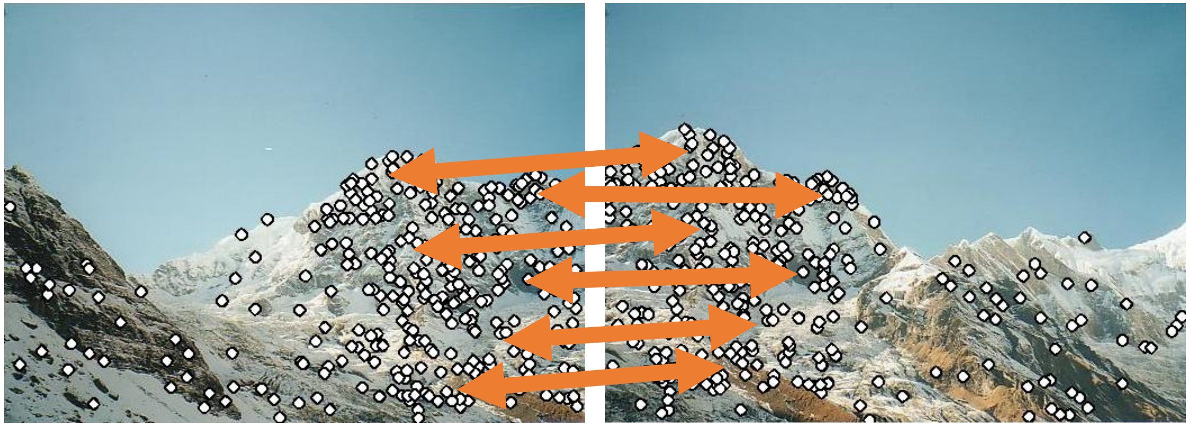
Lecture 12: Transformations and Fitting II

Administrative

HW2 Due Tomorrow, 2/19 at 11:59pm

HW3 is released, due a week from Friday, 2/28 at 11:59pm

So Far



1. How do we find distinctive / easy to locate features? *(Harris/Laplacian of Gaussian)*
2. How do we describe the regions around them? *(histogram of gradients)*
3. How do we match features? (L2 distance)
4. How do we handle outliers? (RANSAC)

Today

As promised: warping one image to another

Why Mosaic?

- Compact Camera FOV = 50 x 35°

Slide credit: Brown & Lowe

Why Mosaic?

- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°

Slide credit: Brown & Lowe

Why Mosaic?

- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°
- Panoramic Mosaic = 360 x 180°

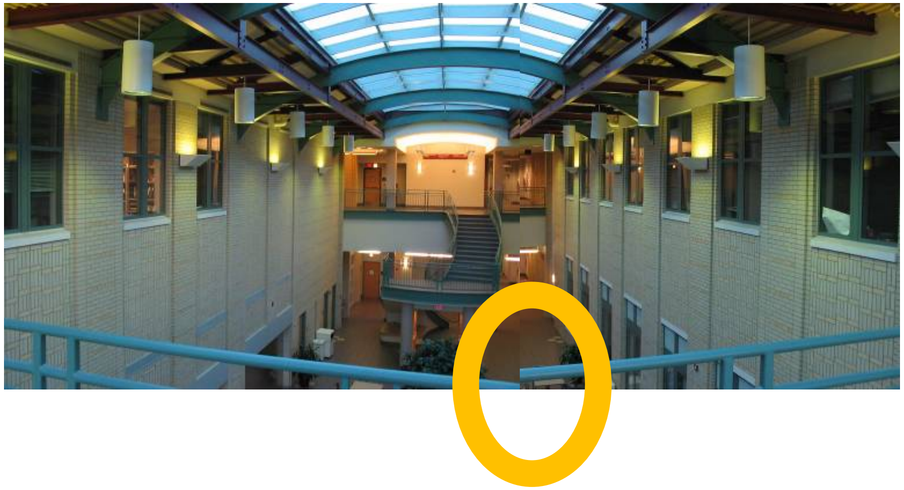
Slide credit: Brown & Lowe

Why Bother with the Math?

