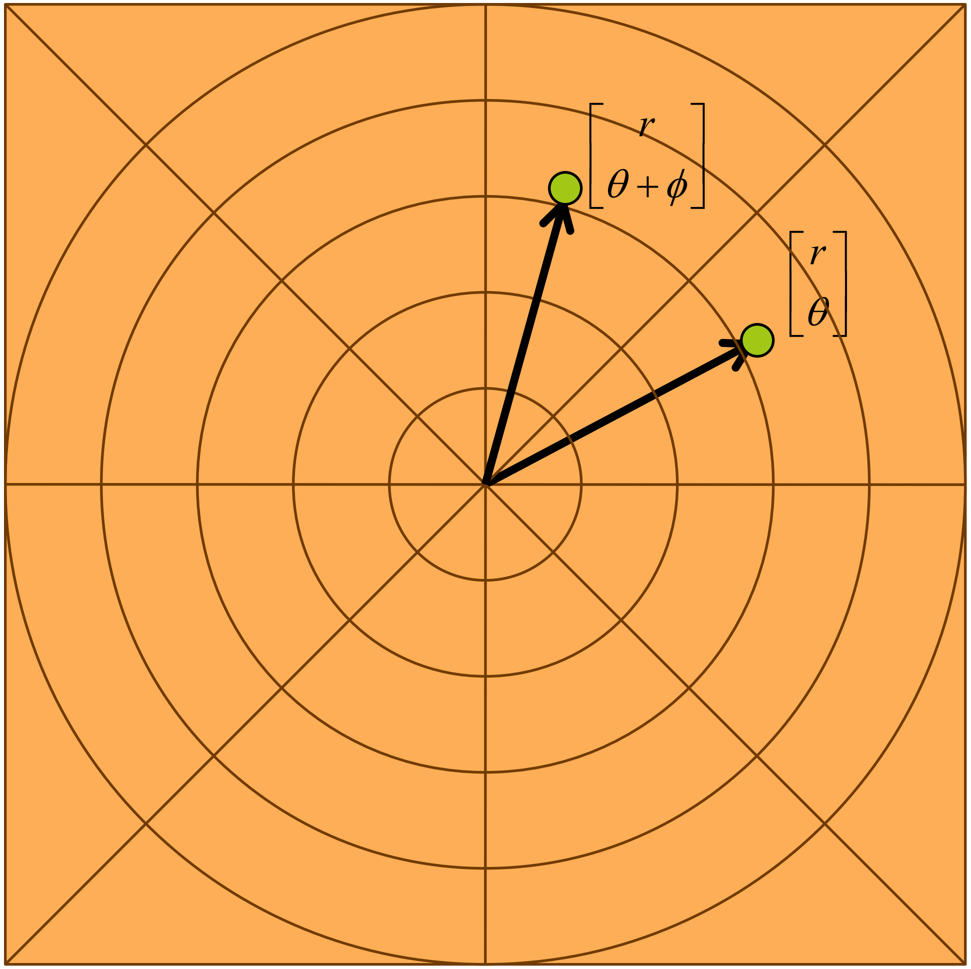
  $x = r\cos\theta,\ y = r\sin\theta$

- Rotate about origin by ϕ

  $x' = r\cos\theta+\phi,\ y' = r\sin\theta+\phi$

- Recall trig. identities

  $x' = r\,(\cos\theta\cos\phi - \sin\theta\sin\phi)$
  $y' = r\,(\sin\theta\cos\phi + \cos\theta\sin\phi)$



2-D Rotation

- Pick a point (x,y)
- Assume polar coords

  $x = r \cos \theta,\ y = r \sin \theta$

- Rotate about origin by ϕ

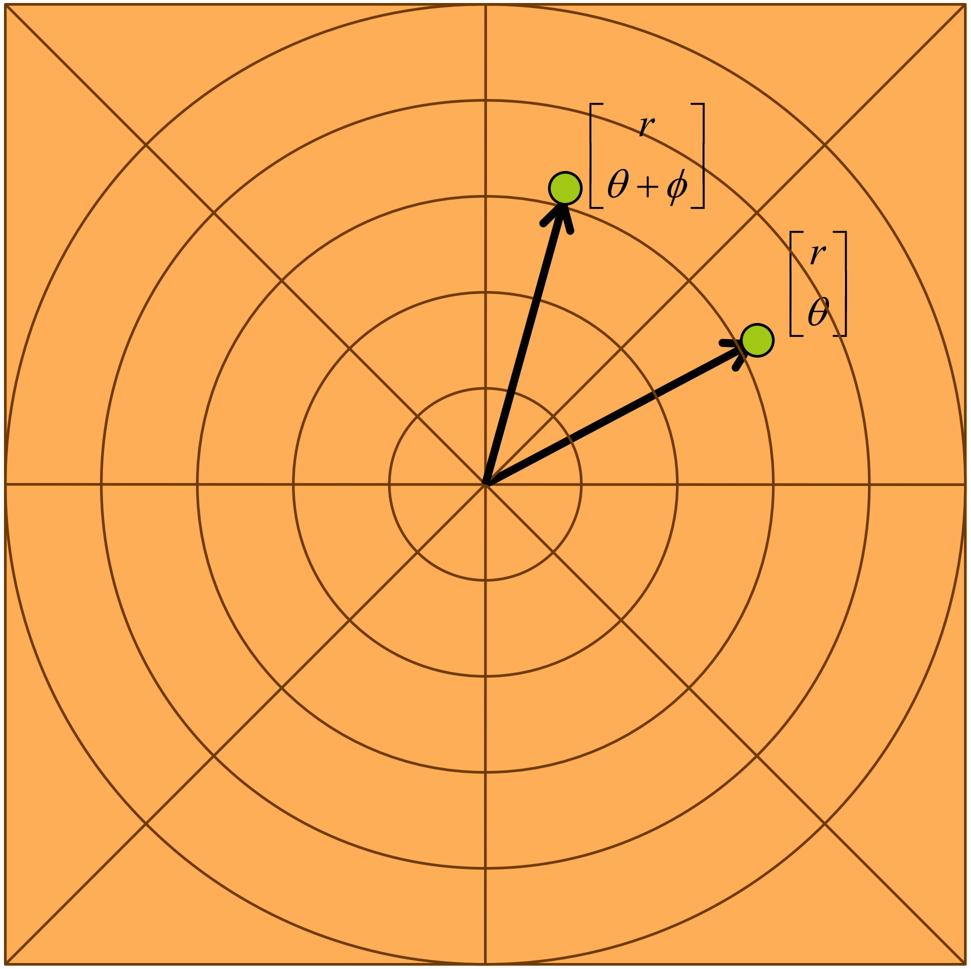
  $x' = r \cos \theta + \phi,\ y' = r \sin \theta + \phi$

