Transformations I

CS5600 Computer Graphics

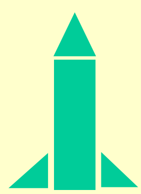
From: Rich Riesenfeld
Spring 2013

Transformations and Matrices

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

Rocket

How to form a rocket?



How to move a 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

$$\begin{tikzcd}
\vec{x}, \vec{y} \arrow[r, "T()"] \arrow[d, "a\times[\ ] + b\times[\ ]"'] & T(\vec{x}), T(\vec{y}) \arrow[d, "a\times[\ ] + b\times[\ ]"] \\
a\vec{x}, b\vec{y} \arrow[r, "T()"] & \begin{array}{l} aT(\vec{x}) + bT(\vec{y}) \\ = T(a\vec{x}) + T(b\vec{y}) \end{array}
\end{tikzcd}$$

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

Scale in x, by 2, say:

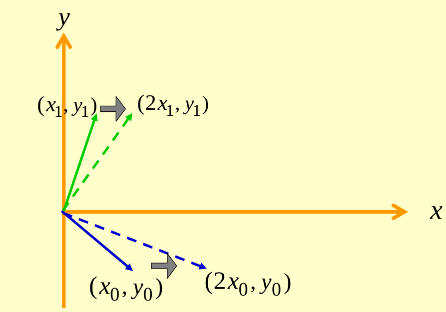
$$\begin{aligned}\big(2(x_0+x_1), y_0+y_1\big) &= \big(2x_0+2x_1, y_0+y_1\big) \\ &= \big(2x_0, y_0\big) + \big(2x_1, y_1\big)\end{aligned}$$

Example:

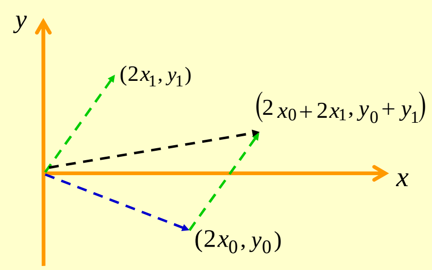
Scale in *x* by $S_{2_x}(v)$

What is the graphical view?

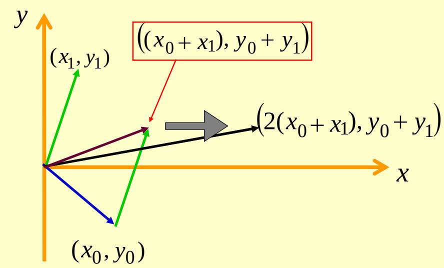
Scale in *x* by 2: $S_{2_x}(v)$



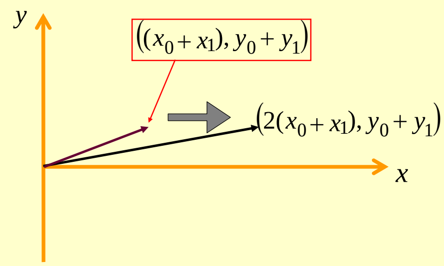
$\left(2x_0 + 2x_1, y_0 + y_1\right)$



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

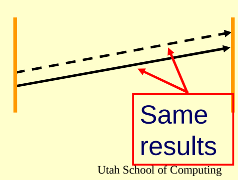


$\left(2(x_0 + x_1), y_0 + y_1\right)$



Summary on *Scale*

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



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: $S_{2_y}(v)$

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

Matrix Representation $S_2(v)$

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

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

Matrix Form of Same

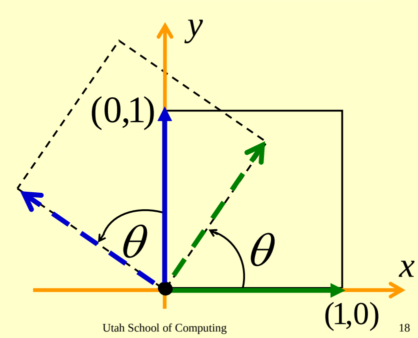
$$\underbrace{\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x_0 + x_1 \\ y_0 + y_1 \end{bmatrix}}_{\text{Add } x \text{ and } y\text{, then scale}} = \begin{bmatrix} 2\left(x_0 + x_1\right) \\ y_0 + y_1 \end{bmatrix}$$

