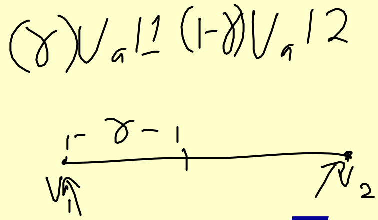


Transformations I

Lecture Set 5

CS5600 Computer Graphics

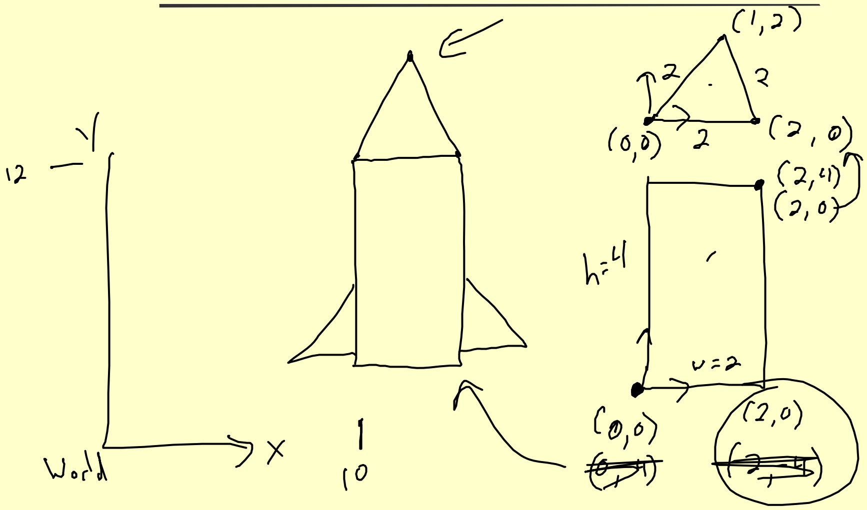
From: Rich Riesenfeld
Spring 2009

Transformations and Matrices

- Transformations are functions
- Matrices are function representations
- Matrices represent linear transf's
- $\{2x2 \text{ Matrices}\} \equiv \{2D \text{ Linear Transf's}\}$

Spring 2009 Utah School of Computing 2

Rocket



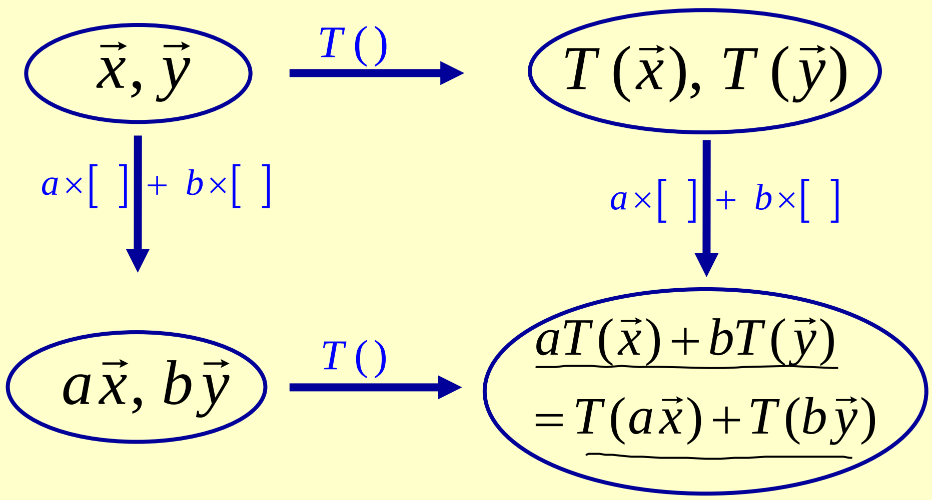
What is a $2D$ *Linear Transf* ?

Recall from Linear Algebra:

$$\underline{Def}: \quad T(a\vec{x}+\vec{y}) = aT(\vec{x}) + T(\vec{y}),$$

for scalar a and vectors $\vec{x}$ and $\vec{y}$.

Look at a Diagram



Spring 2009 Utah School of Computing 5

Scale in *x* by 2: $S_{2_x}(v)$

Scale in *x*, by 2, say:

$$\left(2\left(x_0 + x_1\right), y_0 + y_1\right) = \left(2x_0 + 2x_1, y_0 + y_1\right)$$

$$= \left(2x_0, y_0\right) + \left(2x_1, y_1\right)$$

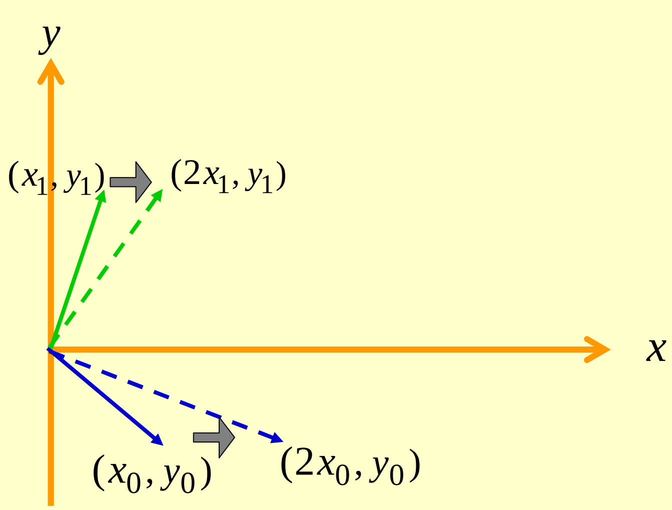
Spring 2009 Utah School of Computing 6

Example:

Scale in x by $S_{2_x}(v)$

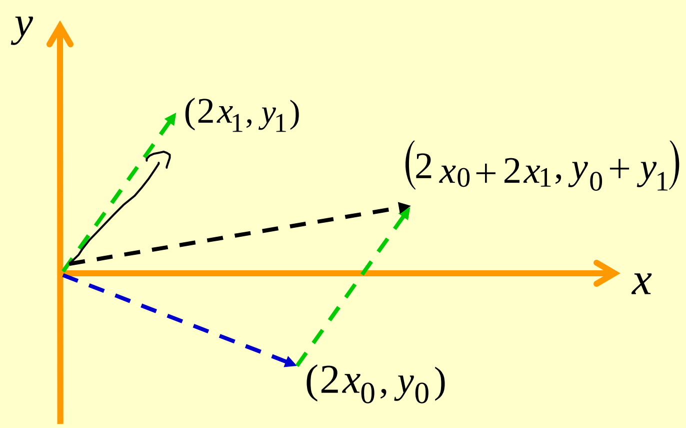
What is the graphical view?

Scale in x by 2: $S_{2_x}(v)$



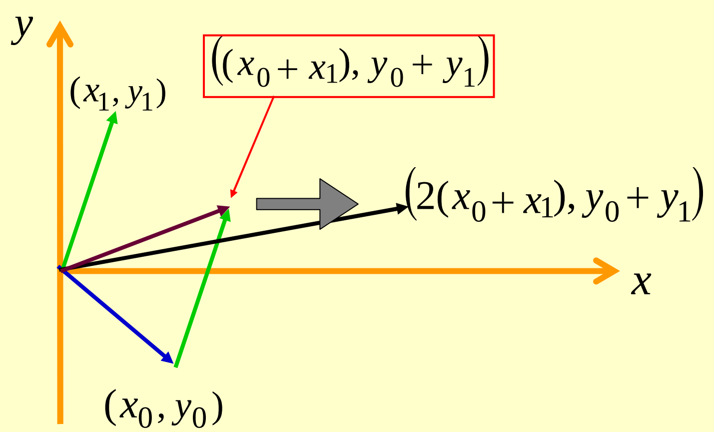
Spring 2009 Utah School of Computing 8

$$\left(2x_0 + 2x_1, y_0 + y_1\right)$$



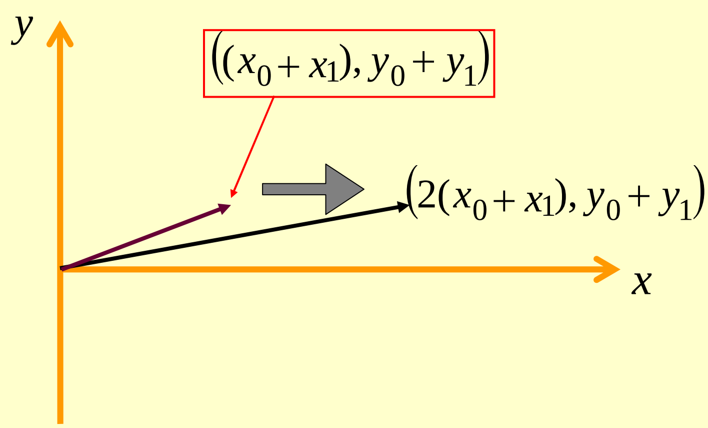
Spring 2009 Utah School of Computing 9

$$\left(2(x_0 + x_1), y_0 + y_1\right)$$



Spring 2009 Utah School of Computing 10

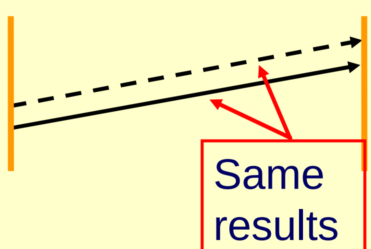
$$\left(2(x_0+x_1),\, y_0+y_1\right)$$



Spring 2009 Utah School of Computing 11

Summary on *Scale*

- "*Scale* then *add*," is same as
- "*Add* then *scale*"



Spring 2009 Utah School of Computing 12

Matrix Representation

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

Matrix Representation of $S_{2_y}(v)$

Scale in *y* by 2: $S_{2_y}(v)$

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

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

Matrix Representation $S_2(v)$

Overall *Scale by 2*: $S_2(v)$

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

Spring 2009 Utah School of Computing 15

Matrix Form of Same

$$\underbrace{\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_0 + x_1 \\ y_0 + y_1 \end{bmatrix}}_{\text{Add } x \text{ and } y\text{, then scale}} = \begin{bmatrix} 2\left(x_0 + x_1\right) \\ y_0 + y_1 \end{bmatrix}$$

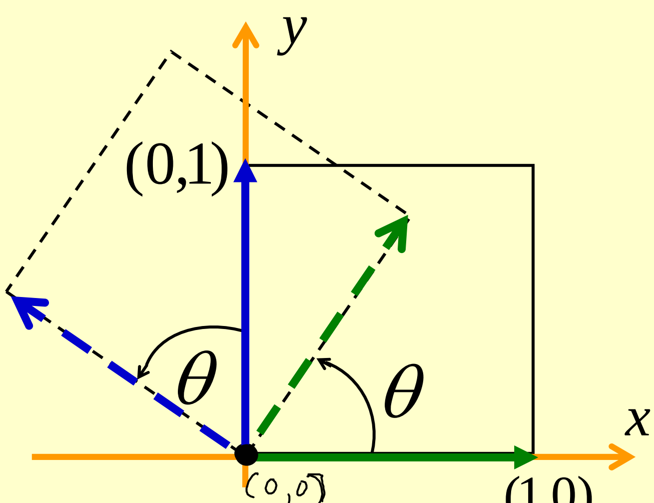
$$\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} + \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \underbrace{\begin{bmatrix} 2x_0 + 2x_1 \\ y_0 + y_1 \end{bmatrix}}_{\text{Scale } x \text{ and } y\text{, then add}}$$

Spring 2009 Utah School of Computing 16

What about Rotation?

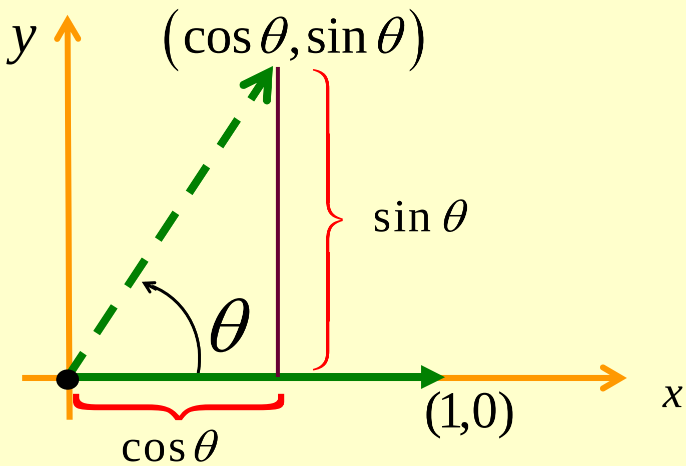
Is it linear?