- Recall trig. identities

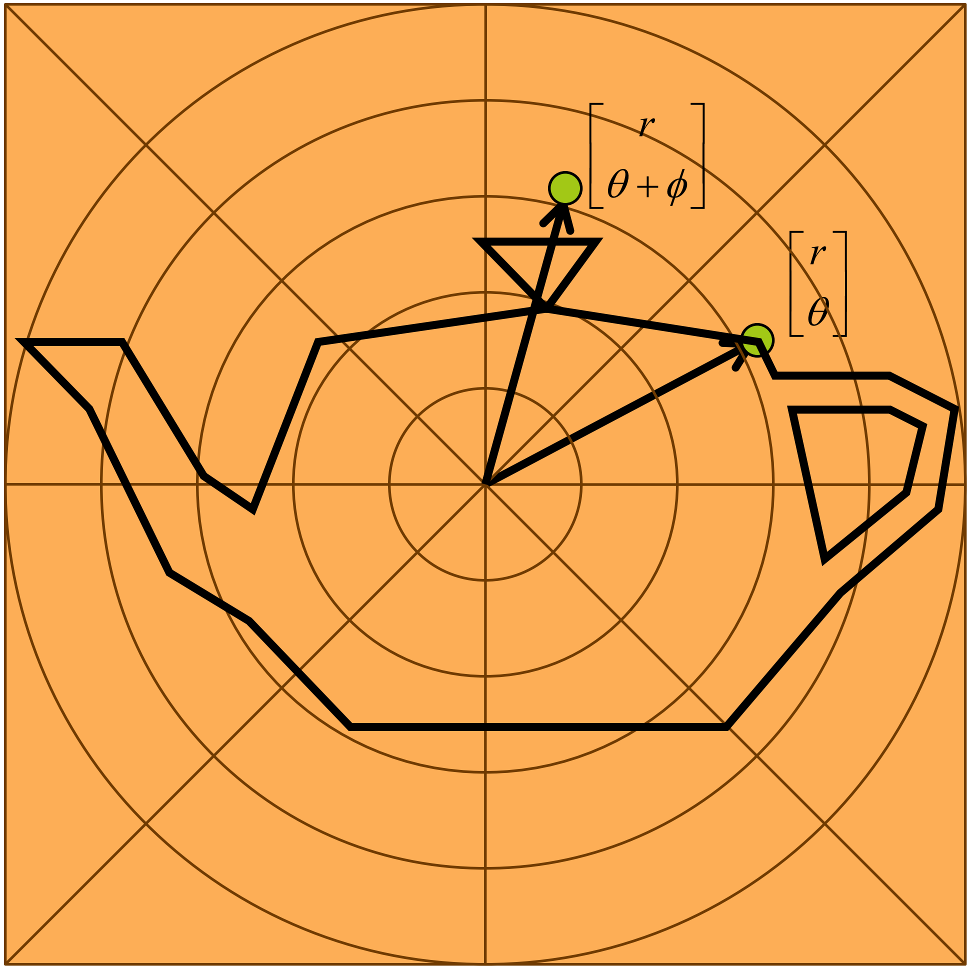
  $x' = r (\cos\theta \cos \phi - \sin\theta \sin\phi)$
  $y' = r (\sin\theta \cos \phi + \cos\theta \sin\phi)$

- Rearrange terms

  $x' = \cos \phi\, (r \cos\theta) - \sin\phi\, (r \sin\theta)$
  $y' = (r \cos\theta) \sin\phi + (r \sin\theta) \cos \phi$



