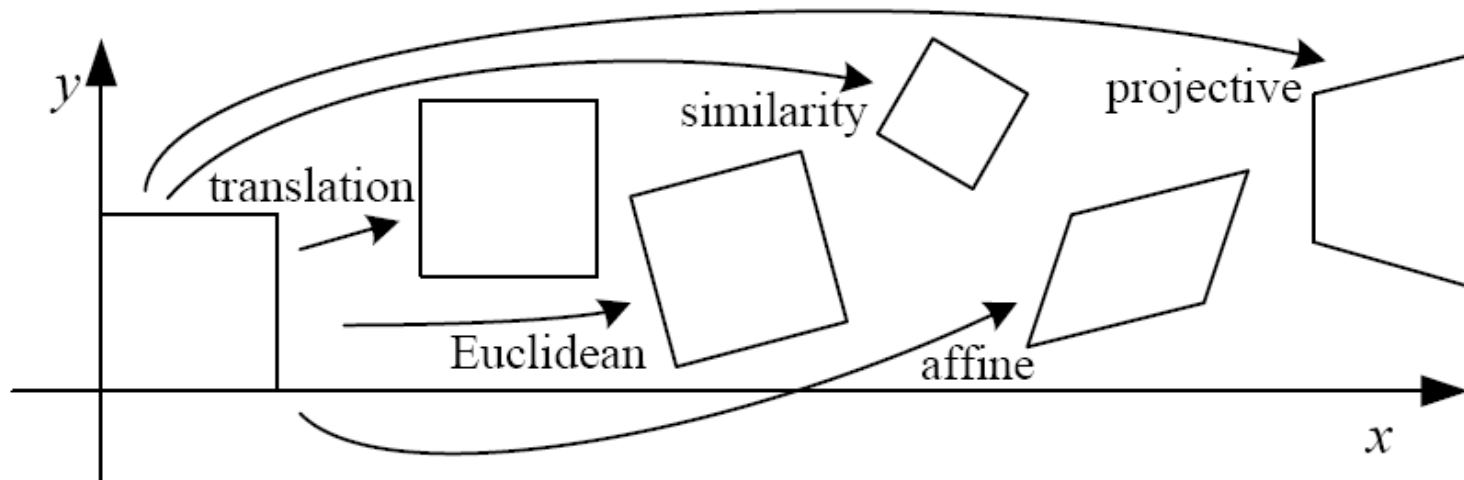# CS5670: Computer Vision
Noah Snavely

Lecture 7: Transformations and warping

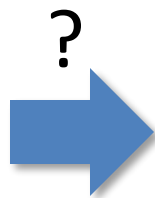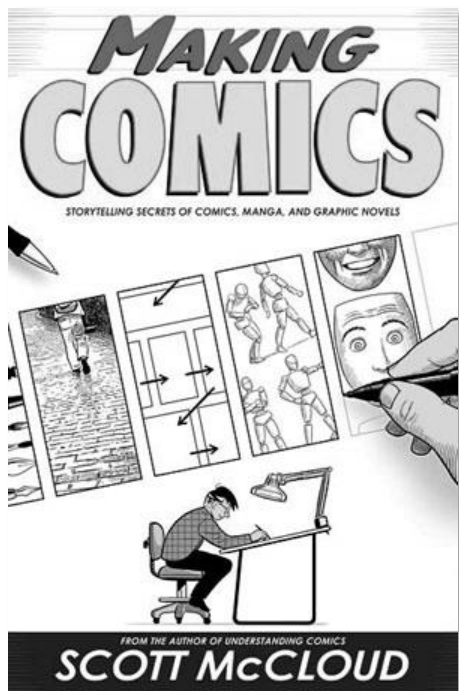# Reading

- Szeliski: Chapter 3.6

Announcements

- Project 1 (Hybrid Images) code due this Wednesday, Feb 14, by 11:59pm

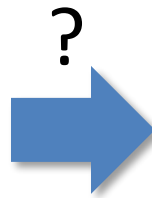- Artifacts due Friday, Feb 16, by 11:59pm

Image alignment

Why don't these image line up exactly?

What is the geometric relationship between these two images?