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

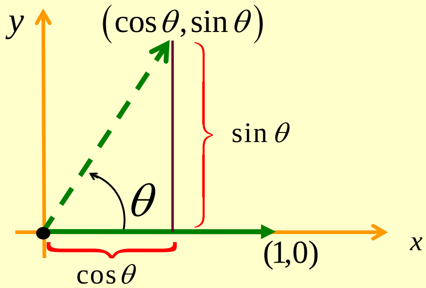
What about Rotation?

Is it linear?

Rotate by θ



Rotate by θ: 1^{st} *Quadrant*



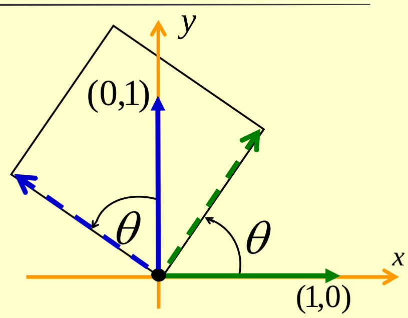
Spring 2013 Utah School of Computing 19

Rotate by θ: 1^{st} *Quadrant*

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

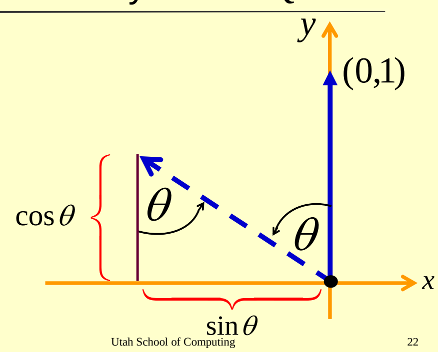
Spring 2013 Utah School of Computing 20

Rotate by θ: 2^{nd} *Quadrant*



Spring 2013 Utah School of Computing 21

Rotate by θ: 2^{nd} *Quadrant*



Spring 2013 Utah School of Computing 22

Rotate by θ: 2^{nd} *Quadrant*

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

Spring 2013 Utah School of Computing 23

Summary of *Rotation by* θ

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

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

Spring 2013 Utah School of Computing 24

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

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

What do the off diagonal elements do?

Off Diagonal Elements

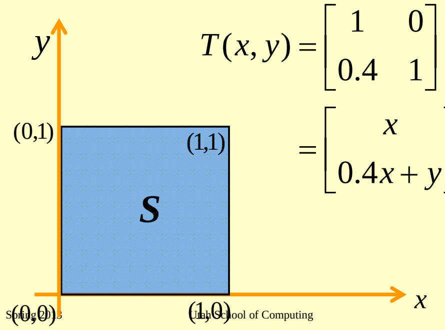
$$\begin{bmatrix} 1 & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + a\,y \\ y \end{bmatrix}$$