2-D Rotation



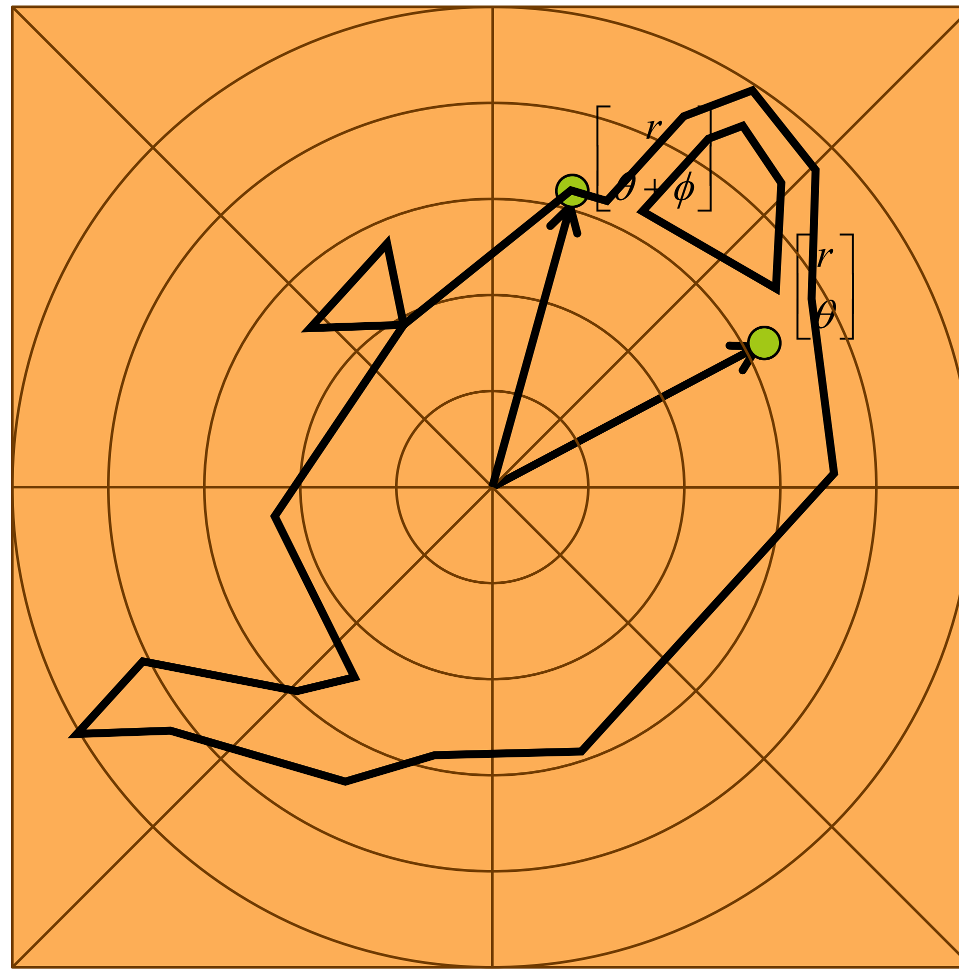
$$x' = \cos\phi\,(r\cos\theta) - \sin\phi\,(r\sin\theta)$$
$$y' = (r\cos\theta)\sin\phi + (r\sin\theta)\cos\phi$$

2-D Rotation

$$x' = \cos\phi\,\overset{x}{(r\cos\theta)} - \sin\phi\,\overset{y}{(r\sin\theta)}$$

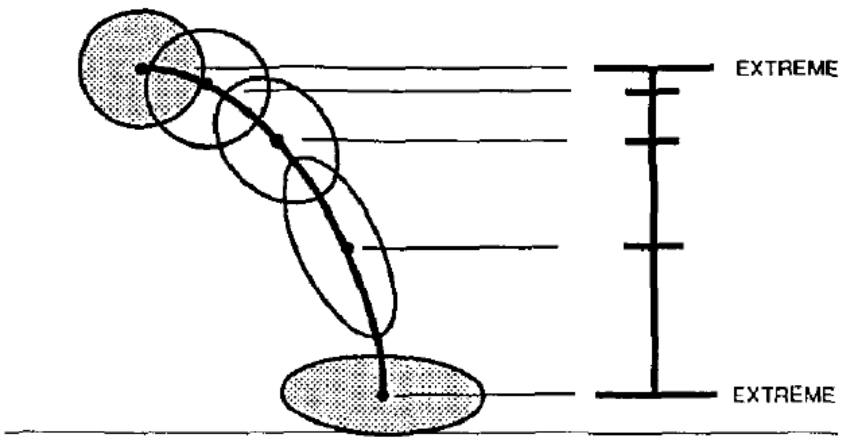
$$y' = \underset{x}{(r\cos\theta)}\sin\phi + \underset{y}{(r\sin\theta)}\cos\phi$$

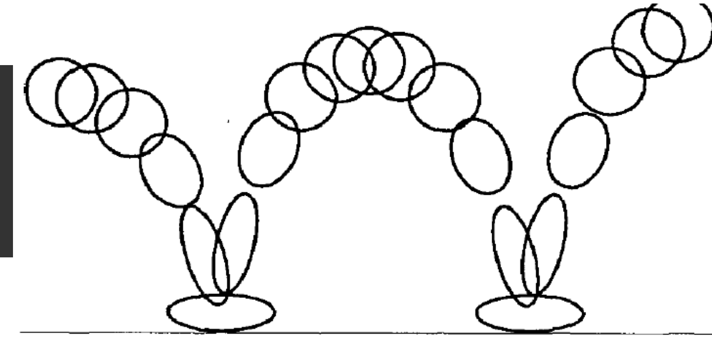
$$\begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