Slide credit: A. Efros

Homework 1 Style


Translation only via alignment




Slide credit: A. Efros

More Sophisticated Result

Slide credit: A. Efros

Today

Categories of Transformations

Fitting Transformations

Applying Transformations

Blending Images

Today

Categories of Transformations

Fitting Transformations

Applying Transformations

Blending Images

Image Transformations

Image filtering: change range of image

$$g(x) = T(f(x))$$

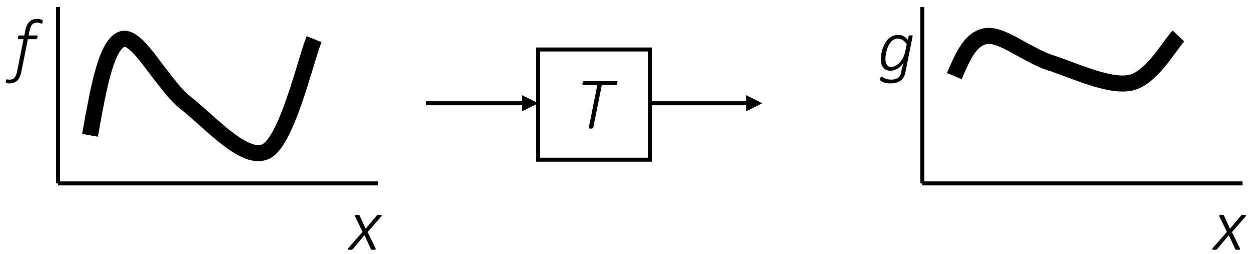
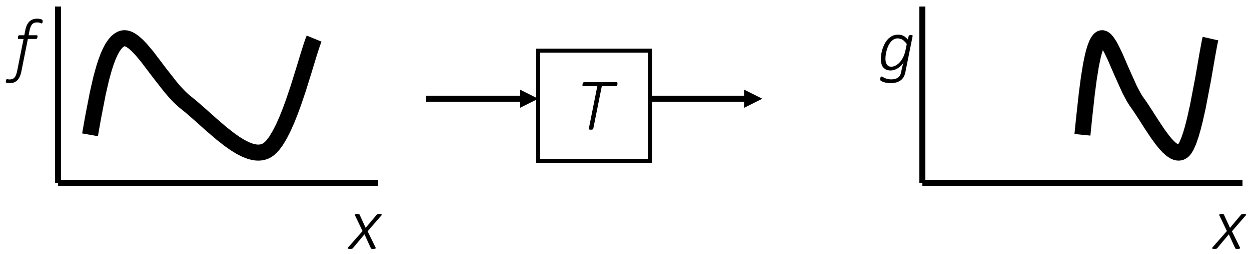