?

Answer: Similarity transformation (translation, rotation, uniform scale)

What is the geometric relationship between these two images?

?

What is the geometric relationship between these two images?
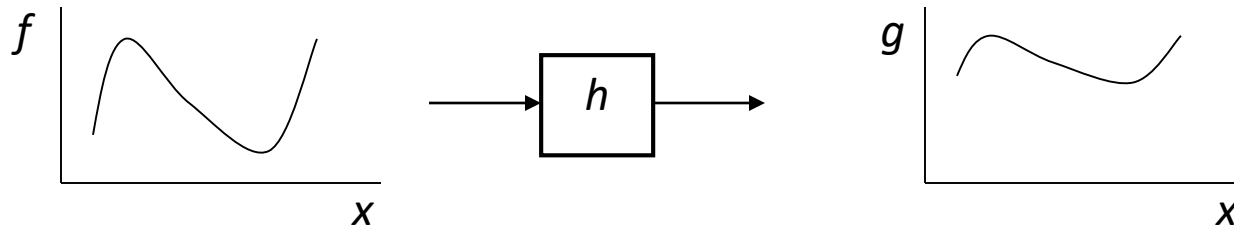

Very important for creating mosaics!

Image Warping

- image filtering: change *range* of image
  - *g(x) = h(f(x))*

- image warping: change *domain* of image
  - *g(x) = f(h(x))*

Image Warping

- image filtering: change *range* of image
  - *g(x) = h(f(x))*

f *h* *g*

- image warping: change *domain* of image
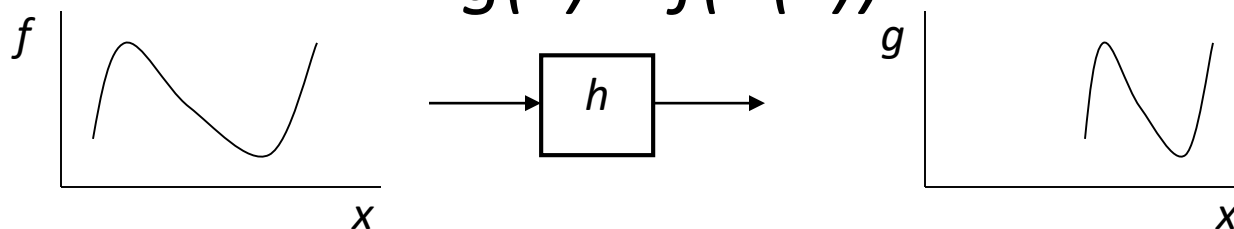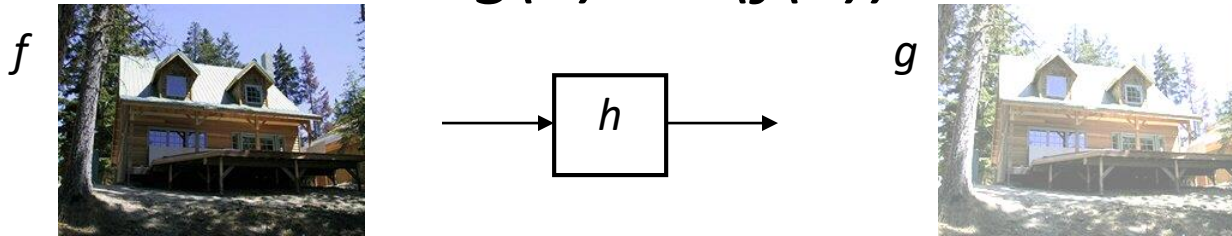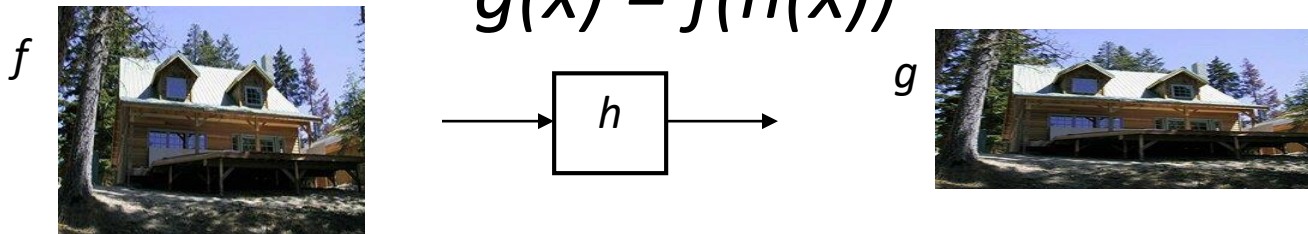 - *g(x) = f(h(x))*