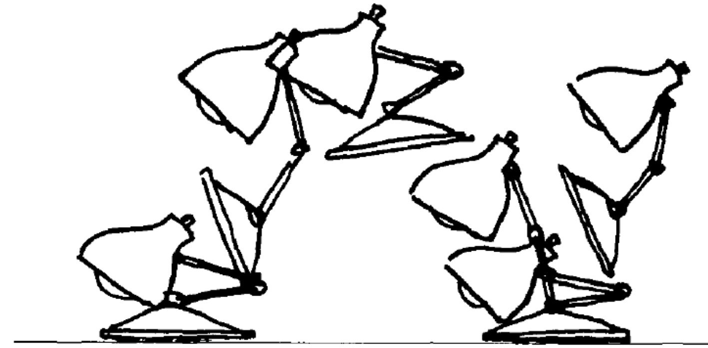


Squash & Stretch

From: *John Lasseter: "Principles of Traditional Animation Applied to 3-D Computer Animation" Proc. SIGGRAPH 87*







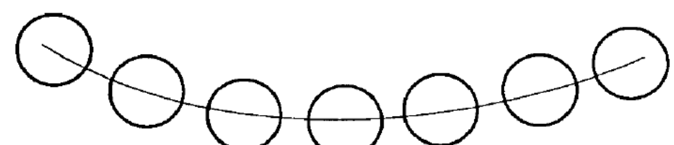


FIGURE 4b. Strobing occurs in a faster action when the object's positions do not overlap and the eye perceives seperate images.

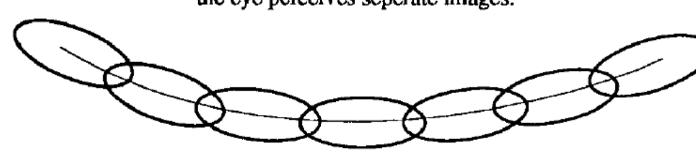
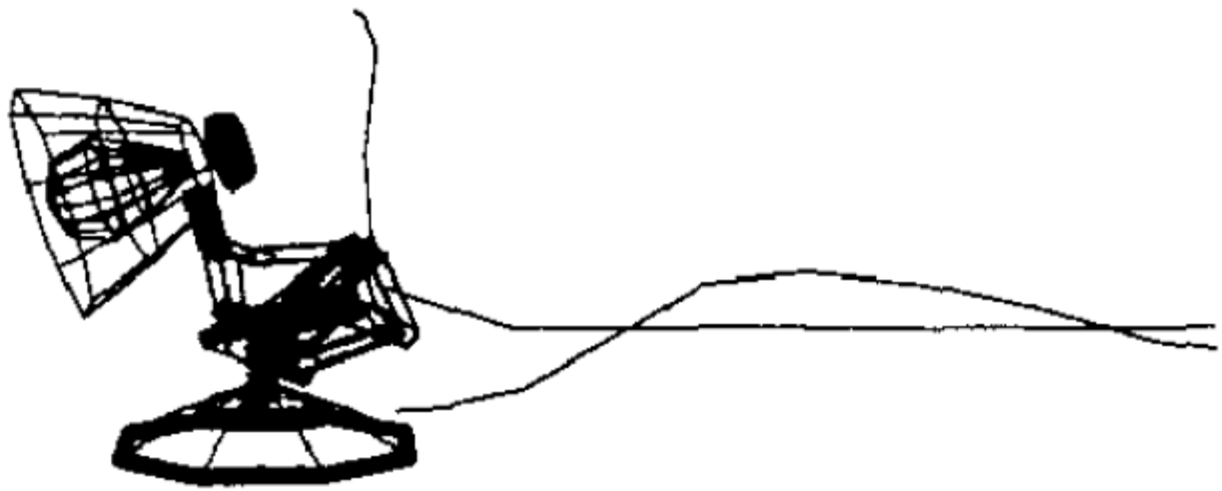
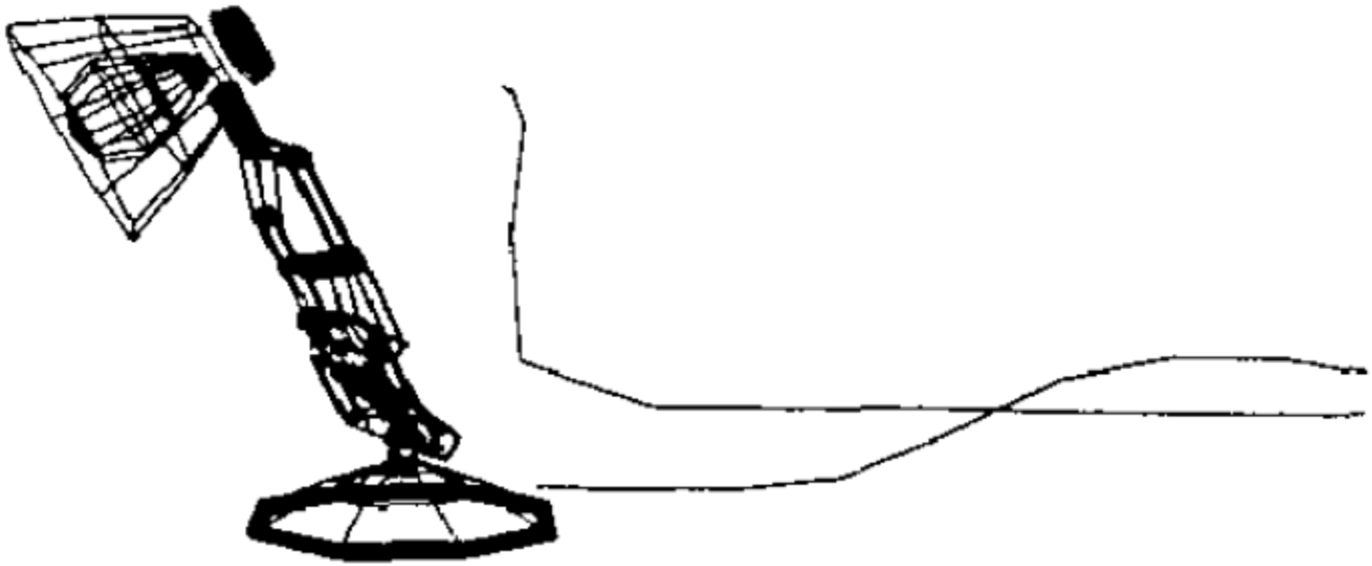