Image warping: change ***domain*** of image

$$g(x) = f(T(x))$$



Slide credit: A. Efros

Image Transformations

Image filtering: change range of image

$$g(x) = T(f(x))$$

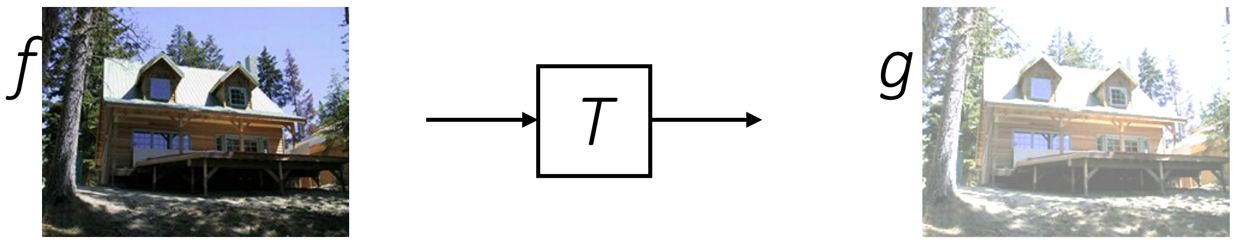
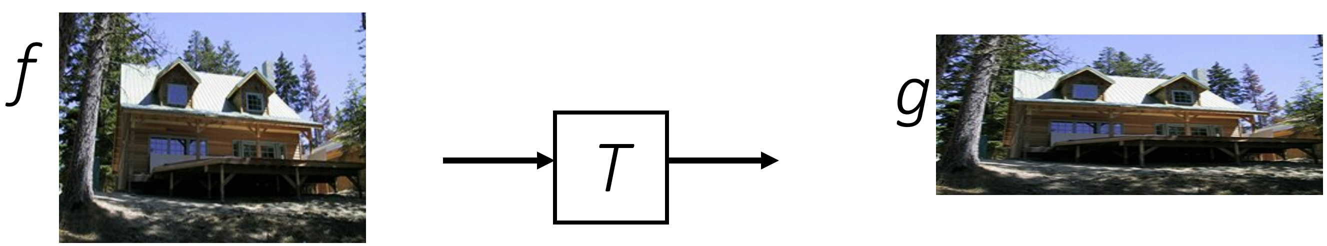


Image warping: change ***domain*** of image

$$g(x) = f(T(x))$$



Slide credit: A. Efros

Parametric (Global) Warping

Examples of parametric warps

translation

rotation

aspect

affine

perspective

cylindrical

Slide credit: A. Efros

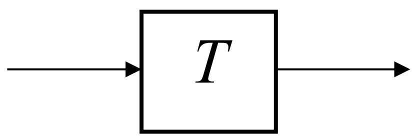
Parametric (Global) Warping

T is a coordinate changing machine

$$\boldsymbol{p}' = T(\boldsymbol{p})$$

Note: T is the same for all points, has relatively few parameters, and does **not** depend on image content

p = (x,y)



p’ = (x’,y’)

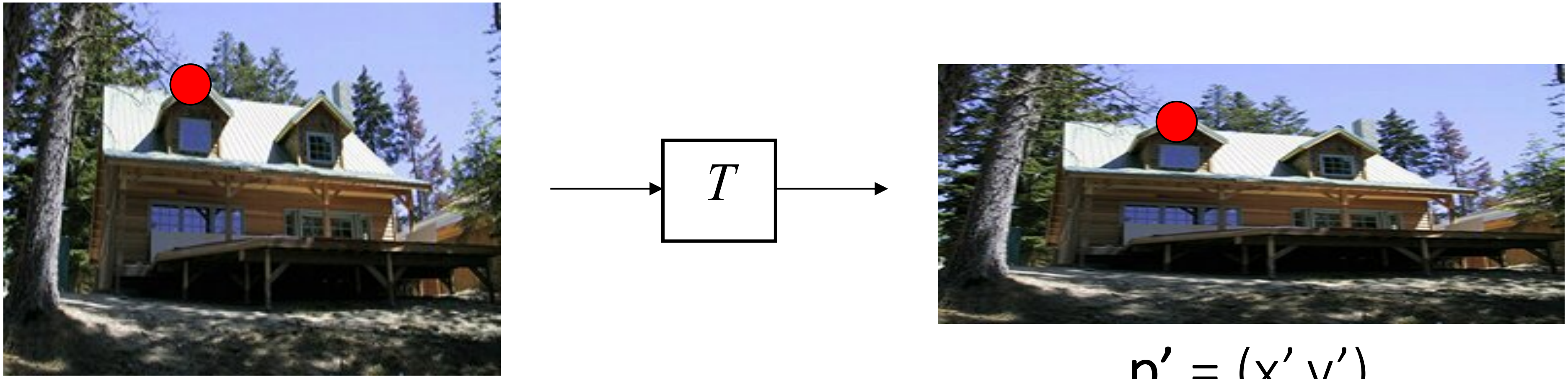
Slide credit: A. Efros

Parametric (Global) Warping

Today we'll deal with linear warps

$$\boldsymbol{p}' \equiv \boldsymbol{T}\boldsymbol{p}$$

T: matrix; p, p': 2D points. Start with normal points and =, then do homogeneous cords and ≡



p = (x,y)