f *h* *g*

Parametric (global) warping

- Examples of parametric warps:

translation

rotation

aspect

Parametric (global) warping
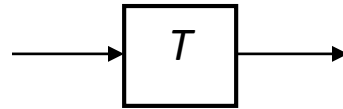
p = (x,y)   **p'** = (x',y')

- Transformation T is a coordinate-changing machine:

$$p' = T(p)$$

- What does it mean that *T* is global?
  - Is the same for any point p
  - can be described by just a few numbers (parameters)
- Let's consider *linear* xforms (can be represented by a 2D matrix):

$$\mathbf{p}' = \mathbf{T}\mathbf{p} \qquad \begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \end{bmatrix}$$

Common linear transformations

- Uniform scaling by s:

(0,0) ●

(0,0) ●

$$\mathbf{S} = \left[\begin{array}{cc} s & 0 \\ 0 & s \end{array} \right]$$

What is the inverse?

Common linear transformations

- Rotation by angle θ (about the origin)

(0,0)

(0,0)

θ
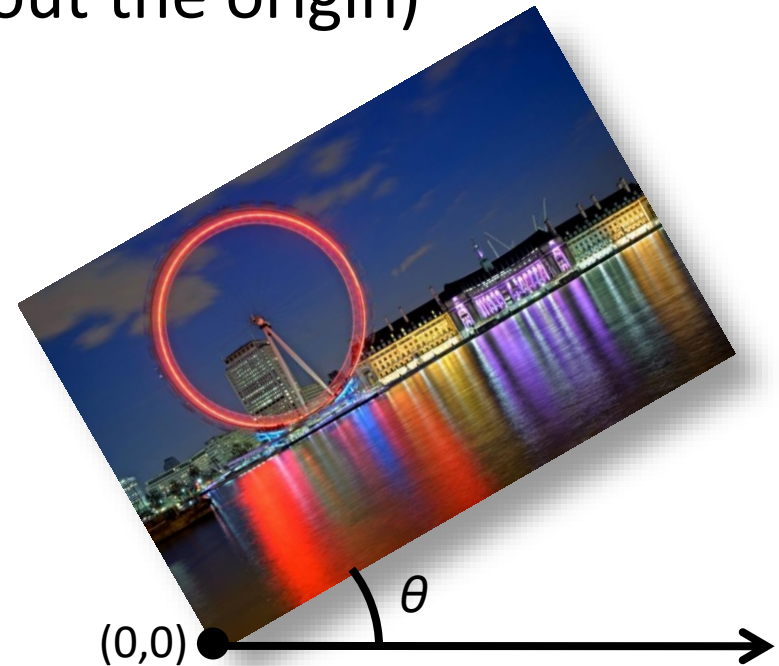
$$\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

What is the inverse?

For rotations:

$$\mathbf{R}^{-1} = \mathbf{R}^T$$

2x2 Matrices

- What types of transformations can be represented with a 2x2 matrix?

2D mirror about Y axis?

$$x' = -x$$
$$y' = y$$

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

2D mirror across line y = x?

$$x' = y$$
$$y' = x$$

$$\mathbf{T} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

2x2 Matrices

- What types of transformations can be represented with a 2x2 matrix?