FIGURE 4c. Stretching the object so that it's positions overlap again will relieve the strobing effect.

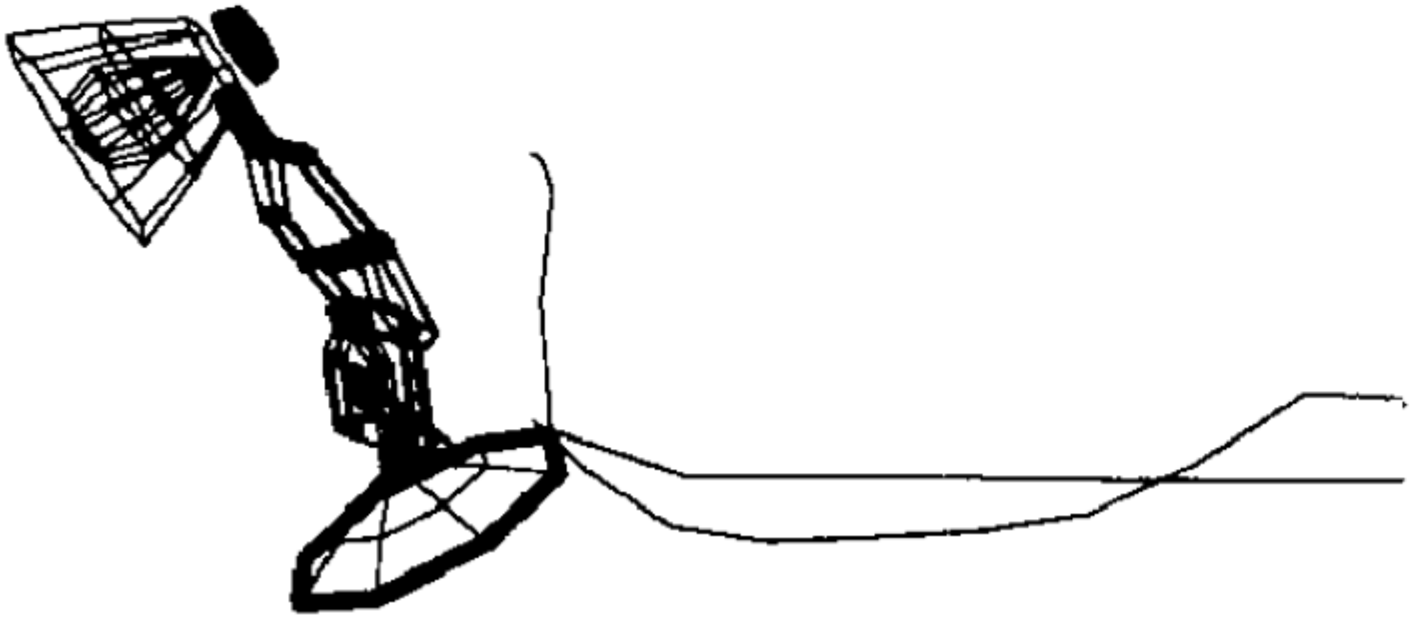
Luxo Jr.