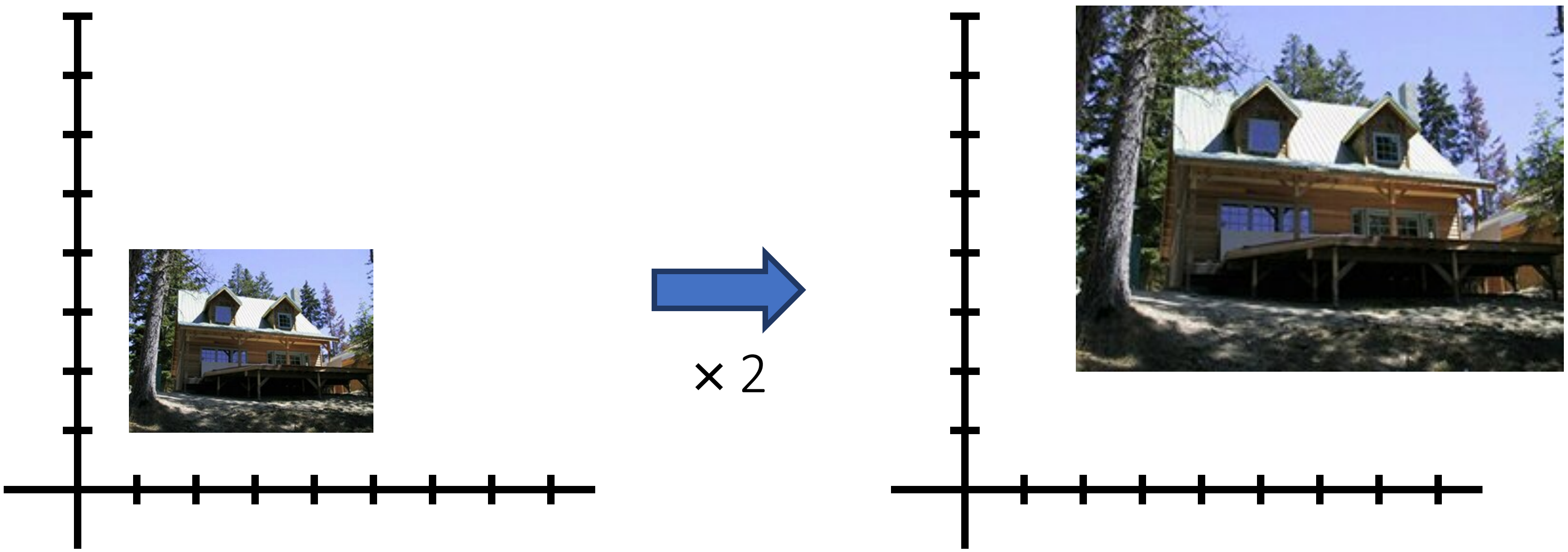
p' = (x',y')

Slide credit: A. Efros

Scaling

Scaling multiplies each component (x,y) by a scalar.
Uniform scaling is the same for all components.