Rotate by θ



Spring 2009 Utah School of Computing 18

Rotate by θ: 1^{st} *Quadrant*



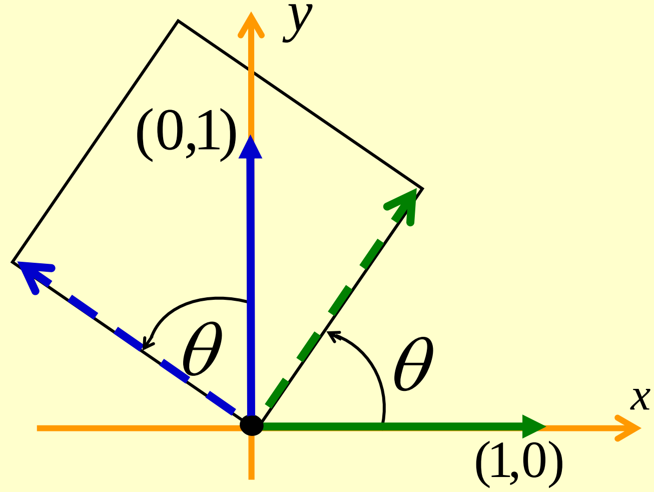
Spring 2009 Utah School of Computing 19

Rotate by θ: 1^{st} *Quadrant*

$$(1,0) \Rightarrow (\cos\theta, \sin\theta)$$

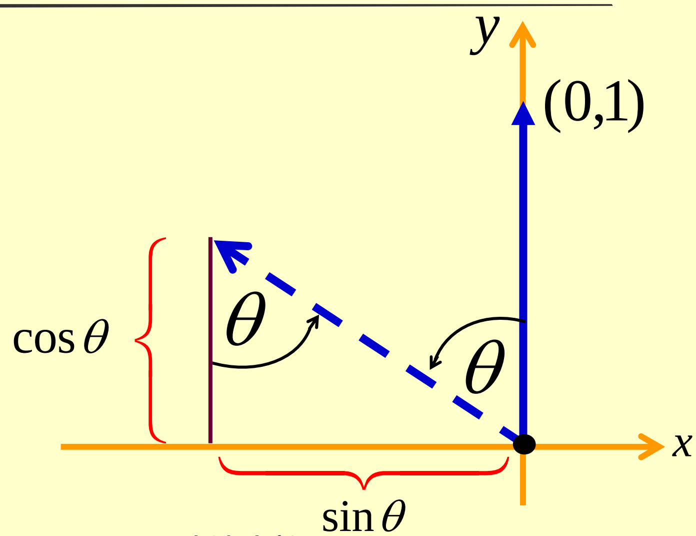
Spring 2009 Utah School of Computing 20

Rotate by θ: 2^{nd} *Quadrant*



Spring 2009 Utah School of Computing 21

Rotate by θ: 2^{nd} *Quadrant*



Spring 2009 Utah School of Computing 22

Rotate by θ: 2^{nd} *Quadrant*

$$(0,1) \Rightarrow (-\sin\theta, \cos\theta)$$

Summary of *Rotation by* θ

$$(1,0) \Rightarrow (\cos\theta, \sin\theta)$$

$$(0,1) \Rightarrow (-\sin\theta, \cos\theta)$$

Summary (Column Form)

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} \Rightarrow \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$$

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix}$$

Using Matrix Notation

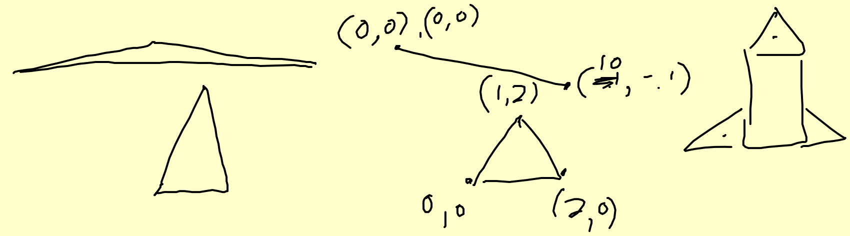
$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}$$

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

(Note that *unit vectors* simply copy columns)

General *Rotation by* θ Matrix

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}$$



Spring 2009 Utah School of Computing 27

What do the off diagonal elements do?

Off Diagonal Elements

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

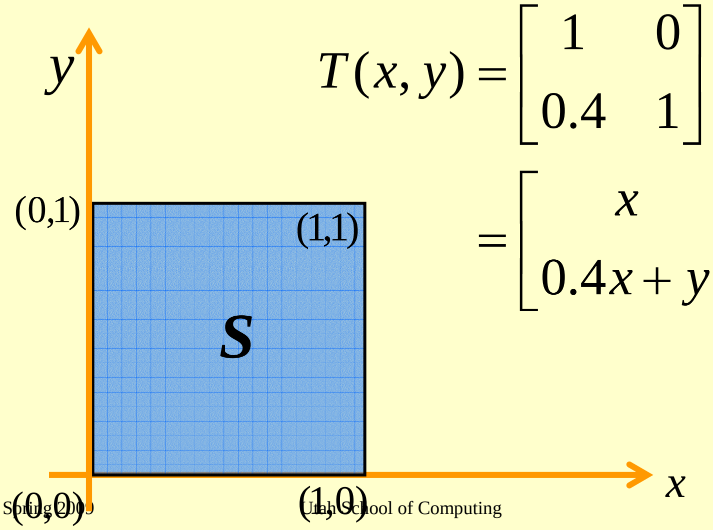
$$\begin{bmatrix} 1 & 0 \\ b & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ b x + y \end{bmatrix}$$

Spring 2009 Utah School of Computing 29

Example 1

$$T(x, y) = \begin{bmatrix} 1 & 0 \\ 0.4 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

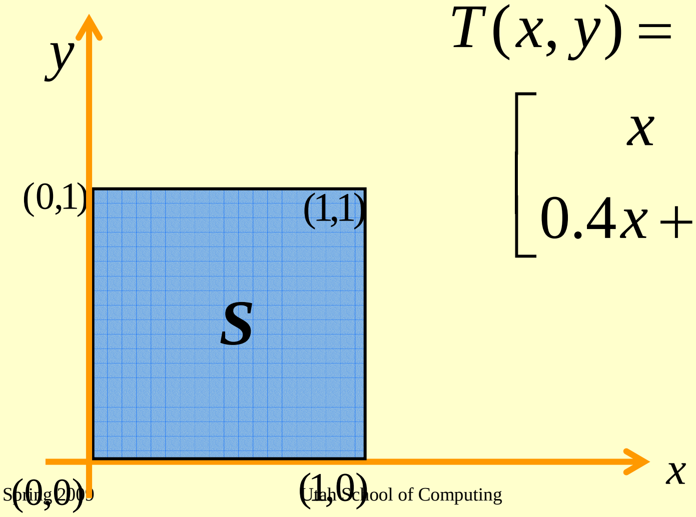
$$= \begin{bmatrix} x \\ 0.4x + y \end{bmatrix}$$



Spring 2009 Utah School of Computing 30

Example 1

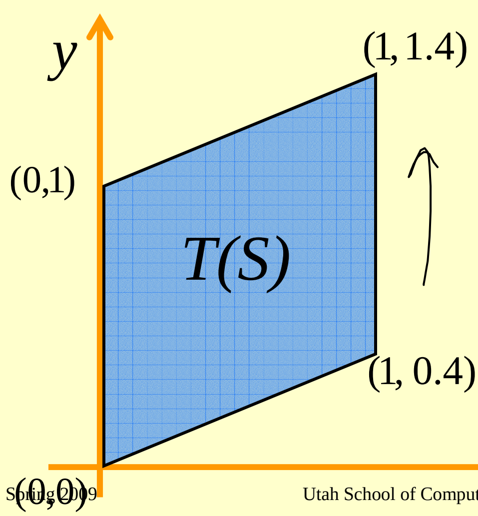
$$T(x, y) = \begin{bmatrix} x \\ 0.4x + y \end{bmatrix}$$



Spring 2009 Utah School of Computing 31

Example 1

$$T(x, y) = \begin{bmatrix} x \\ 0.4x + y \end{bmatrix}$$

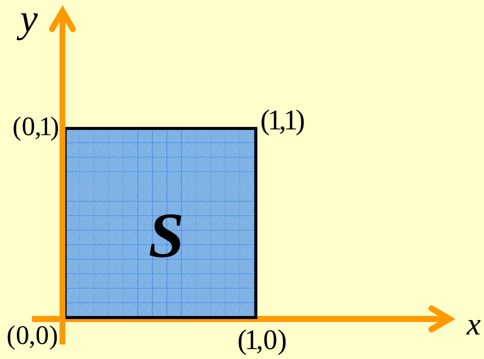


Spring 2009 Utah School of Computing 32

Example 2

$$T(x,y) = \begin{bmatrix} 1 & 0.6 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

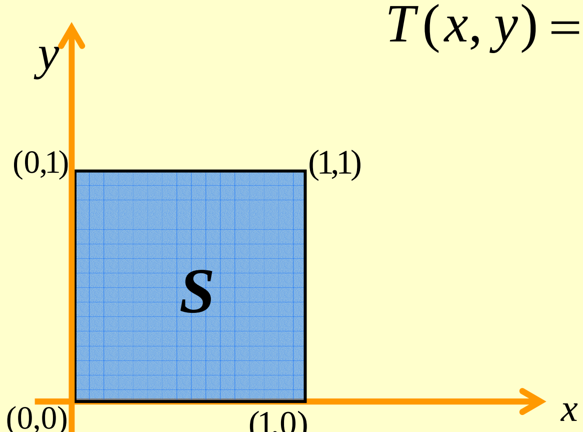
$$= \begin{bmatrix} x + 0.6y \\ y \end{bmatrix}$$



Spring 2009 Utah School of Computing 33

Example 2

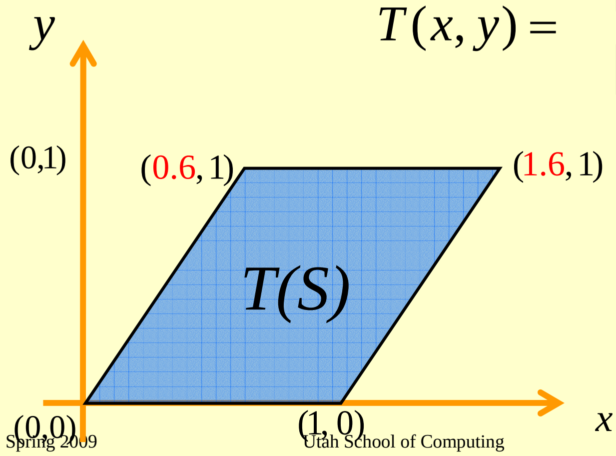
$$T(x,y) = \begin{bmatrix} x + 0.6y \\ y \end{bmatrix}$$



Spring 2009 Utah School of Computing 34

Example 2

$$T(x, y) = \begin{bmatrix} x + 0.6y \\ y \end{bmatrix}$$



Spring 2009 Utah School of Computing 35

Summary

Shear in x:

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

Shear in y:

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

Spring 2009 Utah School of Computing 36

Double Shear: not commutative!