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

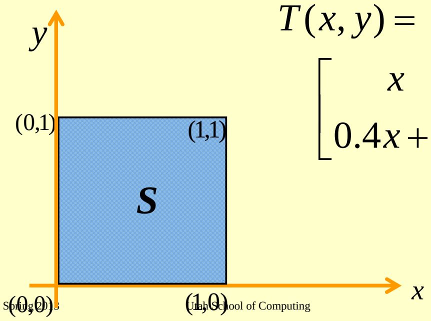
Example 1

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

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



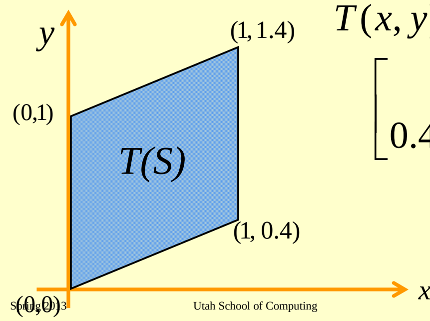
Example 1



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

Spring 2013 Utah School of Computing 31

Example 1



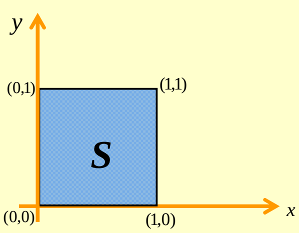
$$T(x,y) = \begin{bmatrix} x \\ 0.4x + y \end{bmatrix}$$

Spring 2013 Utah School of Computing 32

Example 2

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

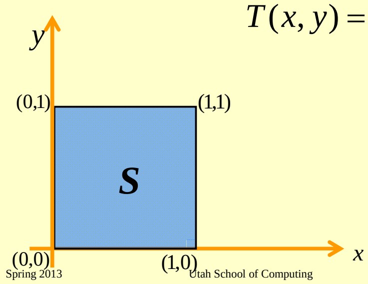
$$= \begin{bmatrix} x + 0.6y \\ y \end{bmatrix}$$



Spring 2013 Utah School of Computing 33

Example 2

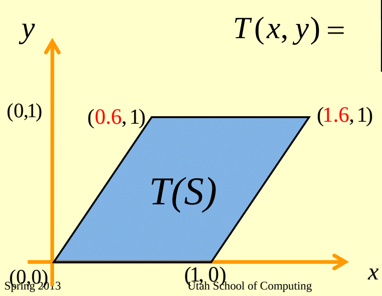
$$T(x,y) = \begin{bmatrix} x + 0.6y \\ y \end{bmatrix}$$



Spring 2013 Utah School of Computing 34

Example 2

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



Spring 2013 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 2013 Utah School of Computing 36

Double Shear: not commutative!

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

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

Spring 2013 Utah School of Computing 37

"Lazy 1"

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

Spring 2013 Utah School of Computing 38

Translation in x

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

Spring 2013 Utah School of Computing 39

Translation in y

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

Spring 2013 Utah School of Computing 40

Homogeneous Coordinates

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

Spring 2013 Utah School of Computing 41

Homogeneous Coordinates

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

Spring 2013 Utah School of Computing 42

Homogeneous Coordinates

For $\lambda \neq 0$,

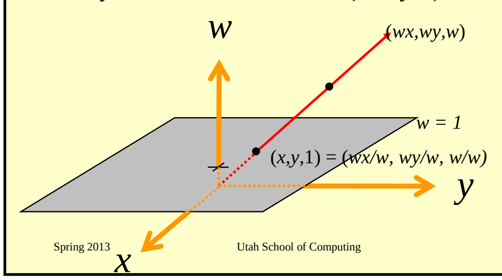
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1/\lambda \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1/\lambda \end{bmatrix} = \begin{bmatrix} \lambda x \\ \lambda y \\ 1 \end{bmatrix} \Leftrightarrow \begin{bmatrix} \lambda x \\ \lambda y \end{bmatrix}$$

Homogeneous term affects overall scaling

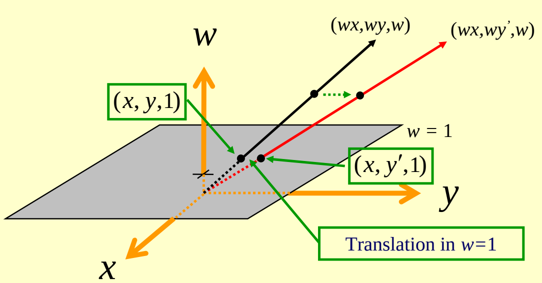
Homogeneous Coordinates

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

They constitute the whole line (wx,wy,w).



What does a *shear* do?