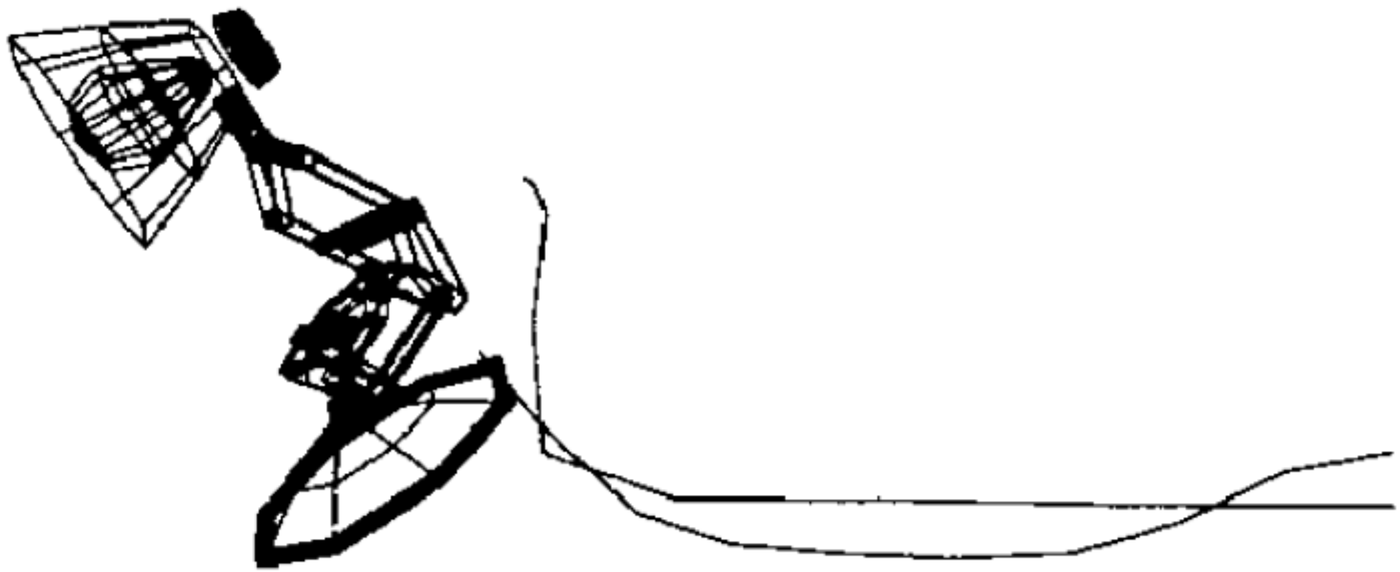
Luxo Jr.



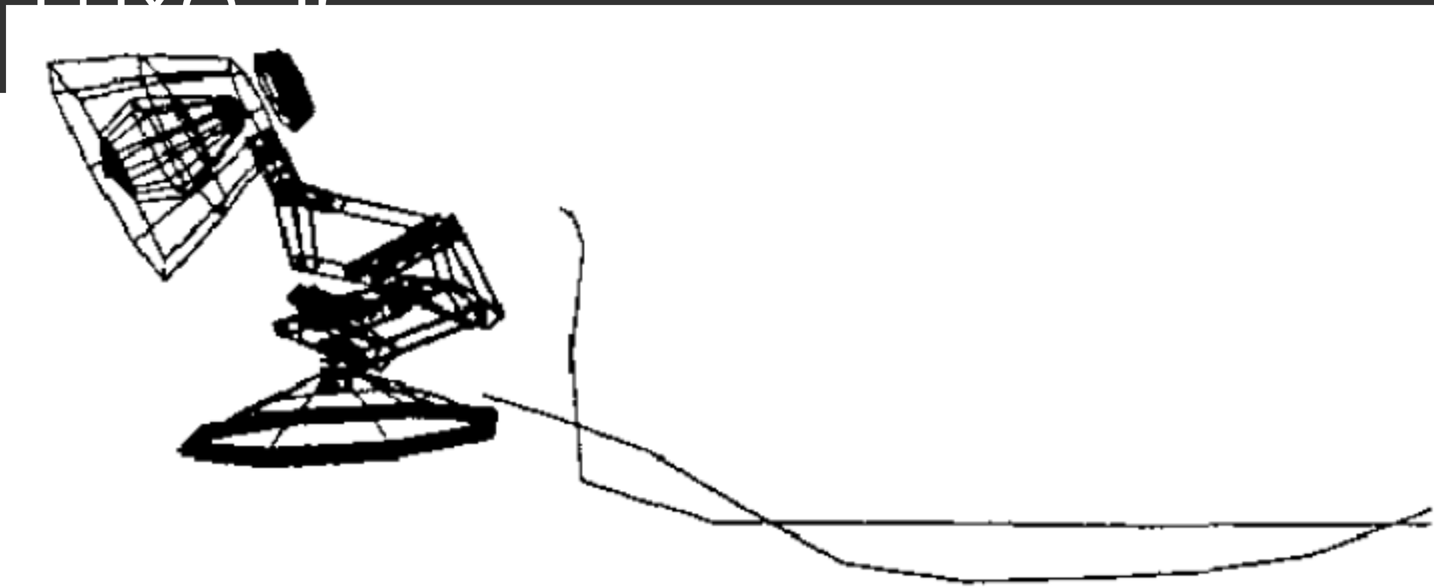
Luxo Jr.