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

$$\begin{bmatrix}1 & 0\\b & 1\end{bmatrix}\begin{bmatrix}1 & a\\0 & 1\end{bmatrix}=\begin{bmatrix}1 & a\\b & (1+ab)\end{bmatrix}$$

Spring 2009 Utah School of Computing 37

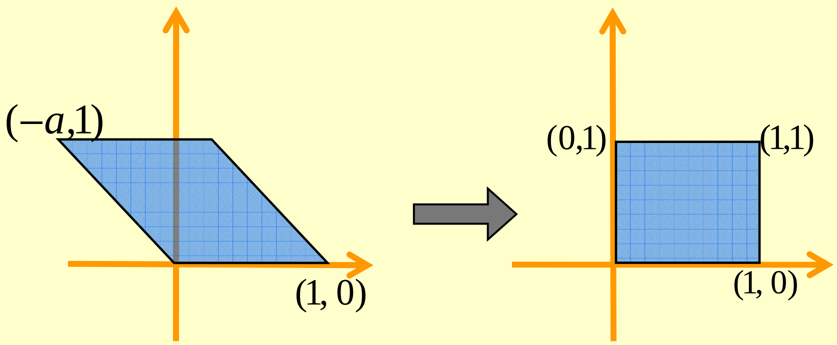
Sample Pts: *unit* pre-images

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

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

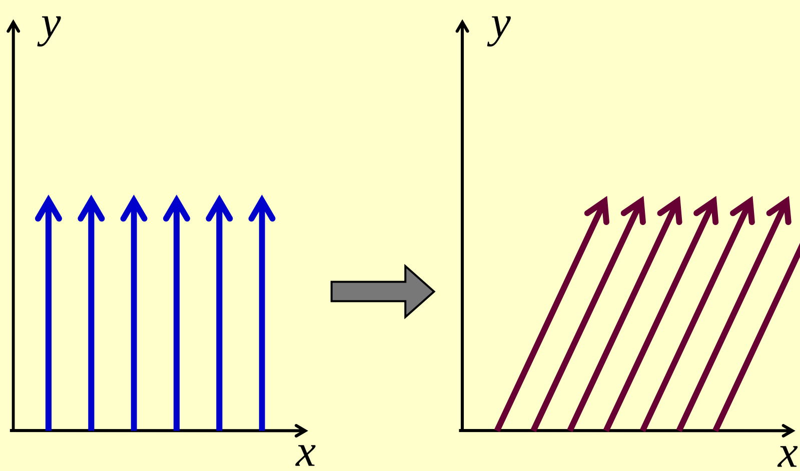
Spring 2009 Utah School of Computing 38

Geometric View of *Shear in x*



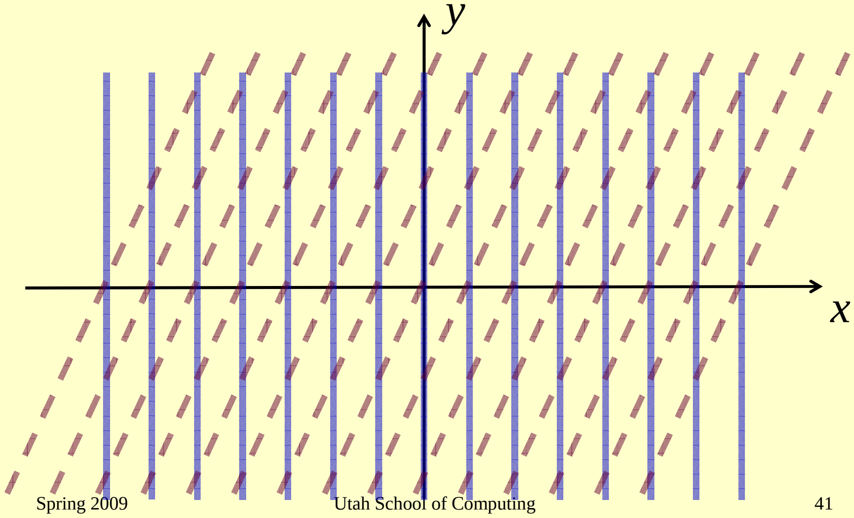
Spring 2009 Utah School of Computing 39

Another Geom View of *Shear in x*



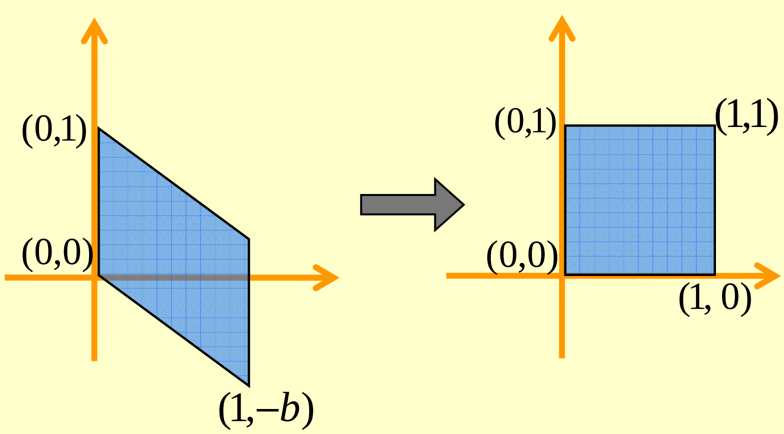
Spring 2009 Utah School of Computing 40

Another Geom View of *Shear in x*

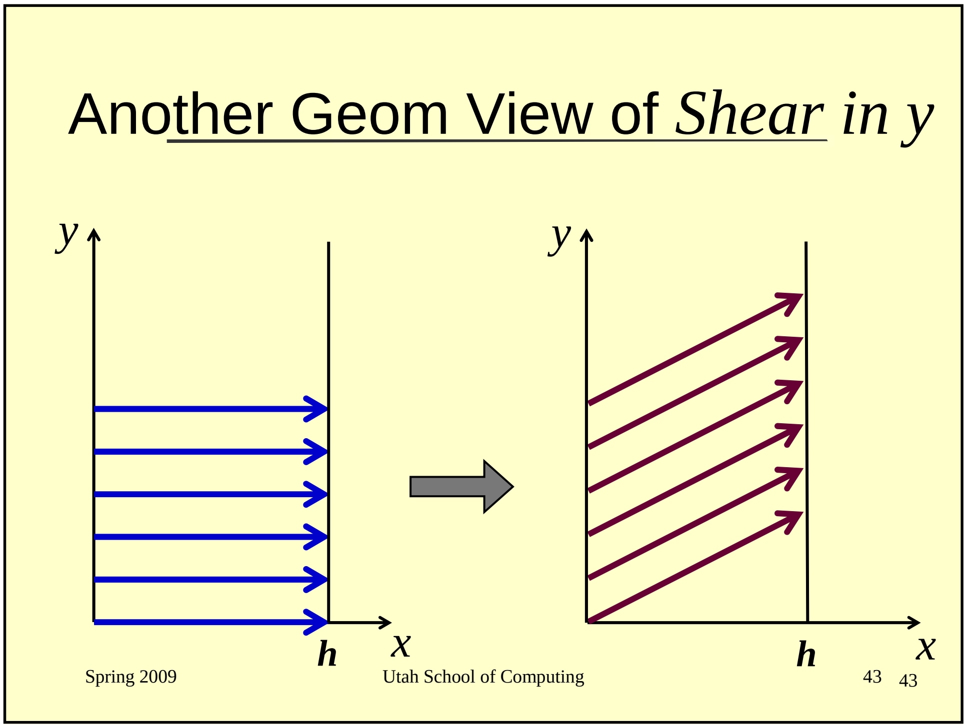


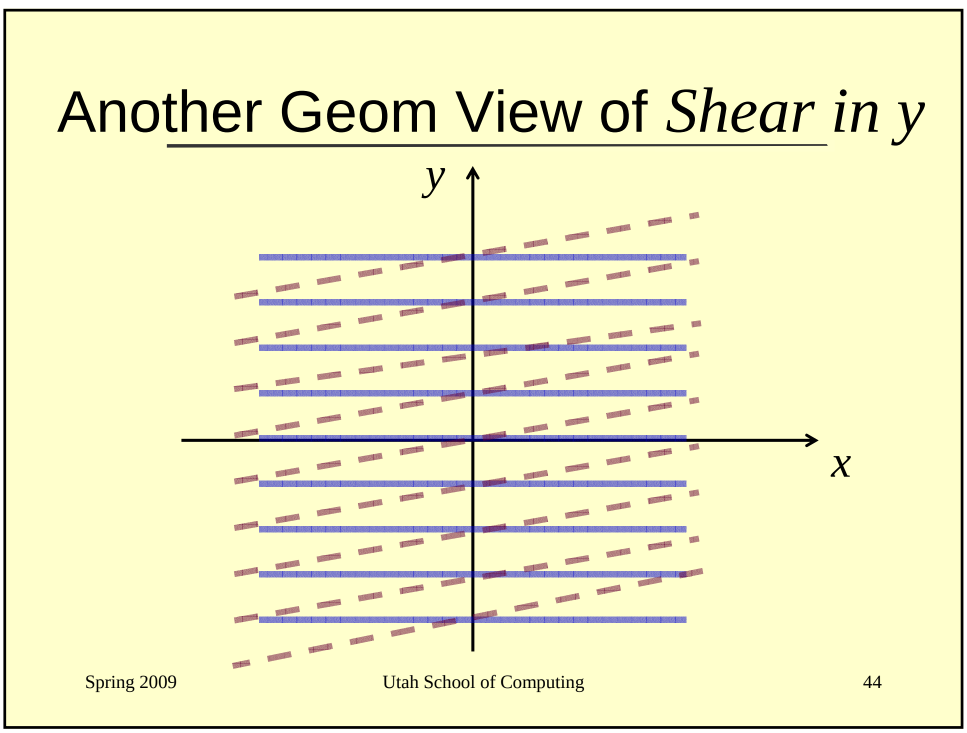
Spring 2009 Utah School of Computing 41

Geometric View of *Shear in y*



Spring 2009 Utah School of Computing 42


Another Geom View of *Shear in y*
y
y
h
x
h
x
Spring 2009
Utah School of Computing
43
43


Another Geom View of *Shear in y*
y
x
Spring 2009
Utah School of Computing
44

"Lazy 1"

Scales Rot shear

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

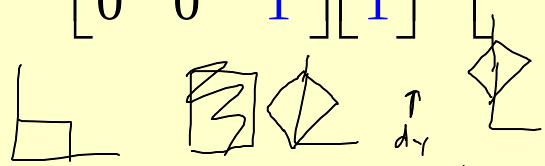
Translation in x

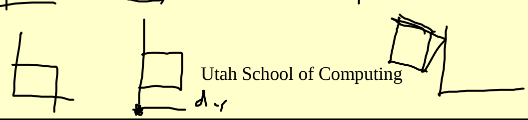
Scale shear Rot — trans

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

Translation in y

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





Homogeneous Coordinates

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

Homogeneous Coordinates

$$\begin{bmatrix} \lambda x \\ \lambda y \\ \lambda \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \leftrightarrow \begin{bmatrix} x \\ y \end{bmatrix}, \quad \text{for } \lambda \neq 0$$

Spring 2009 Utah School of Computing 49

Homogeneous Coordinates

For $\lambda \neq 0$,

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1/\lambda \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1/\lambda \end{bmatrix} = \begin{bmatrix} \lambda x \\ \lambda y \\ 1 \end{bmatrix} \Leftrightarrow \begin{bmatrix} \lambda x \\ \lambda y \end{bmatrix}$$

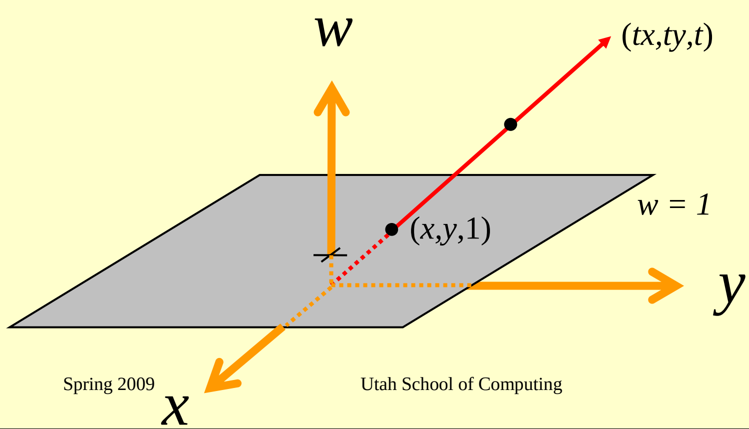
Homogeneous term affects overall scaling

Spring 2009 Utah School of Computing 50

Homogeneous Coordinates

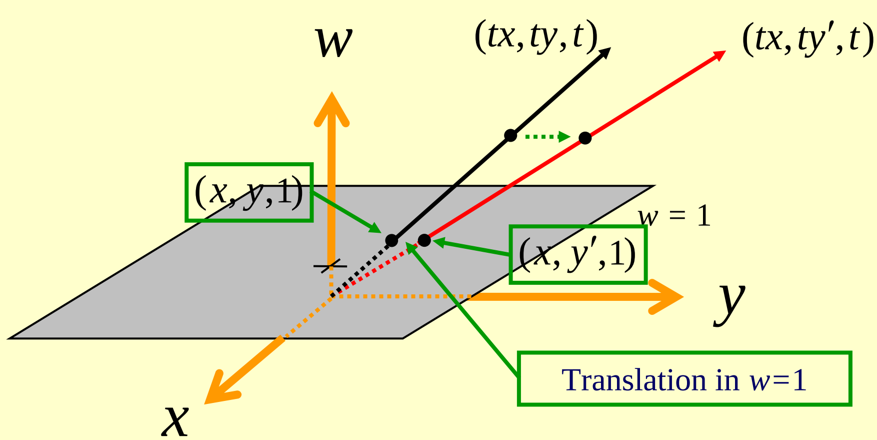
An infinite number of points correspond to $(x,y,1)$.