2D Translation?

$$x' = x + t_x$$ NO!

$$y' = y + t_y$$

Translation is not a linear operation on 2D coordinates

All 2D Linear Transformations

- Linear transformations are combinations of …
  - Scale,
  - Rotation,
  - Shear, and
  - Mirror

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

- Properties of linear transformations:
  - Origin maps to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved
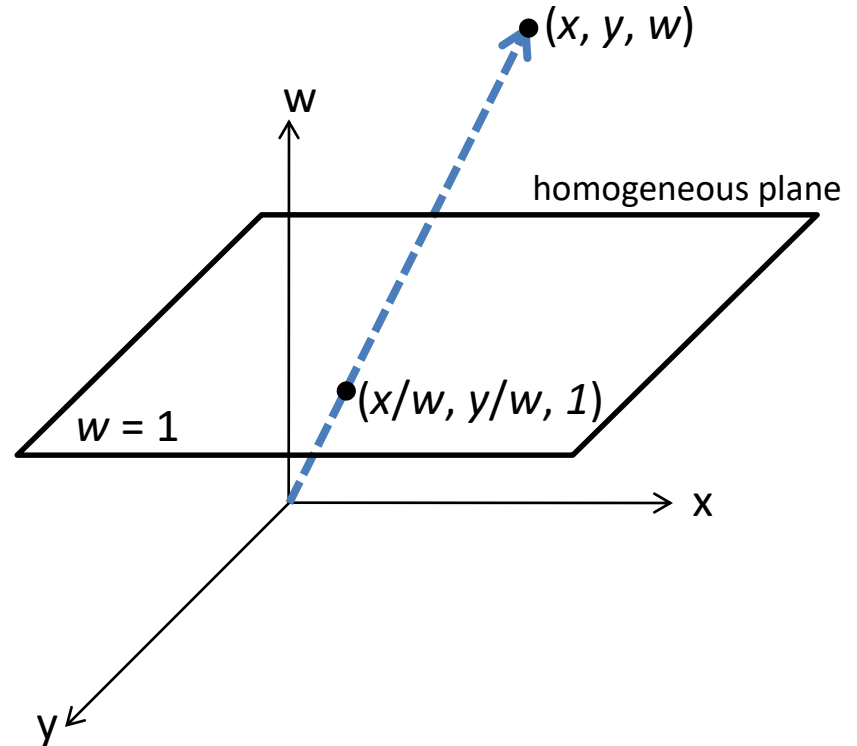 - Closed under composition

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} i & j \\ k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Homogeneous coordinates

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image
coordinates

Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

Translation

- Solution: homogeneous coordinates to the rescue

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

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

Affine transformations

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

any transformation represented by a 3x3 matrix with last row [0 0 1] we call an *affine* transformation

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$$

Basic affine transformations

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

Translate

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

Scale

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

2D *in-plane* rotation

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

Shear

Affine Transformations

- Affine transformations are combinations of …
  - Linear transformations, and
  - Translations

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$
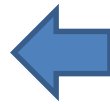
- Properties of affine transformations:
  - Origin does not necessarily map to origin
  - Lines map to lines
  - Parallel lines remain parallel
  - Ratios are preserved
  - Closed under composition

Is this an affine transformation?

Where do we go from here?

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix}$$

← what happens when we mess with this row?

affine transformation

Projective Transformations aka Homographies aka Planar Perspective Maps

$$\mathbf{H} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$$

Called a *homography*
(or *planar perspective map*)

Homographies

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\sim \begin{bmatrix} \frac{ax+by+c}{gx+hy+1} \\ \frac{dx+ey+f}{gx+hy+1} \\ 1 \end{bmatrix}$$
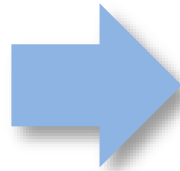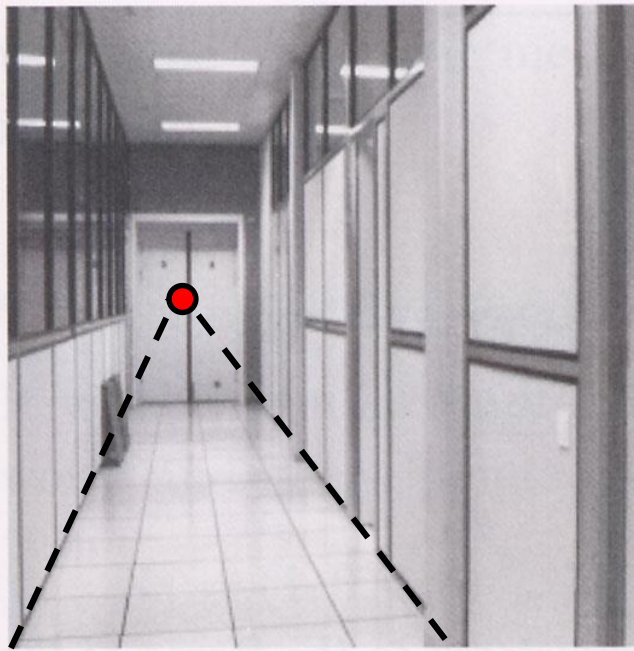
What happens when the denominator is 0?