Using Homogeneous Coord's

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

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

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

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

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

Elementary Transformations

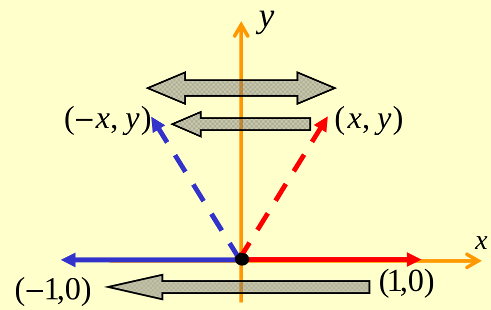
- *Scale*: $S_{\lambda_x}(v),\ S_{\lambda_y}(v)$
- *Rotate*: $R_{\theta_x}(v),\ R_{\theta_y}(v)$
- *Translate*: $T_{d_x}(v),\ T_{d_y}(v)$
- *Shear*: $Sh_{\lambda_x}(v),\ Sh_{\lambda_y}(v)$
- *Reflect*: $Rf_x(v),\ Rf_y(v)$

Refection about *y-axis*

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

$$\boxed{x \leftrightarrow -x}$$

Reflection about *y-axis*

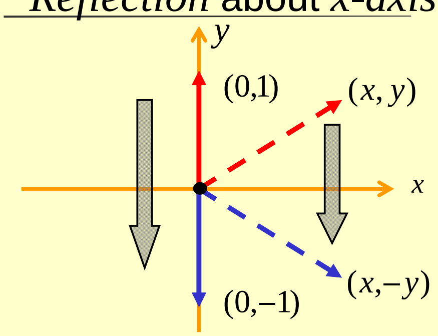


Reflection about *x-axis*

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

$$\boxed{y \leftrightarrow -y}$$

Reflection about *x-axis*

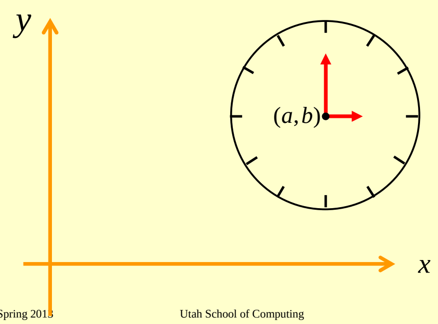


Is *Reflection* *"Elementary"* ?

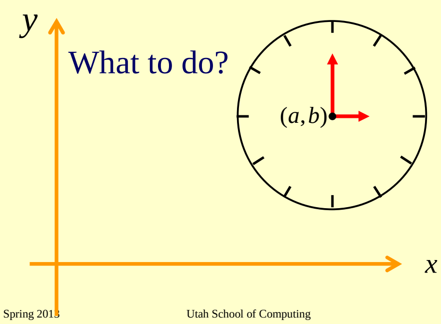
- Can we effect reflection in an elementary way?
- (More elementary means scale, shear, rotation, translation.)

Reflection = *Scale* (-1)