They constitute the whole line (tx,ty,t).



Spring 2009 Utah School of Computing 51

What does a *shear* do?



Spring 2009 Utah School of Computing 52

Using Homogeneous Coord's

- *Shear* in $3D$
- Effects *translation* in $2D$
- We have used a *linear transformation* (*shear*) in $3D$ to effect a *nonlinear transformation* (*translation*) in $2D$

Spring 2009 Utah School of Computing 53

Translation by $\vec{d}: T(\vec{x}) = \vec{x} + \vec{d}$

$$
\begin{aligned}
T(\vec{u}+\vec{v}) &= (\vec{u}+\vec{v})+\vec{d} \\
T(\vec{u})+T(\vec{v}) &= (\vec{u}+\vec{d})+(\vec{v}+\vec{d}) \\
&= \vec{u}+\vec{v}+2\vec{d} \\
&= T(\vec{u}+\vec{v})+\vec{d}
\end{aligned}
$$

$$T(\vec{u}+\vec{v}) \neq T(\vec{u})+T(\vec{v})$$

Spring 2009 Utah School of Computing 54

Lots Going On Here!

We've got Affine Transformations*:*

Linear + Translation

Compound Transformations

- Build up *compound transformations* by *concatenating* elementary ones
- Use for complicated motion
- Use for complicated modeling

$$Rot_y \cdot Trans_x \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Elementary Transformations

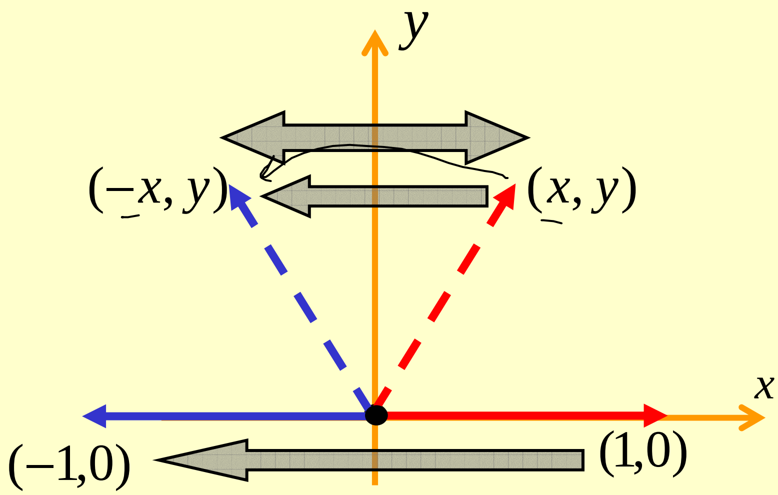
- *Scale*: $S_{\lambda_x}(v),\ S_{\lambda_y}(v)$ ✓ +
- *Rotate*: $R_{\theta_x}(v),\ R_{\theta_y}(v)$ ✓ +
- *Translate*: $T_{d_x}(v),\ T_{d_y}(v)$ ✓ +
- *Shear*: $Sh_{\lambda_x}(v),\ Sh_{\lambda_y}(v)$ −
- *Reflect*: $Rf_x(v),\ Rf_y(v)$ −

Refection about *y-axis*

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

$$x \leftrightarrow -x$$

Reflection about *y-axis*

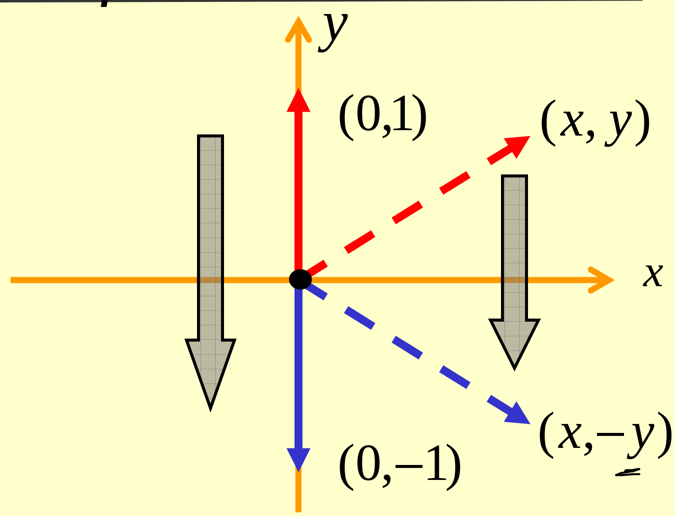


Reflection about *x-axis*

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

$y \leftrightarrow -y$

Reflection about *x-axis*



Spring 2009 Utah School of Computing 61

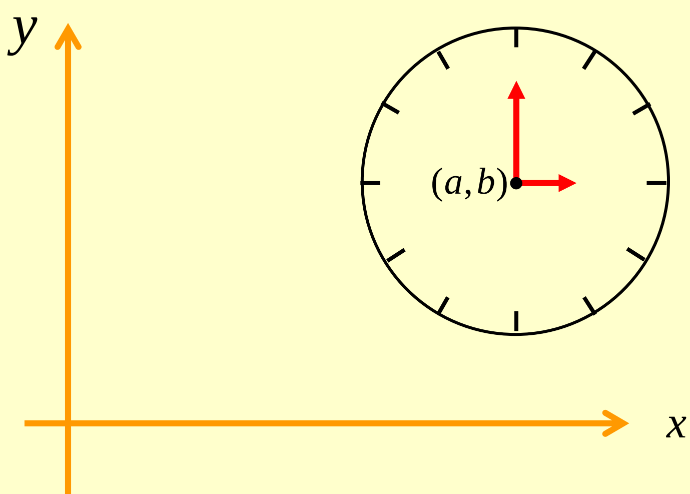
Is *Reflection* "*Elementary*"?

- Can we effect reflection in an elementary way?
- (More elementary means scale, shear, rotation, translation.)

Spring 2009 Utah School of Computing 62

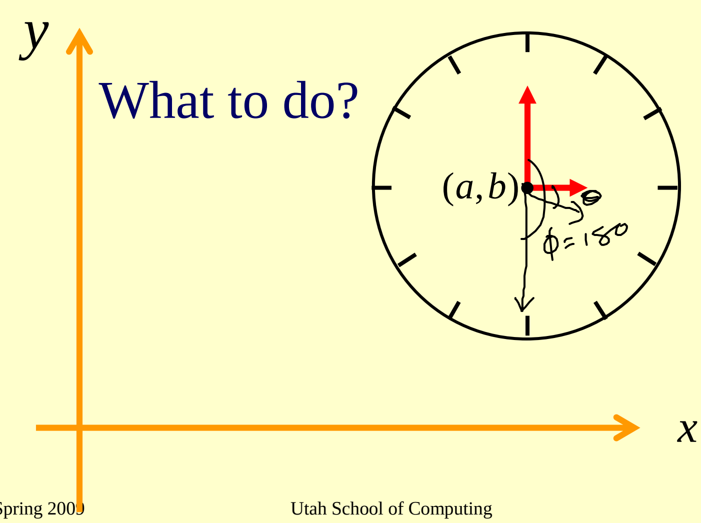
Reflection = Scale (-1)

Ex: Advance clock hands



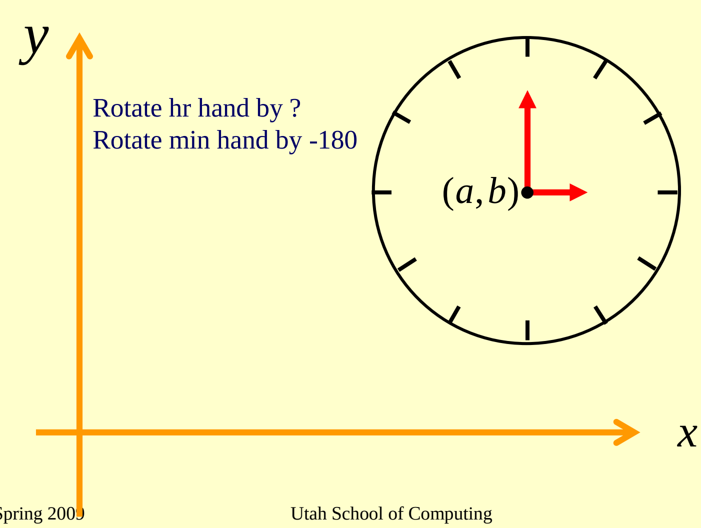
Ex: Advance clock hands: 30mins

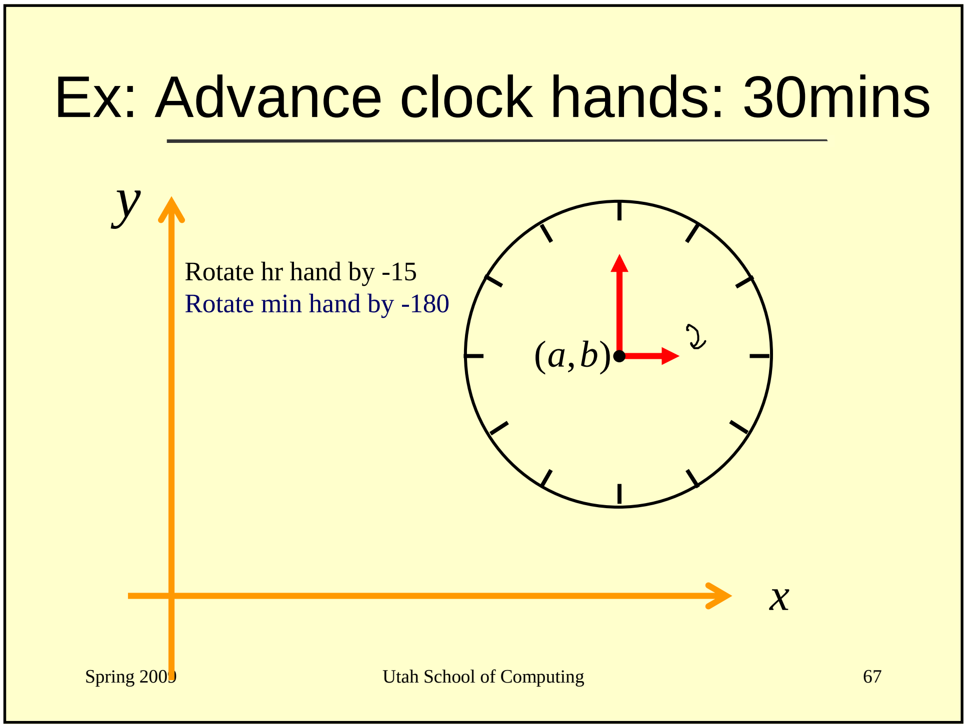
What to do?