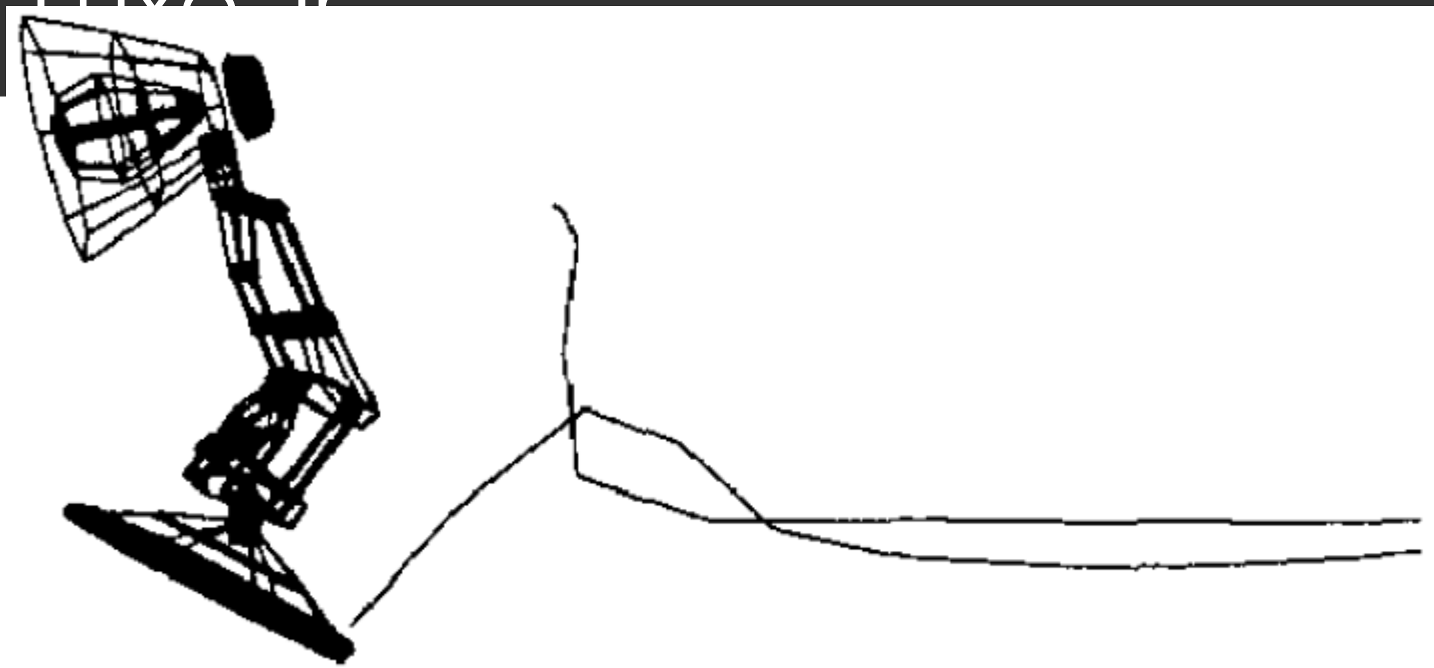
Luxo Jr



Luxo Jr

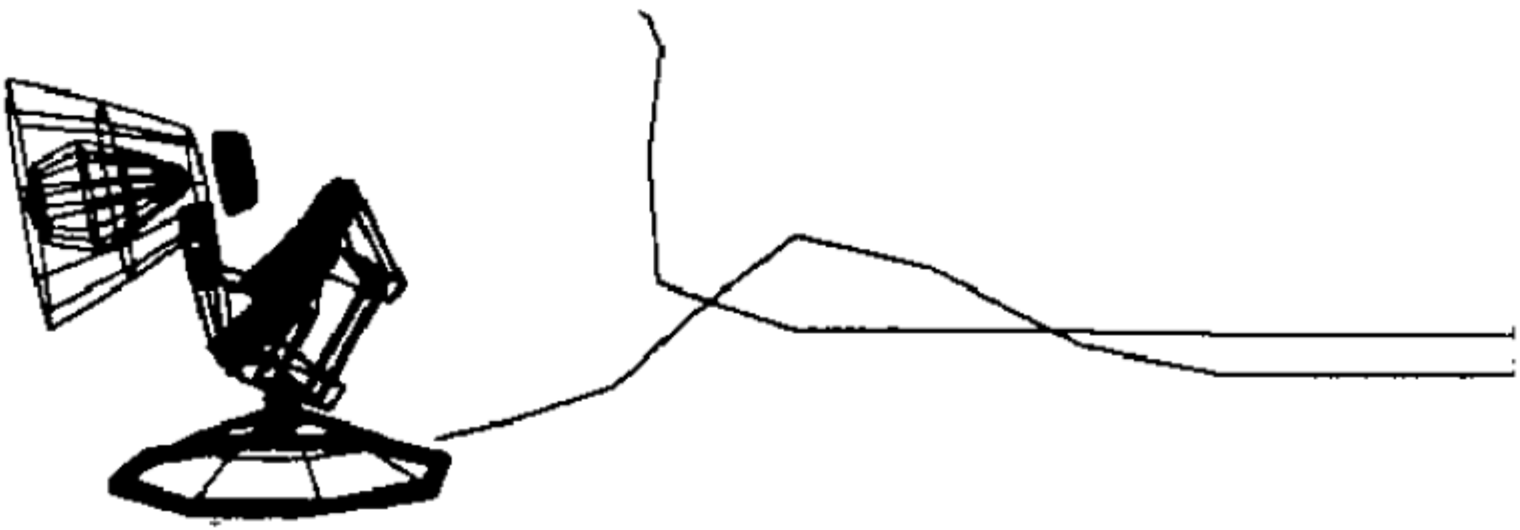


Luxo Jr

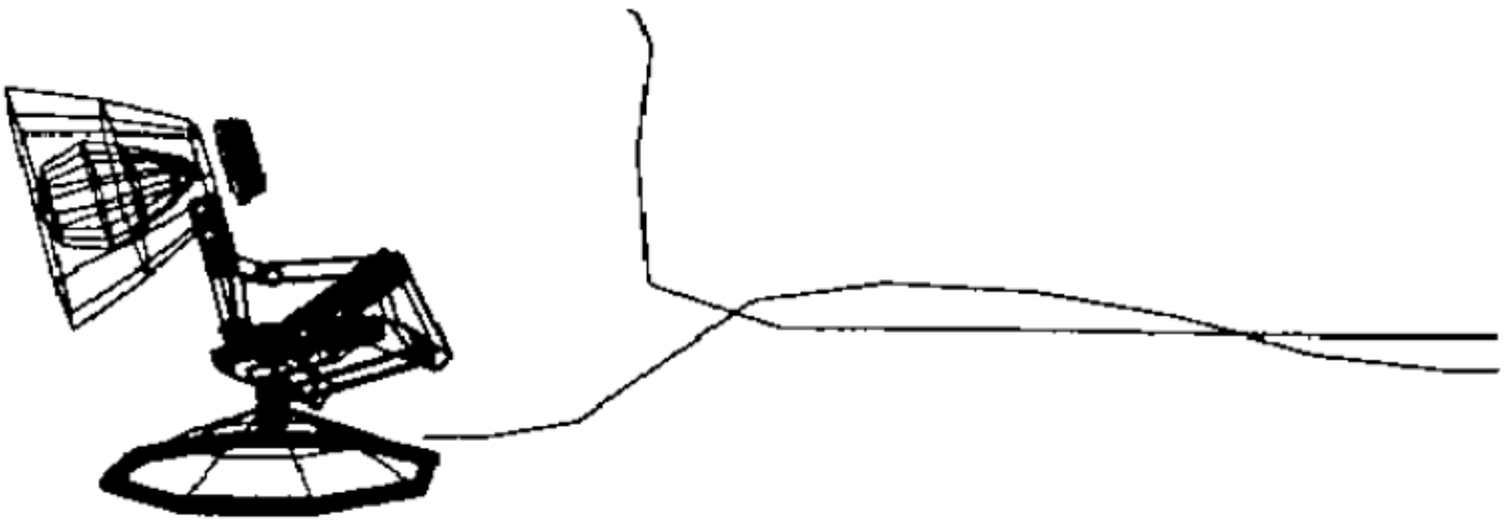