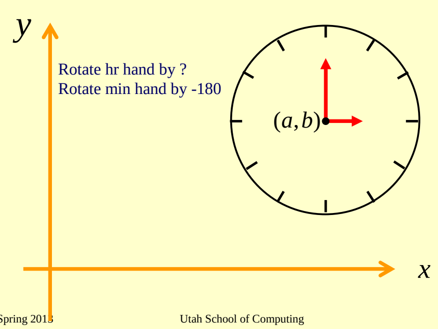
Ex: Advance clock hands



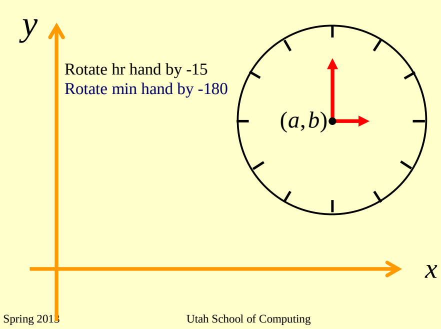
Ex: Advance clock hands: 30mins

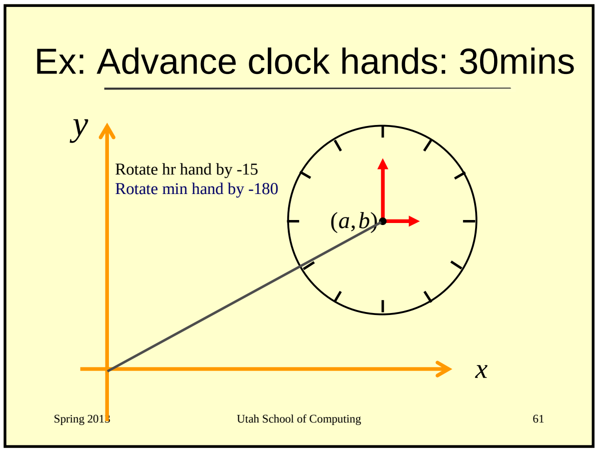


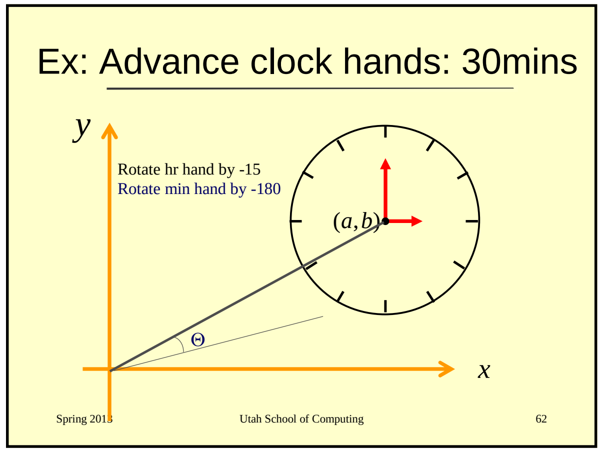
Ex: Advance clock hands: 30mins

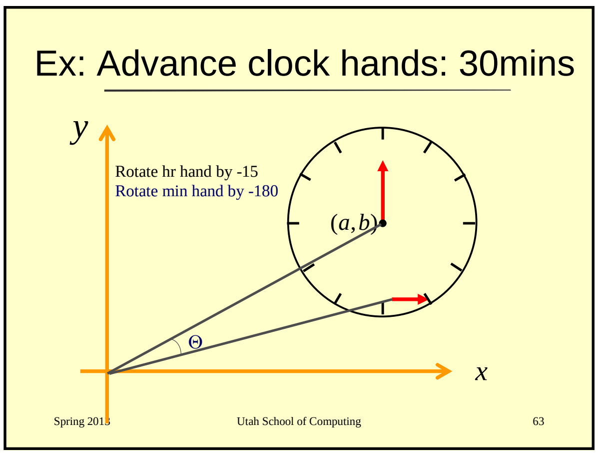


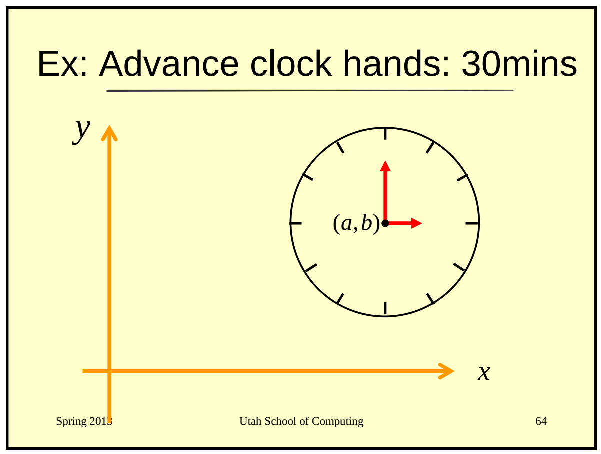
Ex: Advance clock hands: 30mins

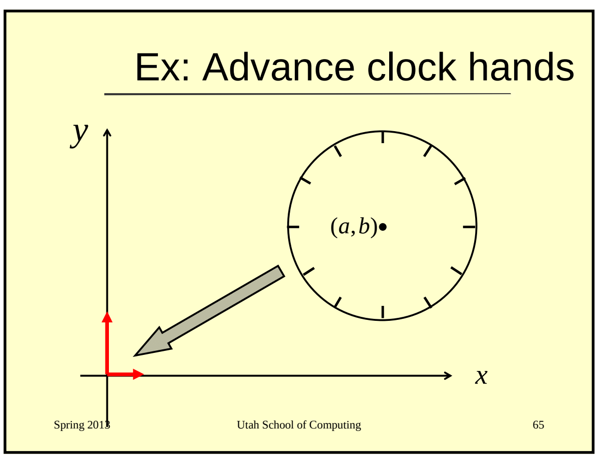


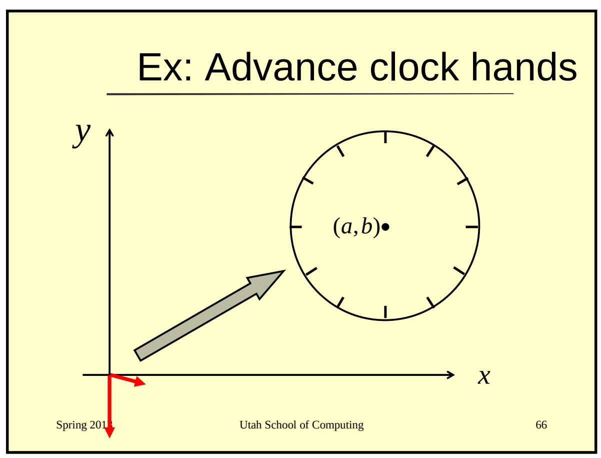

Ex: Advance clock hands: 30mins
y
Rotate hr hand by -15
Rotate min hand by -180
(a,b)
x
Spring 2013
Utah School of Computing
61


Ex: Advance clock hands: 30mins
y
Rotate hr hand by -15
Rotate min hand by -180
(a,b)
Θ
x
Spring 2013
Utah School of Computing
62

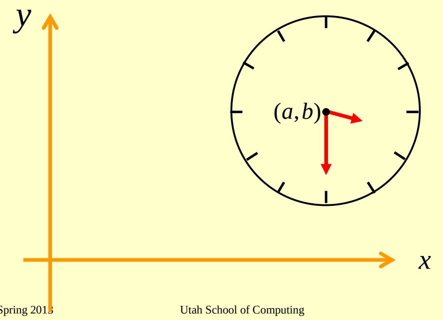

Ex: Advance clock hands: 30mins
y
Rotate hr hand by -15
Rotate min hand by -180
(a,b)
Θ
x
Spring 2013
Utah School of Computing
63


Ex: Advance clock hands: 30mins
y
(a,b)
x
Spring 2013
Utah School of Computing
64