Points at infinity

Image warping with homographies

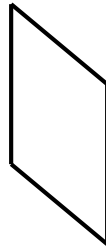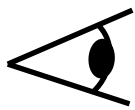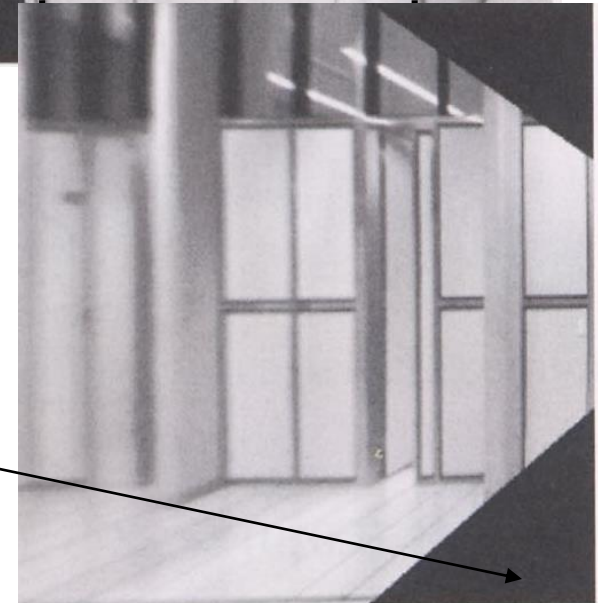$\mathbf{H}_1$
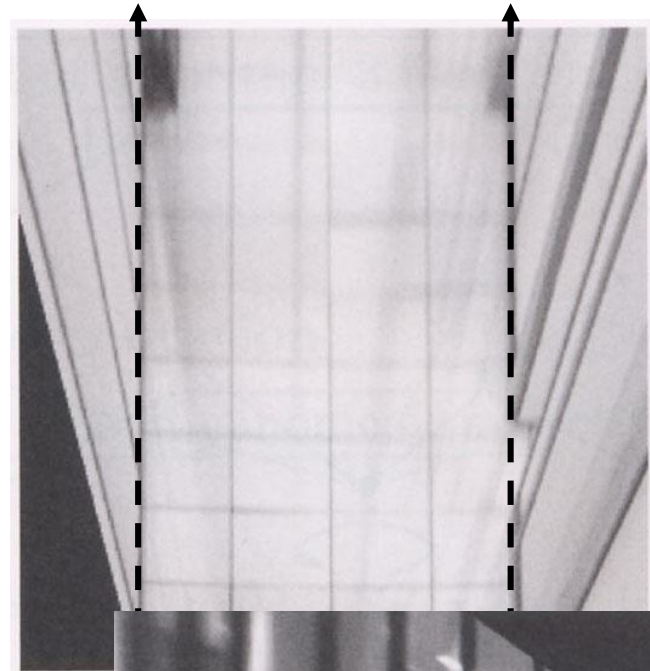
$\mathbf{H}_2$

image plane in front

black area where no pixel maps to

Homographies

Homographies

- Homographies …
  - Affine transformations, and
  - Projective warps

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Properties of projective transformations:
  - Origin does not necessarily map to origin
  - Lines map to lines
  - Parallel lines do not necessarily remain parallel
  - Ratios are not preserved
  - Closed under composition