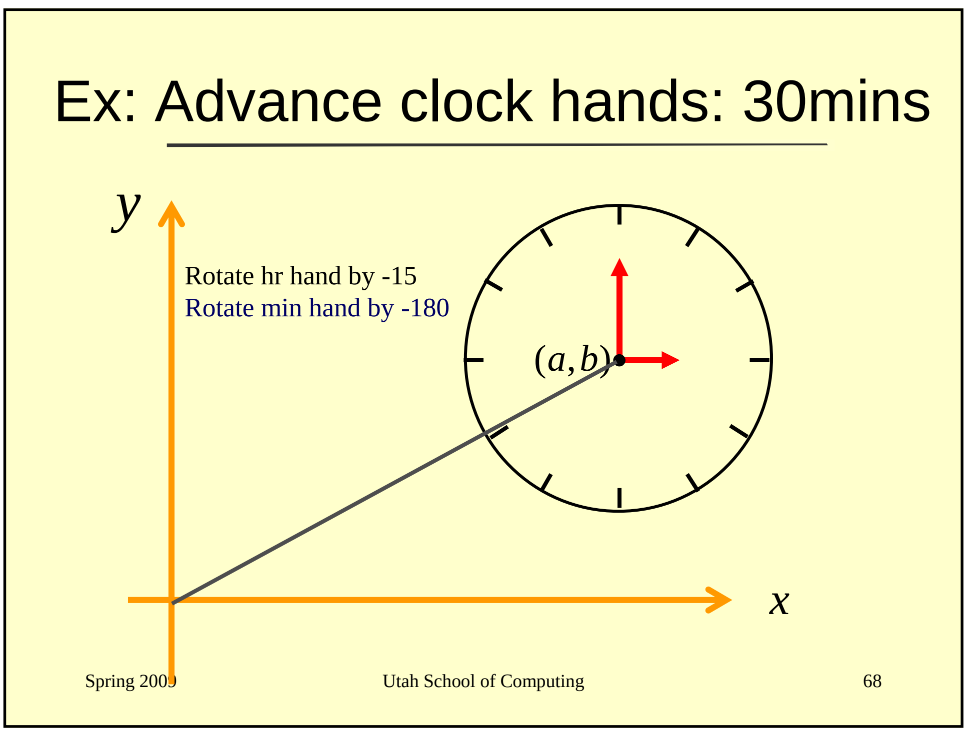


Ex: Advance clock hands: 30mins

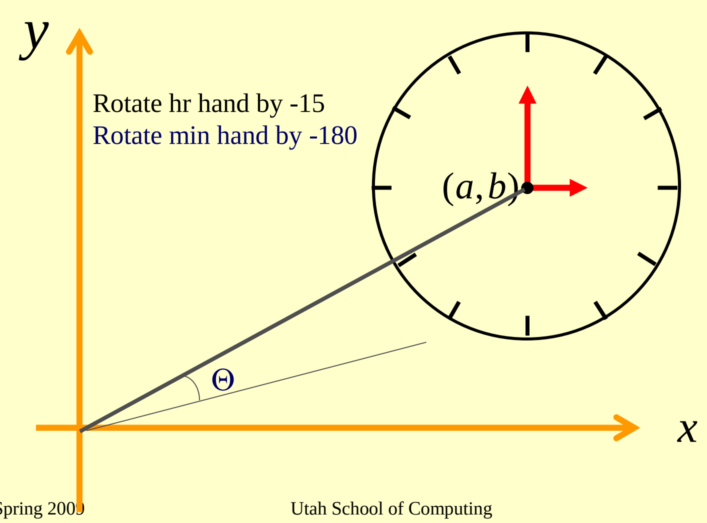
Rotate hr hand by ?
Rotate min hand by -180




Ex: Advance clock hands: 30mins
y
Rotate hr hand by -15
Rotate min hand by -180
(a,b)
x
Spring 2009
Utah School of Computing
67

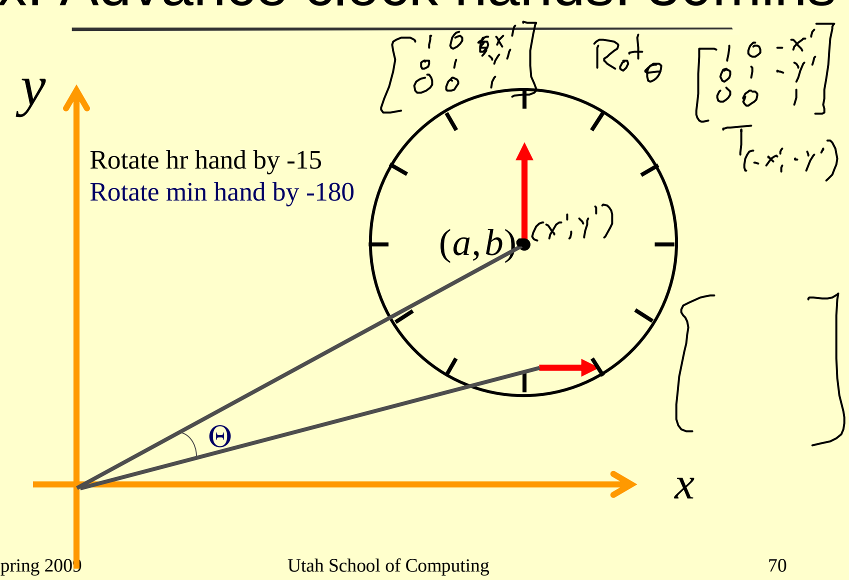

Ex: Advance clock hands: 30mins
y
Rotate hr hand by -15
Rotate min hand by -180
(a,b)
x
Spring 2009
Utah School of Computing
68

Ex: Advance clock hands: 30mins

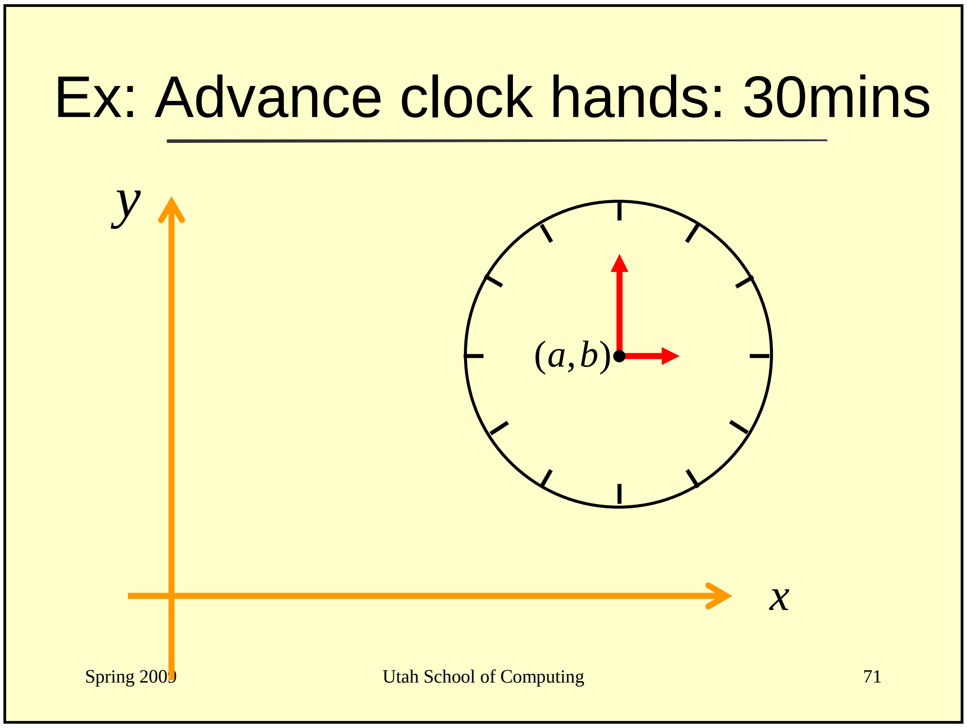


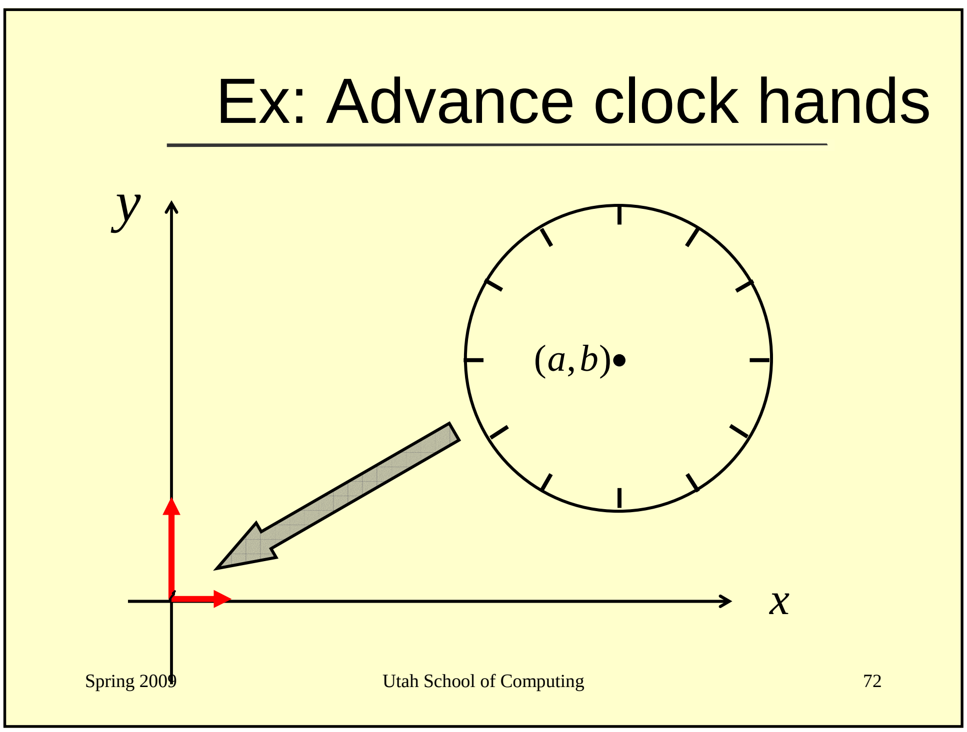
Spring 2009 Utah School of Computing 69

Ex: Advance clock hands: 30mins

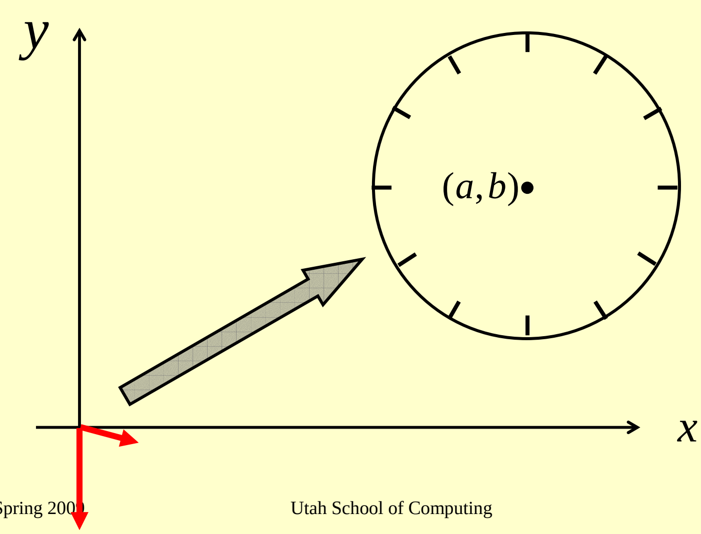


Spring 2009 Utah School of Computing 70


Ex: Advance clock hands: 30mins
y
x
(a,b)
Spring 2009
Utah School of Computing
71