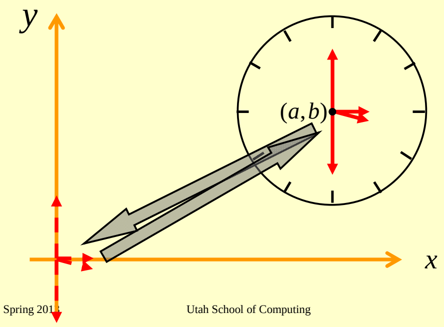

Ex: Advance clock hands
y
(a,b)
x
Spring 2013
Utah School of Computing
65


Ex: Advance clock hands
y
(a,b)
x
Spring 2013
Utah School of Computing
66

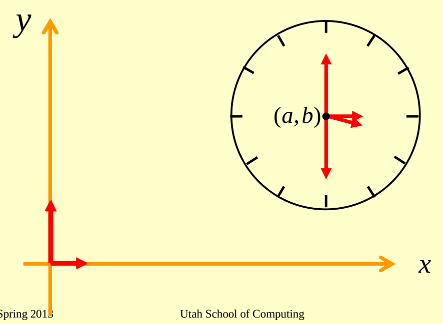
Ex: Advance clock hands



Ex: Advance clock hands



Ex: Advance clock hands



Clock *Transformations*

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

Clock Transformations

$$T_S = T(a,b)\,R(t)\,T(-a,-b)$$

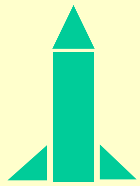
$$T_b = T(a,b)\,R(12 * t)\,T(-a,-b)$$

$$\text{where} \quad t = -15^\circ$$

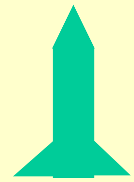
Clock Transformations

$$\underbrace{\begin{bmatrix}1&0&a\\0&1&b\\0&0&1\end{bmatrix}}_{\text{Translate Back}}\underbrace{\begin{bmatrix}\cos(t)&-\sin(t)&0\\\sin(t)&\cos(t)&0\\0&0&1\end{bmatrix}}_{\text{Rotate About Origin}}\underbrace{\begin{bmatrix}1&0&-a\\0&1&-b\\0&0&1\end{bmatrix}}_{\text{Translate to Origin}}\begin{bmatrix}x\\y\\1\end{bmatrix} = \begin{bmatrix}x'\\y'\\1\end{bmatrix}$$

Rocket revisited

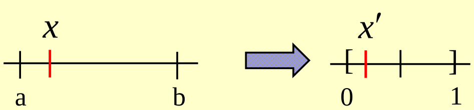


Rocket revisited



How to move a rocket?

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



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

- Translate to Origin

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

- Map x to translated interval

$$x \to x-a$$

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

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

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

- Normalize the interval

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

- Map x to normalized interval

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

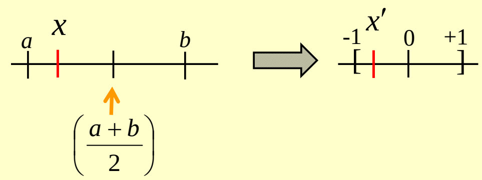
Just Look at $\Upsilon^{[1]}$

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

This is a homogeneous form for 1D

Spring 2013 Utah School of Computing 79

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



Spring 2013 Utah School of Computing 80

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 2013 Utah School of Computing 81

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 2013 Utah School of Computing 82

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

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

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

Spring 2013 Utah School of Computing 83

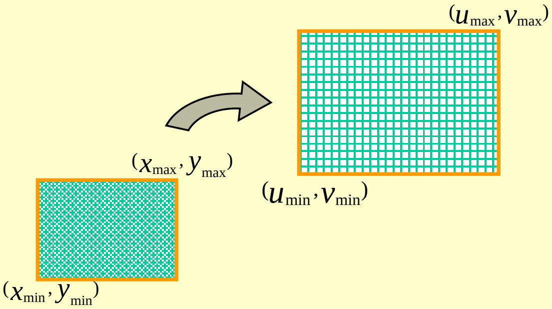
All Together: Map: $[a,b] \Rightarrow [c,d\,]$

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

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

Spring 2013 Utah School of Computing 84

Now Map Rectangles



Transformation in x and y

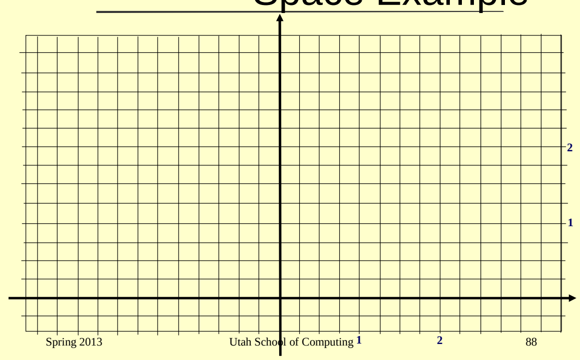
$$\begin{bmatrix} 1 & 0 & u_{min} \\ 0 & 1 & v_{min} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda_x & 0 & 0 \\ 0 & \lambda_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_{min} \\ 0 & 1 & -y_{min} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