Alternate formulation for homographies

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

where the length of the vector $[h_{00}\ h_{01} \ldots h_{22}]$ is 1

2D image transformations
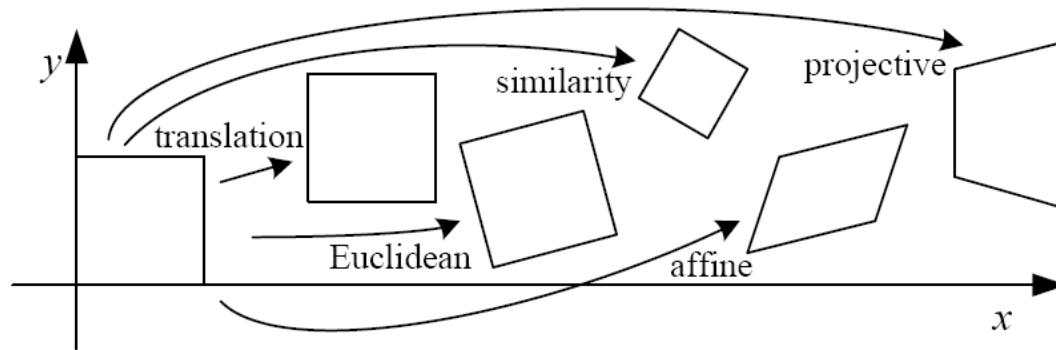
Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\left[\begin{array}{c|c} I & t \end{array}\right]_{2\times 3}$	2	orientation $+\cdots$	☐
rigid (Euclidean)	$\left[\begin{array}{c|c} R & t \end{array}\right]_{2\times 3}$	3	lengths $+\cdots$	◇
similarity	$\left[\begin{array}{c|c} sR & t \end{array}\right]_{2\times 3}$	4	angles $+\cdots$	◇
affine	$\left[\begin{array}{c} A \end{array}\right]_{2\times 3}$	6	parallelism $+\cdots$	▱
projective	$\left[\begin{array}{c} \tilde{H} \end{array}\right]_{3\times 3}$	8	straight lines	⏢

These transformations are a nested set of groups
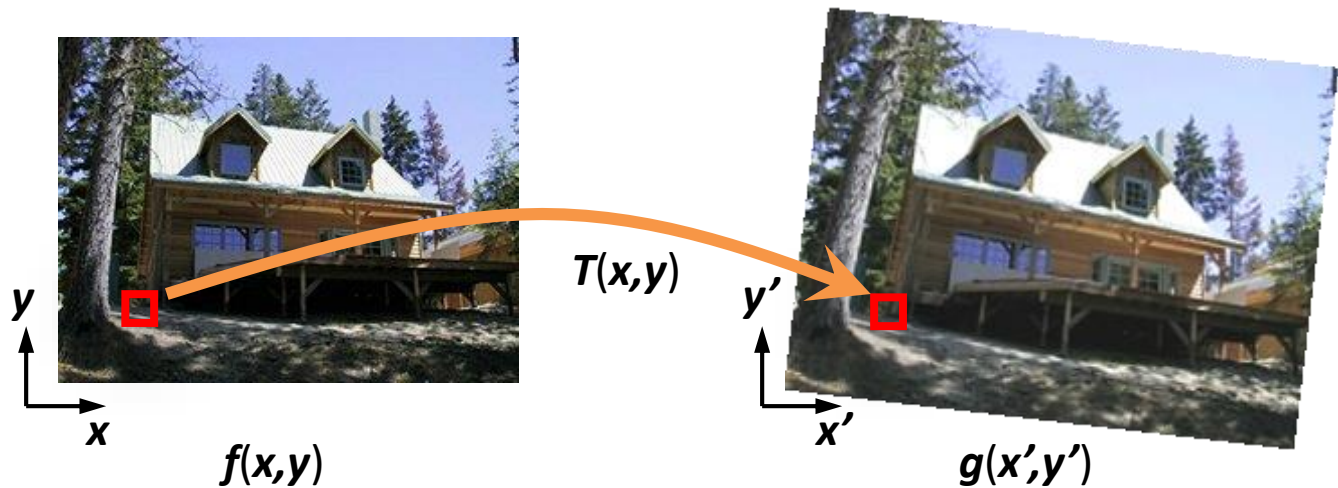- Closed under composition and inverse is a member