Luxo Jr.

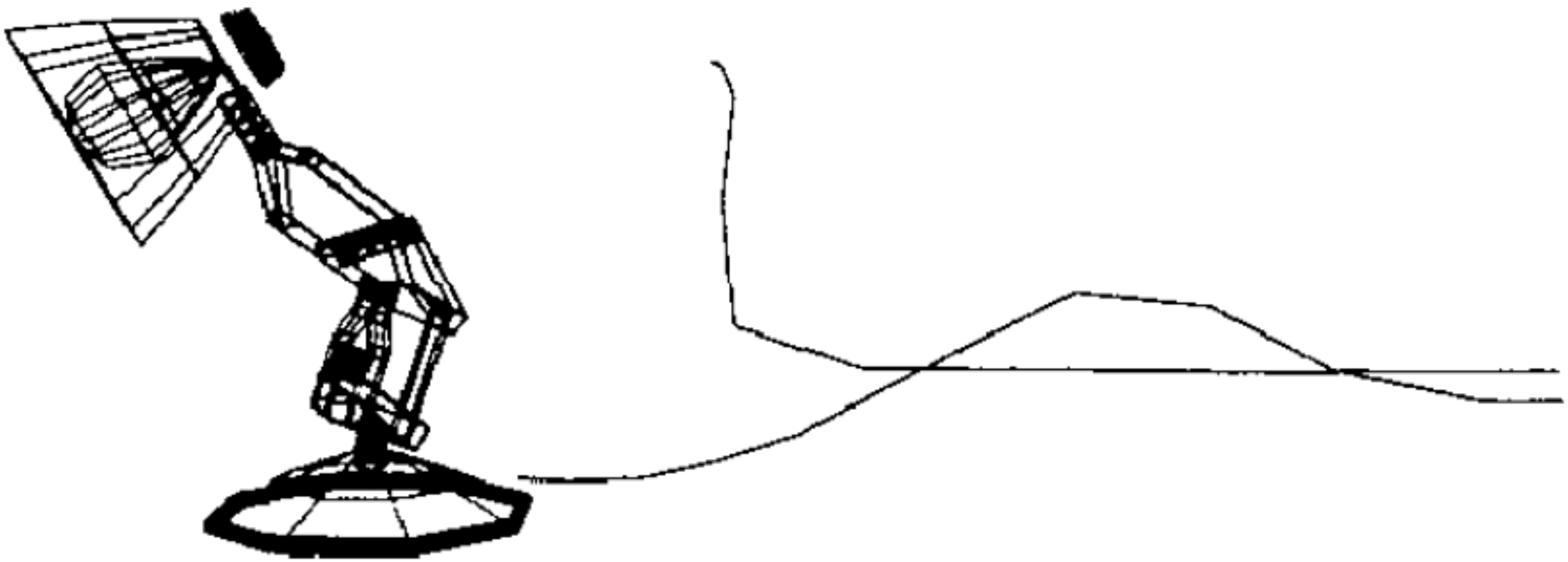
Luxo Jr.



Luxo Jr.



Luxo Jr.



Squash

- Scale one coordinate by matrix multiplication