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

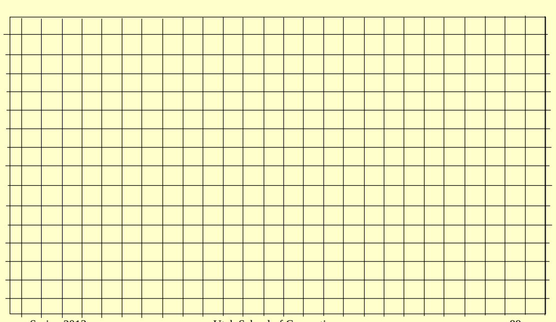
This is *Viewport Transformation*

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

Space Example



Space Example



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* $Sh_x(d),\ Sh_y(d),\ Sh_z(d)$

3D Scale in x

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

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

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

Overall 3D Scale

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

Overall 3D Scale

Same in x, y and z:

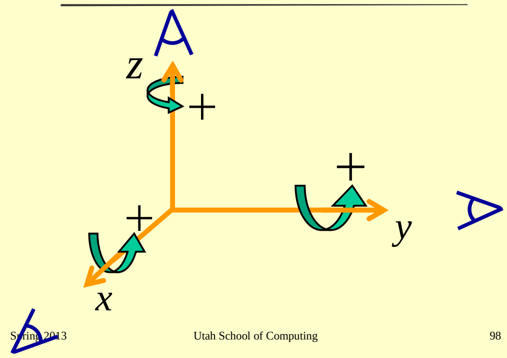
$$\begin{bmatrix} x \\ y \\ z \\ (1/\lambda) \end{bmatrix} = \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda z \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} \lambda x \\ \lambda y \\ \lambda z \end{bmatrix}$$

Positive *Rotation* in $3D$?

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

Spring 2013 Utah School of Computing 97

$3D$ Positive *Rotations*



Spring 2013 Utah School of Computing 98

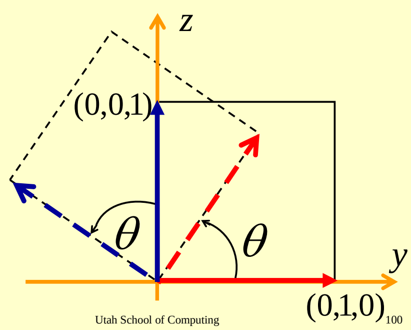
$3D$ *Rotation* about *z-axis* by θ

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 2013 Utah School of Computing 99

$3D$ *Rotation* about *x-axis* by θ



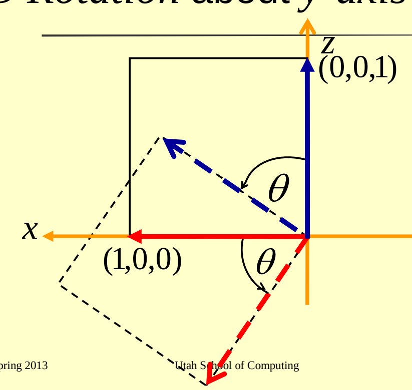
Spring 2013 Utah School of Computing 100

$3D$ *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 2013 Utah School of Computing 101

$3D$ *Rotation* about *y-axis* by θ



Spring 2013 Utah School of Computing 102

3D *Rotation* about *y-axis* by θ

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

Spring 2013 Utah School of Computing 103

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*: $Rf_x(v),\ Rf_y(v)$

Spring 2013 Utah School of Computing 104

The End

Transformations I

Lecture Set 5