Implementing image warping

- Given a coordinate xform $(x',y') = T(x,y)$ and a source image $f(x,y)$, how do we compute an xformed image $g(x',y') = f(T(x,y))$?
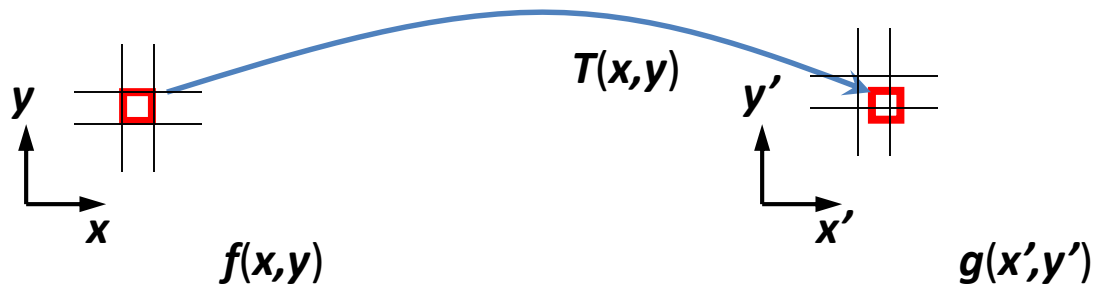

y

x

$f(x,y)$

$T(x,y)$

y'

x'

$g(x',y')$
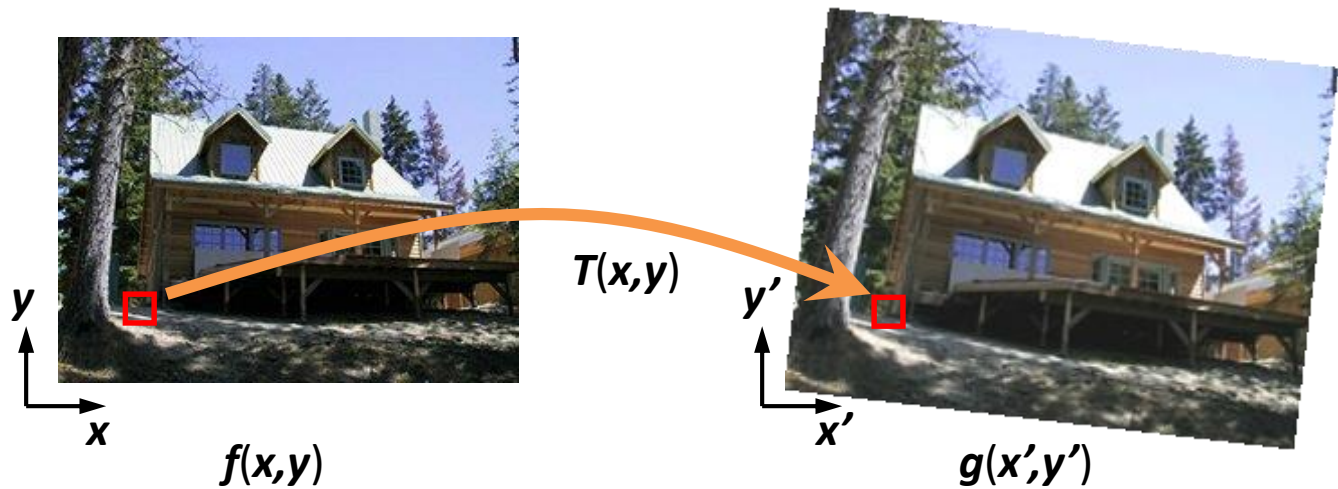
Forward Warping

- Send each pixel $f(x)$ to its corresponding location $(x',y') = T(x,y)$ in $g(x',y')$
  - What if pixel lands "between" two pixels?


$T(x,y)$

$f(x,y)$

$g(x',y')$

Forward Warping

- Send each pixel $f(x,y)$ to its corresponding location $x' = h(x,y)$ in $g(x',y')$
  - What if pixel lands "between" two pixels?
  - Answer: add "contribution" to several pixels, normalize later (*splatting*)
 - Can still result in holes

Inverse Warping

- Get each pixel *g*(*x',y'*) from its corresponding location (*x,y*) = *T*$^{-1}$(*x,y*) in *f*(*x,y*)
  - Requires taking the inverse of the transform
  - What if pixel comes from "between" two pixels?

