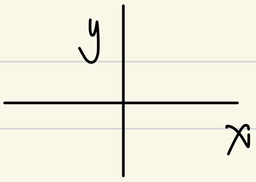
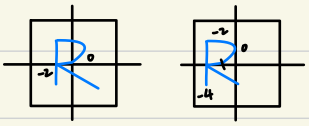
geometric operation
stay in euclidean coordinates



coords of new image
$J(x,y) = I(T(x,y))$ geometric
coords of old vs
$J(x,y) = T(I(x,y))$ point operation

eg: $J(x,y) = I(x+2,y) \Rightarrow J(0,0) = I(2,0)$
$J(-2,0) = I(0,0)$



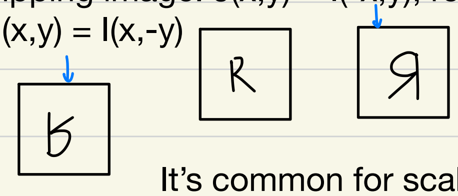
translate left by 2 pixels

$J(x,y) = I(x, y-10)$ translate up by 10 pixels
new old

scaling eg: $J(x,y) = I(2x, 2y)$ shrink down, smaller

$J(x,y) = I(\frac{1}{2}x, \frac{1}{2}y)$ shrink up, larger

flipping image: $J(x,y) = I(-x,y)$, reflect image across y axis
$J(x,y) = I(x,-y)$



It's common for scale+shift+flip to be combined into a 2D linear transformation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cdot & \cdot \\ \cdot & \cdot \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \cdot \\ \cdot \end{bmatrix}$$

coord of transformed image $J(x,y)$
coord of original image $I(x,y)$

where does (x,y) in the old image go to? (forward mapping)
translation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

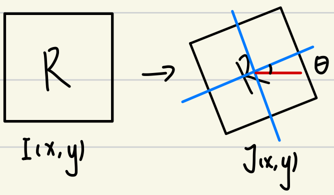
Scale

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \alpha & 0 \\ 0 & \beta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

flip

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

we can do things to image that we can't do to 1-D signals
eg: rotation, rotate by θ counterclockwise around origin



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

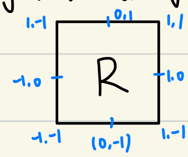
any combination of scale, shift, rotate is called a similarity transformation

preserve parallel lines

if α, β = ±1, isometric transformation (rigid motion): preserve shape, angle

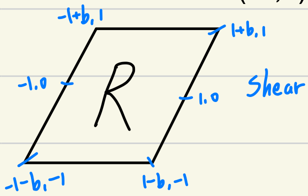
we can also "bend" the image

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

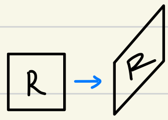


$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x+by \\ y \end{bmatrix}$$

(0,0)=(0,0) (0,1)=(b,1)
(1,0)=(1,0) (1,1)=(1+b,1)
(-1,0)=(-1,0)
(-1,1)=(-1+b,1)

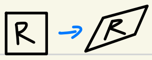


$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ c & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$



vertical shear

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

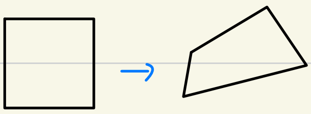


rectangle -> parallelogram

a transform of the form

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

is called affine transformation



rectangle -> quadrilateral,
projective transformation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \dfrac{a_{11}x + a_{12}y + b_1}{c_1x + c_2y + 1} \\ \dfrac{a_{21}x + a_{22}y + b_2}{c_1x + c_2y + 1} \end{bmatrix}$$

$\Downarrow$

$$\begin{bmatrix} \tilde{x}' \\ \tilde{y}' \\ \tilde{z}' \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \qquad x' = \frac{\tilde{x}'}{\tilde{z}'} \quad y' = \frac{\tilde{y}'}{\tilde{z}'}$$

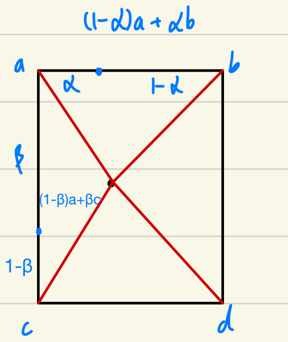
how to actually create the output image

how to get image colors/intensity on the new grid.
It's more conventional to use backward mapping instead of afterward mapping

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underset{2\times2}{A} \begin{bmatrix} x \\ y \end{bmatrix} + \underset{2\times1}{b} \qquad \begin{bmatrix} x \\ y \end{bmatrix} = A^{-1}\left(\begin{bmatrix} x' \\ y' \end{bmatrix} - b\right)$$

$$= A^{-1}\begin{bmatrix} x' \\ y' \end{bmatrix} - A^{-1}b$$

inverse transformation



$\chi = [(1-\beta)a + \beta c](1-\alpha) + [(1-\beta)b+\beta d]\alpha$
$=(1-\alpha)(1-\beta)a$
$+(1-\beta)\alpha * b$
$+(1-\alpha)\beta * c$
$+\alpha\beta * d$ bilinear interpolation

bicubic interpolation uses more points; and look smoother