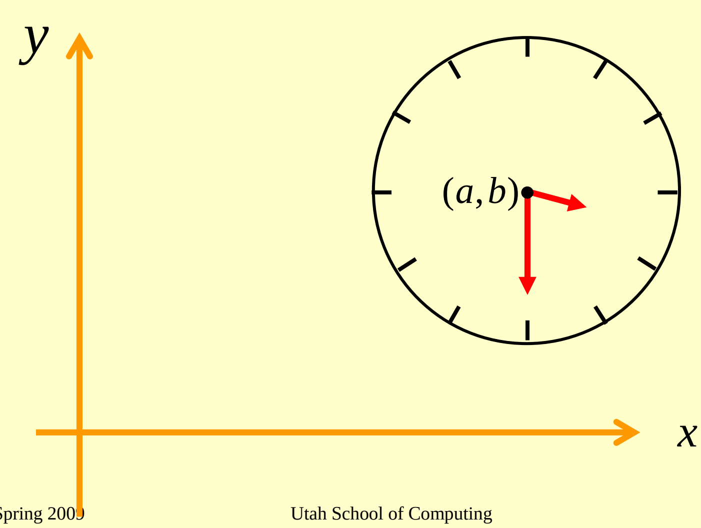

Ex: Advance clock hands
y
x
(a,b)
Spring 2009
Utah School of Computing
72

Ex: Advance clock hands

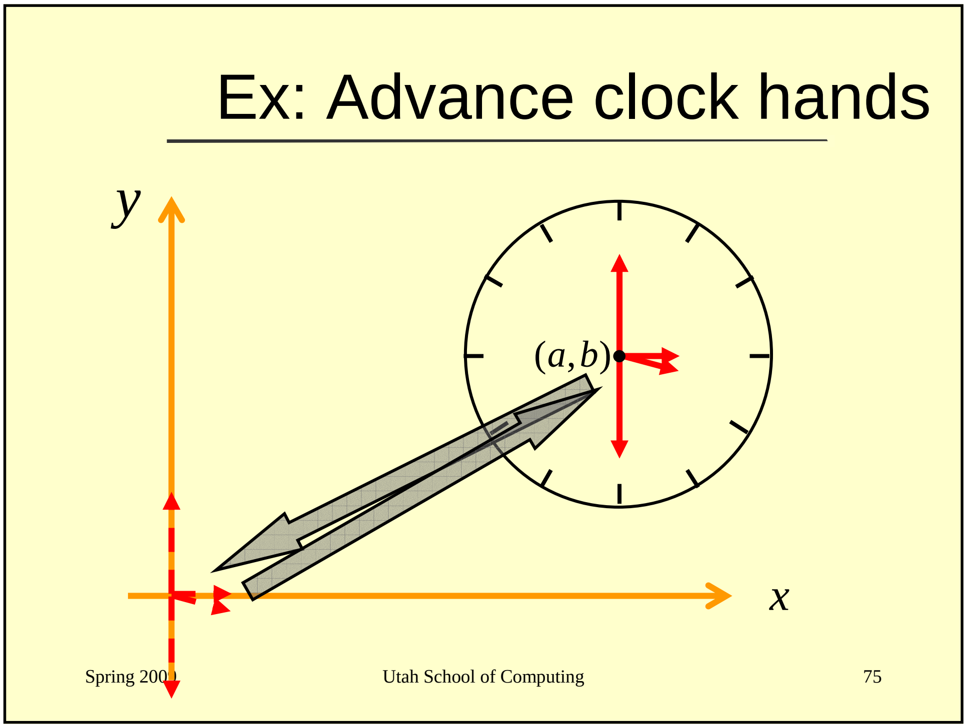


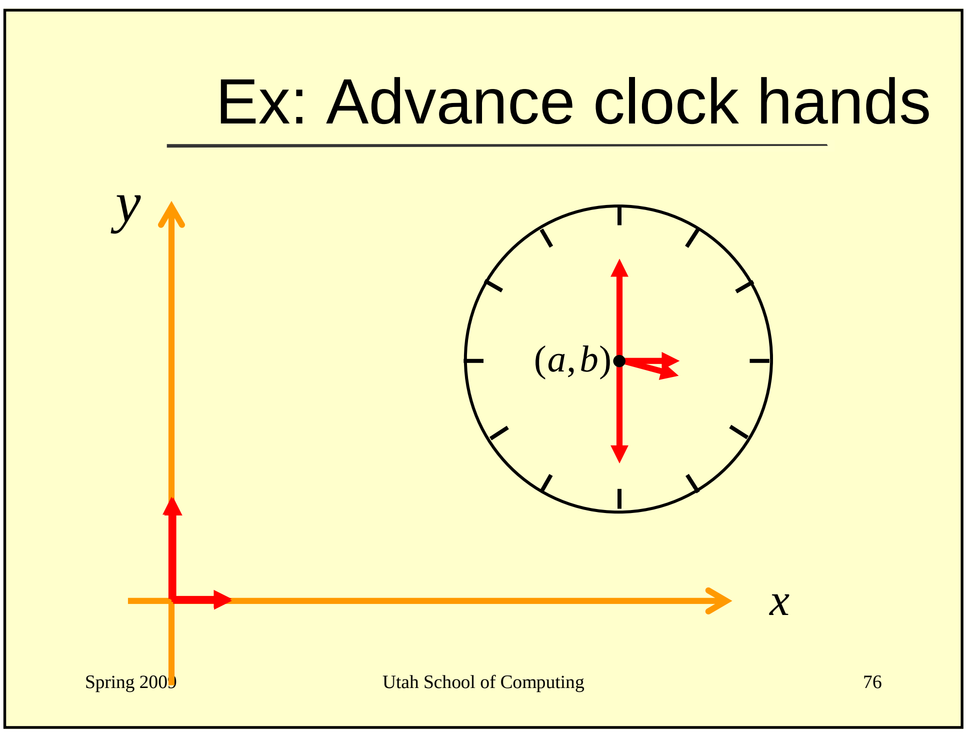
Spring 2009 Utah School of Computing 73

Ex: Advance clock hands



Spring 2009 Utah School of Computing 74


Ex: Advance clock hands
y
(a,b)
x
Spring 2009
Utah School of Computing
75


Ex: Advance clock hands
y
(a,b)
x
Spring 2009
Utah School of Computing
76

Clock *Transformations*

- *Translate* to *Origin*
- *Move* hand with *rotation*
- ~~*Move*~~ Translate hand back to clock
- *Do* other hand

Spring 2009 Utah School of Computing 77

Clock Transformations

$$T_s = T(a,b)\ R(t)\ T(-a,-b) \leftarrow \text{Point}$$

$$T_b = T(a,b)\ R(12*t)\ T(-a,-b)$$

$$where \quad t = -15^\circ$$

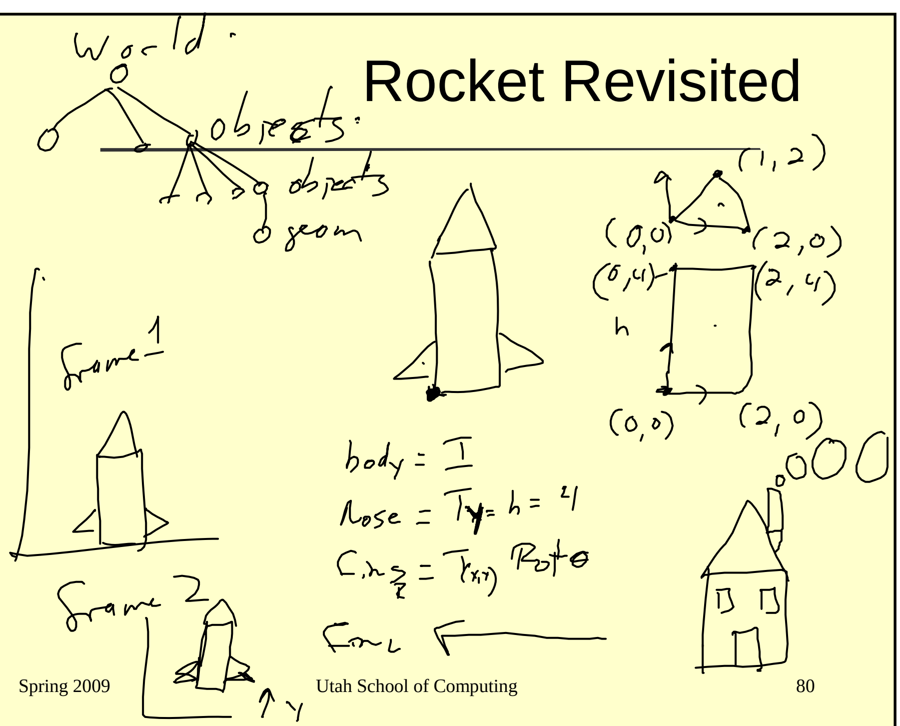
Spring 2009 Utah School of Computing 78

Clock Transformations

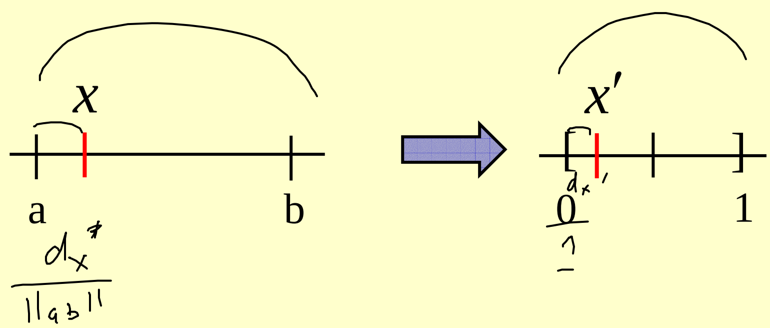
$$\underbrace{\begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Translate Back}} \underbrace{\begin{bmatrix} \cos(t) & -\sin(t) & 0 \\ \sin(t) & \cos(t) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Rotate About Origin}} \underbrace{\begin{bmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Translate to Origin}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \quad \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}$$

Rocket Revisited



Map: $[a,b] \Rightarrow [0,1]$



Spring 2009 Utah School of Computing 81

Map: $[a,b] \Rightarrow [0,1]$

- Translate to Origin

$$[a,b] \to [a-a, b-a] = [0, b-a]$$

- Map x to translated interval

$$x \to x-a$$

Spring 2009 Utah School of Computing 82

Map: $[a,b] \Rightarrow [0,1]$

$$\begin{bmatrix} \frac{1}{b-a} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -a \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \left(\frac{x-a}{b-a}\right) \\ y \\ 1 \end{bmatrix}$$

$$S_x\left(\frac{1}{b-a}\right) \qquad T_x(-a)$$

origin

Spring 2009 Utah School of Computing 83

Map: $[a,b] \Rightarrow [0,1]$

- Normalize the interval

$$[0, b-a] \rightarrow \frac{1}{b-a}[a-a, b-a] = [0,1]$$

- Map x to normalized interval

$$x \rightarrow \frac{x-a}{b-a}$$

Spring 2009 Utah School of Computing 84

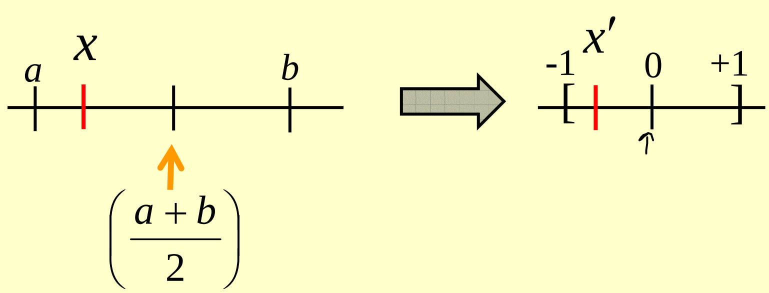
Just Look at Υ[1]

$$\begin{bmatrix} \left(\frac{1}{b-a}\right) & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & -a \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} \left(\frac{x-a}{b-a}\right) \\ 1 \end{bmatrix}$$

$S_x\left(\frac{1}{b-a}\right)$ $T_x(-a)$

This is a homogeneous form for 1D

Map: [a,b] $\Rightarrow$ [-1,1]



Map: $[a,b] \Rightarrow [-1,1]$

- Translate center of interval to origin

$$x \rightarrow \left[x - \frac{a+b}{2}\right]$$

- Normalize interval to [-1,1]

$$\left[x - \frac{a+b}{2}\right] \rightarrow \frac{2}{b-a}\left[x - \frac{a+b}{2}\right]$$

Spring 2009 Utah School of Computing 87

Map: $[a,b] \Rightarrow [c,d]$

- First map $[a,b]$ to $[0,1]$
  - (We already did this)
- Then map $[0,1]$ to $[c,d]$

Spring 2009 Utah School of Computing 88

Map: $[0,1] \Rightarrow [c,d\,]$

- Scale $[0,1]$ by $[c,d\,]$
- Then translate by c
- That is, in $1D$ homogeneous form:

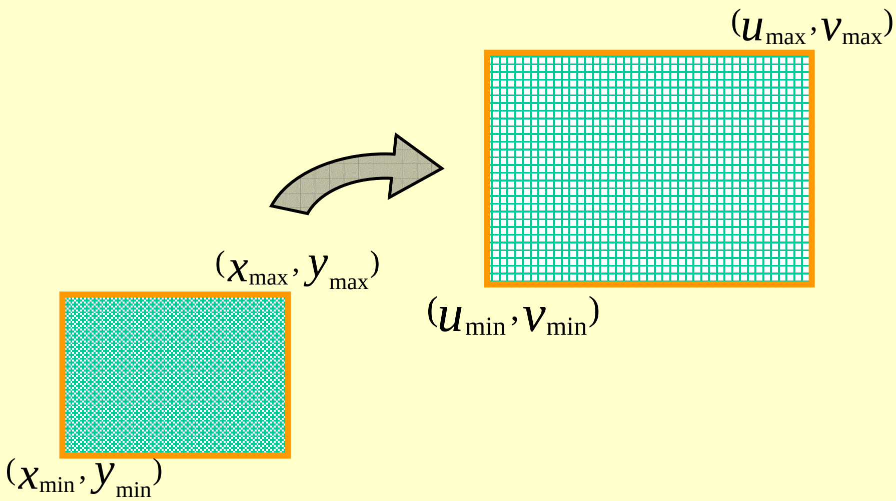
$$\underbrace{\begin{bmatrix}1 & c\\0 & 1\end{bmatrix}}\underbrace{\begin{bmatrix}(d-c) & 0\\0 & 1\end{bmatrix}}\begin{bmatrix}x\\1\end{bmatrix}=\begin{bmatrix}(d-c)x+c\\1\end{bmatrix}$$

All Together: Map: $[a,b] \Rightarrow [c,d\,]$

$$\underbrace{\begin{bmatrix}1 & c\\0 & 1\end{bmatrix}\begin{bmatrix}(d-c) & 0\\0 & 1\end{bmatrix}}\underbrace{\begin{bmatrix}\left(\frac{1}{b-a}\right) & 0\\0 & 1\end{bmatrix}\begin{bmatrix}1 & -a\\0 & 1\end{bmatrix}}\begin{bmatrix}x\\1\end{bmatrix}$$

$$=\begin{bmatrix}1 & c\\0 & 1\end{bmatrix}\begin{bmatrix}\left(\frac{d-c}{b-a}\right) & 0\\0 & 1\end{bmatrix}\begin{bmatrix}1 & -a\\0 & 1\end{bmatrix}\begin{bmatrix}x\\1\end{bmatrix}$$

Now Map Rectangles



Spring 2009 Utah School of Computing 91

Transformation in x and y

$$\begin{bmatrix} 1 & 0 & u_{min} \\ 0 & 1 & v_{min} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_x & 0 & 0 \\ 0 & \lambda_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_{min} \\ 0 & 1 & -y_{min} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

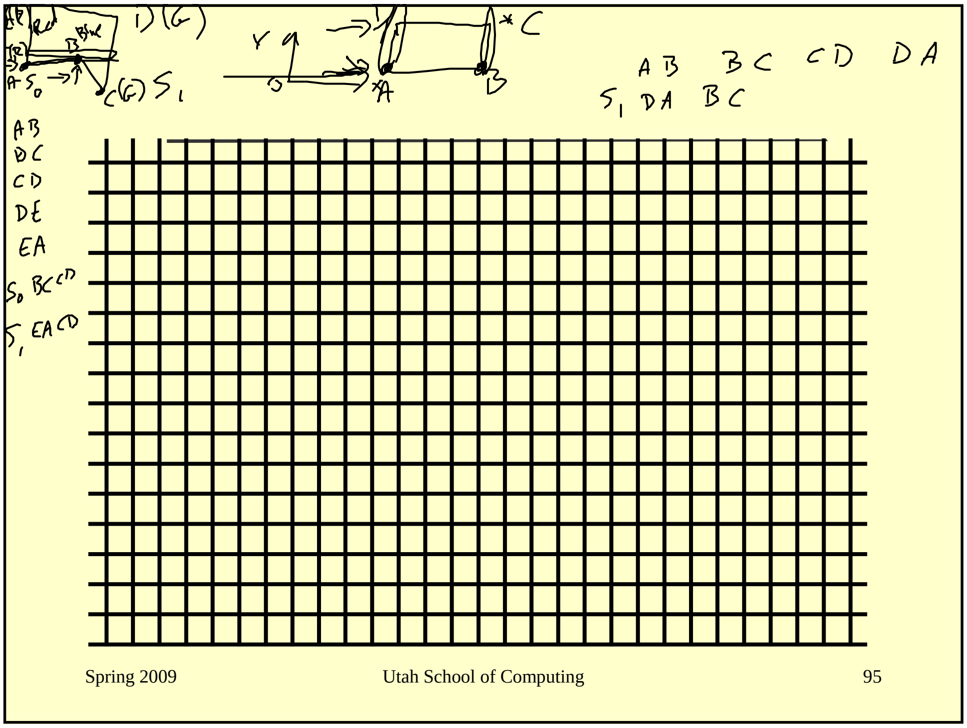
where, $\lambda_x = \left(\frac{u_{max} - u_{min}}{x_{max} - x_{min}} \right), \quad \lambda_y = \frac{v_{max} - v_{min}}{y_{max} - y_{min}}$

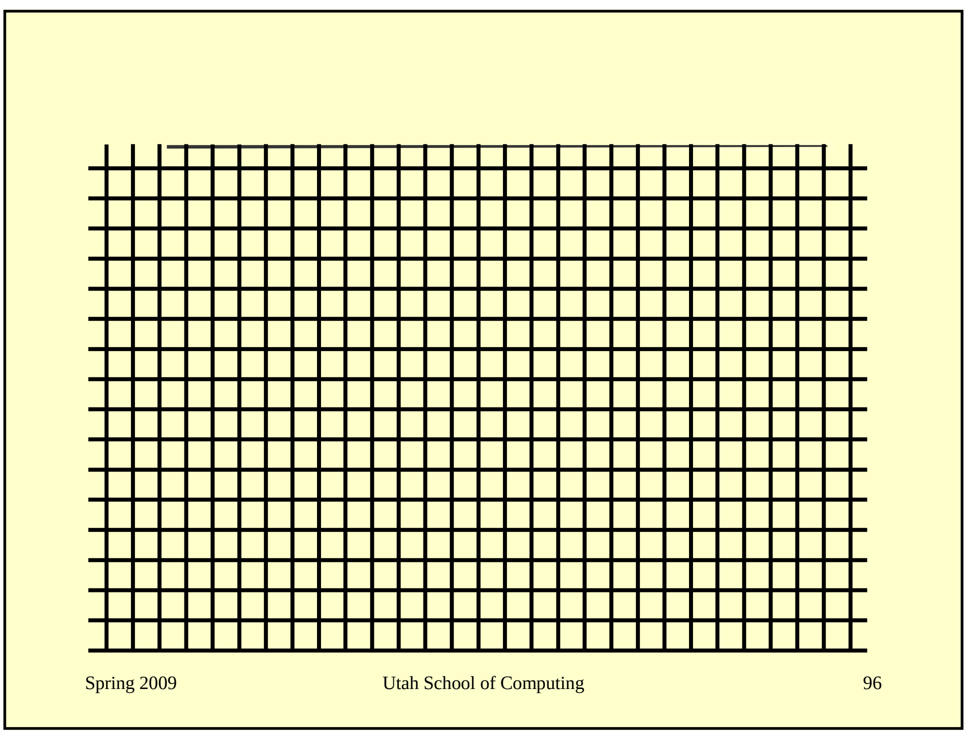
Spring 2009 Utah School of Computing 92

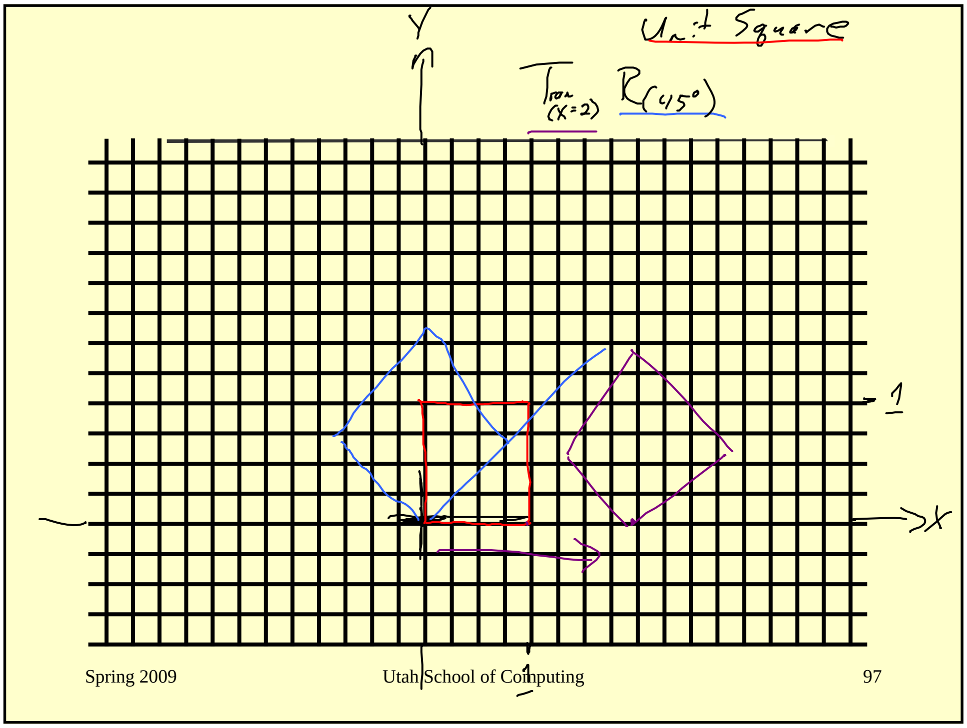
This is *Viewport Transformation*

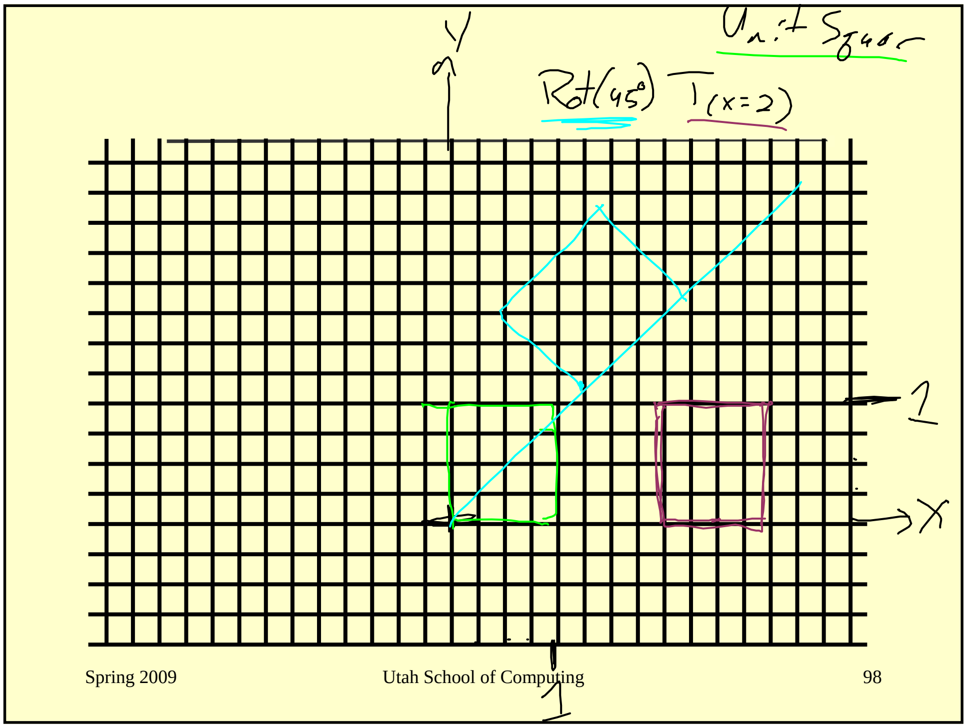
- Good for mapping objects from one coordinate system to another
- This is what we do with windows and viewports