T^{-1}(*x,y*)

f(*x,y*)   *g*(*x',y'*)
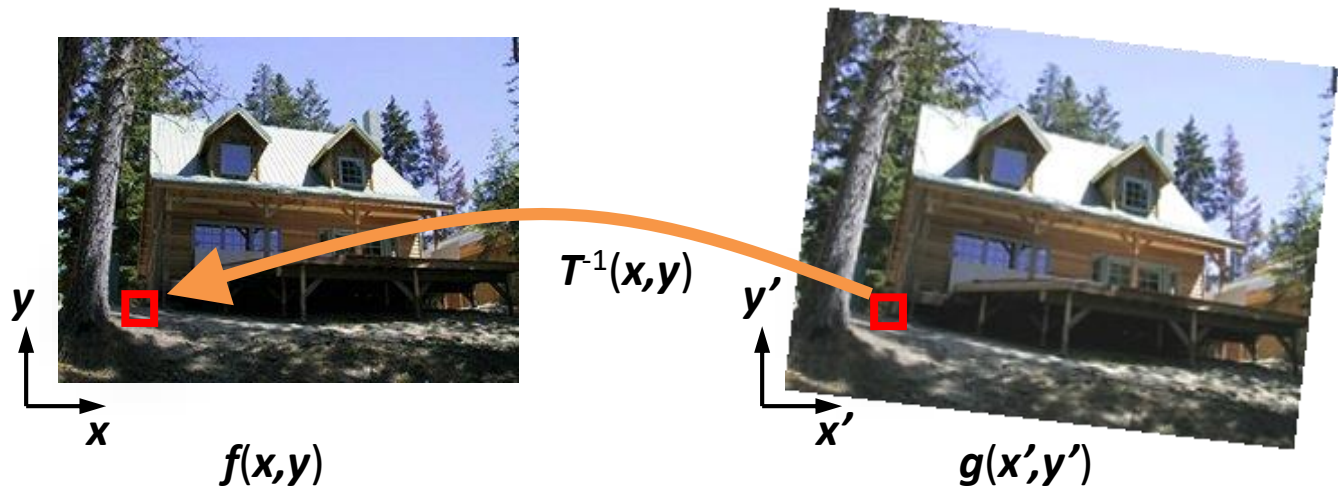
Inverse Warping

- Get each pixel $g(x')$ from its corresponding location $x' = h(x)$ in $f(x)$
  - What if pixel comes from "between" two pixels?
  - Answer: *resample* color value from *interpolated* (*prefiltered*) source image
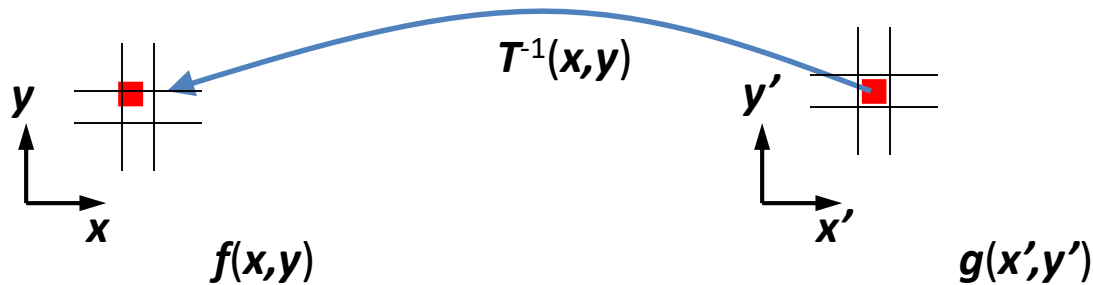
Interpolation

- Possible interpolation filters:
  - nearest neighbor
  - bilinear
  - bicubic
  - sinc

- Needed to prevent "jaggies" and "texture crawl"

  (with prefiltering)

Questions?