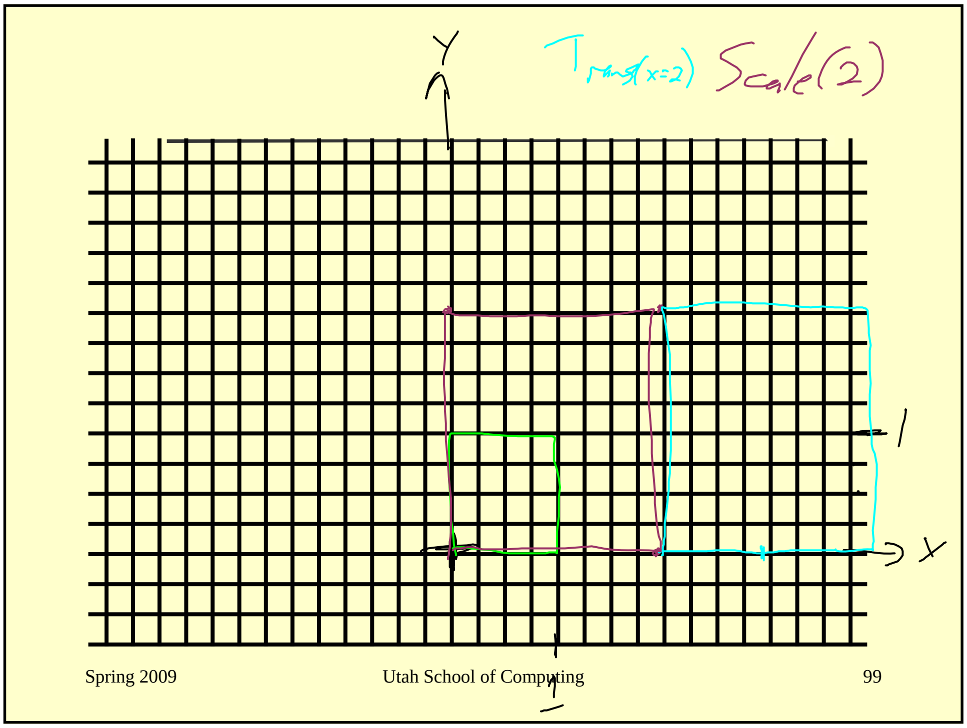
Space Example

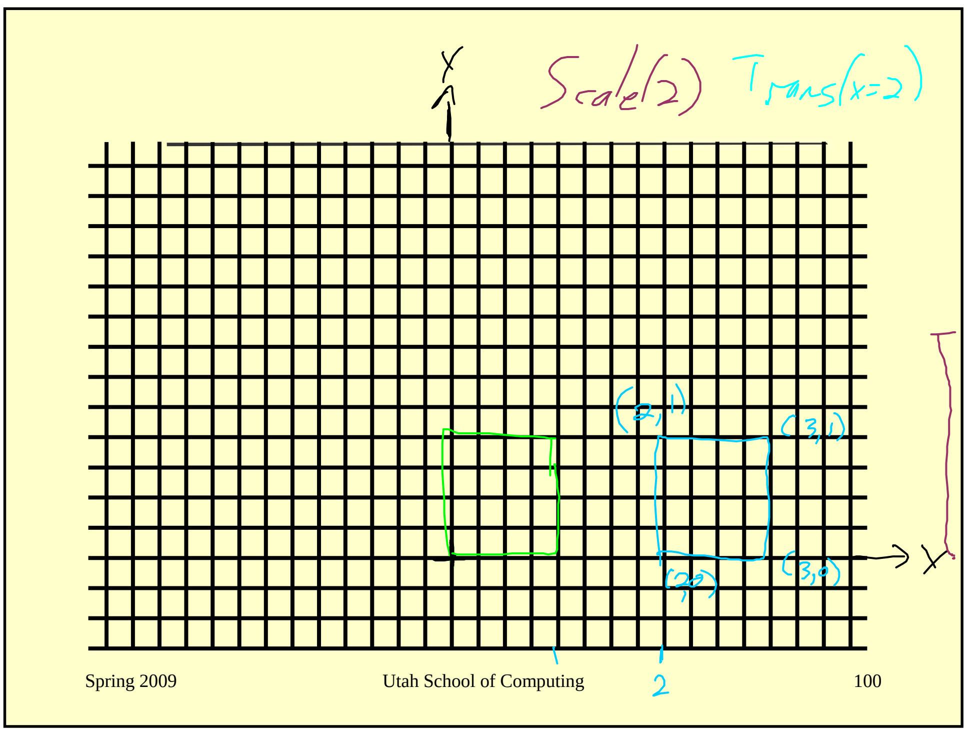

AB BC CD DA
S_1 DA BC
Spring 2009
Utah School of Computing
95


Spring 2009
Utah School of Computing
96


Unit Square
Ttran (x=2) R(45°)
Y
X
1
Spring 2009
Utah School of Computing
97


Unit Squar
Rot(45°) T(x=2)
Y
X
1
Spring 2009
Utah School of Computing
98


Y
Trans(x=2) Scale(2)
1
X
1
Spring 2009
Utah School of Computing
99


Y
Scale(2) Trans(x=2)
(2,1)
(3,1)
(3,0)
X
1
2
Spring 2009
Utah School of Computing
100

3D Transformations

- *Scale* $S_x(\lambda),\ S_y(\lambda),\ S_z(\lambda)$
- *Rotate* $R_x(\theta),\ R_y(\theta),\ R_z(\theta)$
- *Translate* $T_x(d), T_y(d), T_z(d)$
- *Shear* $Sh_x(d),\ Sh_y(d),\ Sh_z(d)$

Spring 2009 Utah School of Computing 101

3D Scale in x

$$S_x(\lambda) = \begin{bmatrix} \lambda & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

homog. Coord

Spring 2009 Utah School of Computing 102

3D Scale in x

$$S_x = \begin{bmatrix} \lambda & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} \lambda x \\ y \\ z \\ 1 \end{bmatrix}$$

3D Scale in y

$$S_y(\lambda) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ \lambda y \\ z \\ 1 \end{bmatrix}$$

3D Scale in z

$$S_z(\lambda) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \lambda & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ \lambda z \\ 1 \end{bmatrix}$$

Spring 2009 Utah School of Computing 105

Overall 3D Scale

$$S(\lambda) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \left(1/\lambda\right) \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ \left(1/\lambda\right) \end{bmatrix}$$

Spring 2009 Utah School of Computing 106

Overall 3*D Scale*

Same in *x,y* and *z:*

$$\begin{bmatrix} x \\ y \\ z \\ \left(\frac{1}{\lambda}\right) \end{bmatrix} = \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda z \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda z \end{bmatrix}$$

Spring 2009 Utah School of Computing 107

Stop Here.

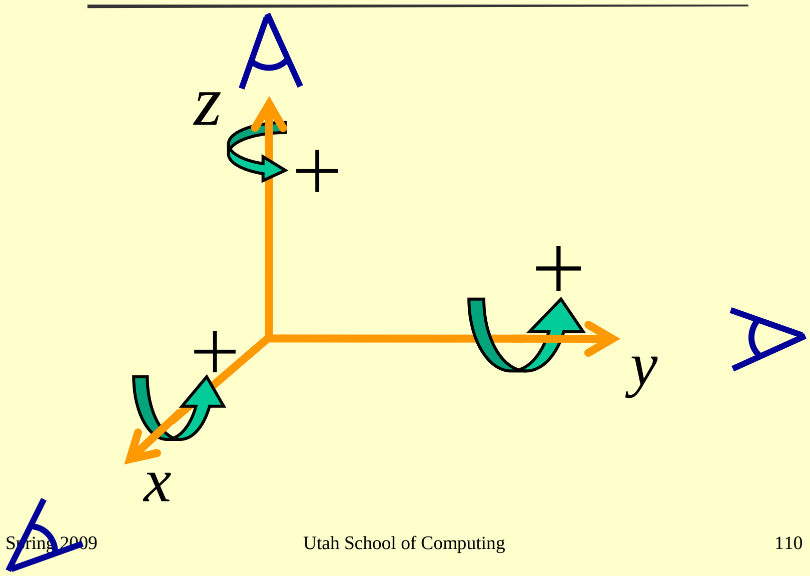
Spring 2009 Utah School of Computing 108

Positive *Rotation* in 3*D*?

- Sit at $+\infty$ end of given axis
- Look at Origin
- CC Rotation is in *Positive* direction

Spring 2009 Utah School of Computing 109

3*D* Positive *Rotations*



Spring 2009 Utah School of Computing 110

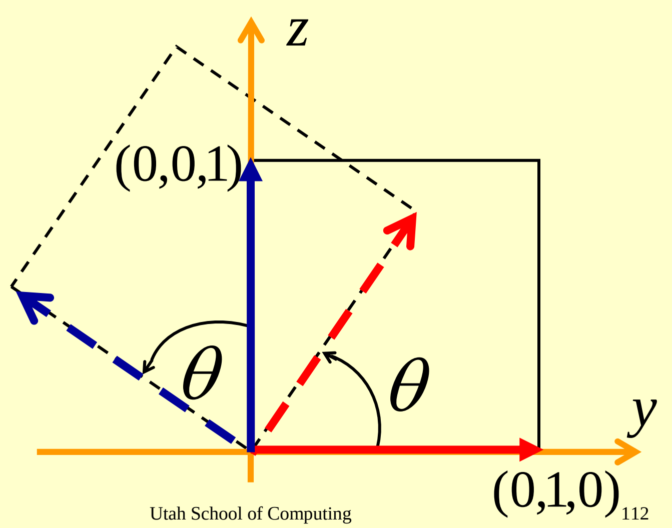
3*D Rotation* about *z-axis* by θ

We have already done this:

$$R_Z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Spring 2009 Utah School of Computing 111

3*D Rotation* about *x-axis* by θ



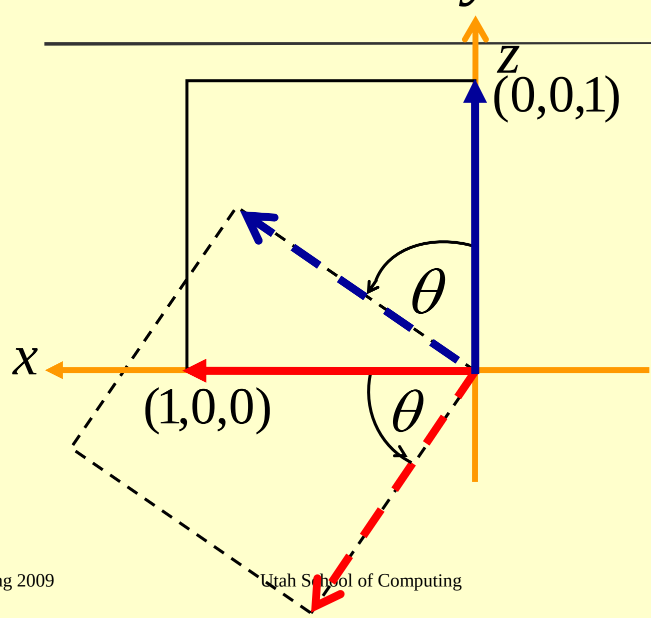
Spring 2009 Utah School of Computing 112

3D *Rotation* about *x-axis* by θ

$$R_X(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Spring 2009 Utah School of Computing 113

3D *Rotation* about *y-axis* by θ



Spring 2009 Utah School of Computing 114

3D *Rotation* about *y-axis* by θ

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Elementary Transformations

- *Scale*: $S_{\lambda_x}(v),\ S_{\lambda_y}(v)$
- *Rotate*: $R_{\theta_x}(v),\ R_{\theta_y}(v)$
- *Translate*: $T_{d_x}(v),\ T_{d_y}(v)$
- *Shear*: $Sh_{\lambda_x}(v),\ Sh_{\lambda_y}(v)$
- *Reflect*: $Rf_x(v),\ Rf_y(v)$

The End

Transformations I

Lecture Set 5