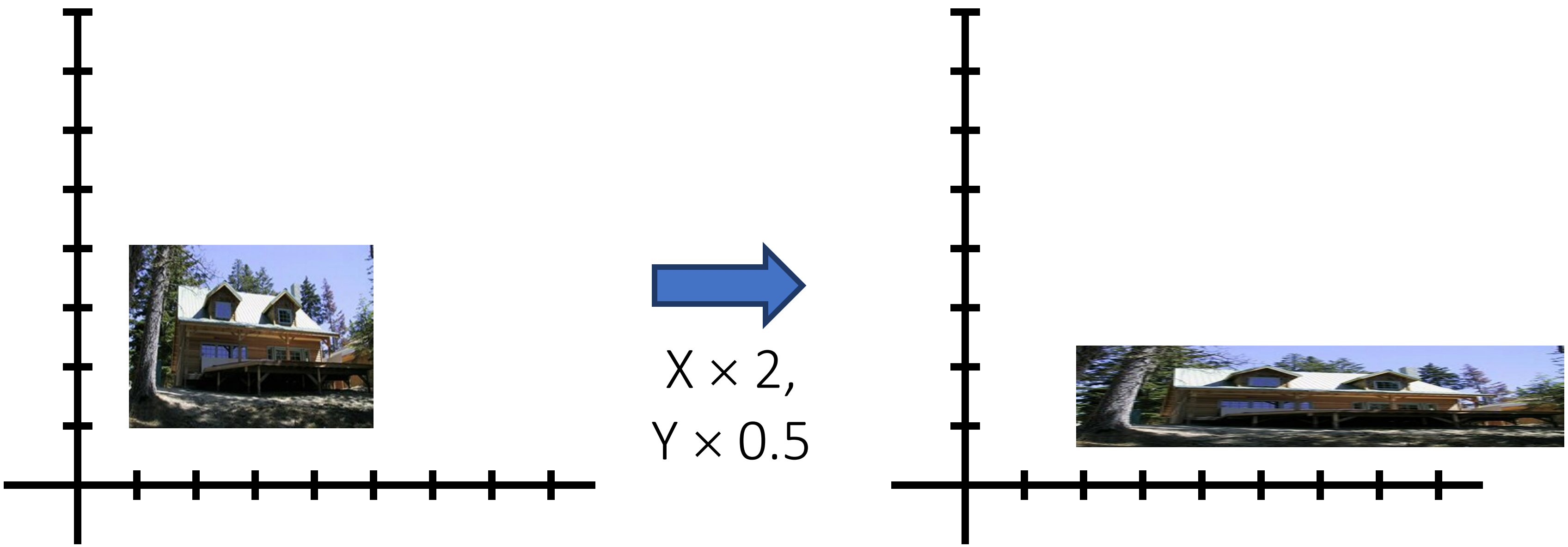
Note the corner goes from (1,1) to (2,2)



Slide credit: A. Efros

Scaling

Non-uniform scaling multiplies each component by a different scalar.



Slide credit: A. Efros

Scaling

What does T look like?

$$x' = ax$$
$$y' = by$$

Let's convert to a matrix:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\textit{scaling matrix } S} \begin{bmatrix} x \\ y \end{bmatrix}$$

What's the inverse of S?

Slide credit: A. Efros

2D Rotation

Rotation Matrix

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

But wait! Aren’t sin/cos non-linear?

x’ *is* a linear combination/function of *x, y*

x’ *is not* a linear function of θ

What’s the inverse of R_θ? $\boldsymbol{I} = \boldsymbol{R}_{\boldsymbol{\theta}}^{T} \boldsymbol{R}_{\boldsymbol{\theta}}$

Slide credit: A. Efros

Things you can do with 2x2

Identity / No Transformation

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

Shear

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

Slide credit: A. Efros

Things you can do with 2x2

Before

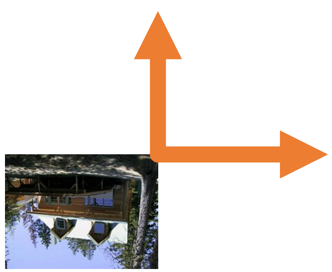
After

2D Mirror About Y-Axis

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

Before

After



2D Mirror About X,Y

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

Slide credit: A. Efros

What is Preserved?

3D lines project to 2D lines so lines are preserved

Projections of parallel 3D lines are not necessarily parallel, so not parallelism

Distant objects are smaller so size is not preserved

2x2: What is Preserved

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = T \begin{bmatrix} x \\ y \end{bmatrix}$$

After multiplication by T (irrespective of T)

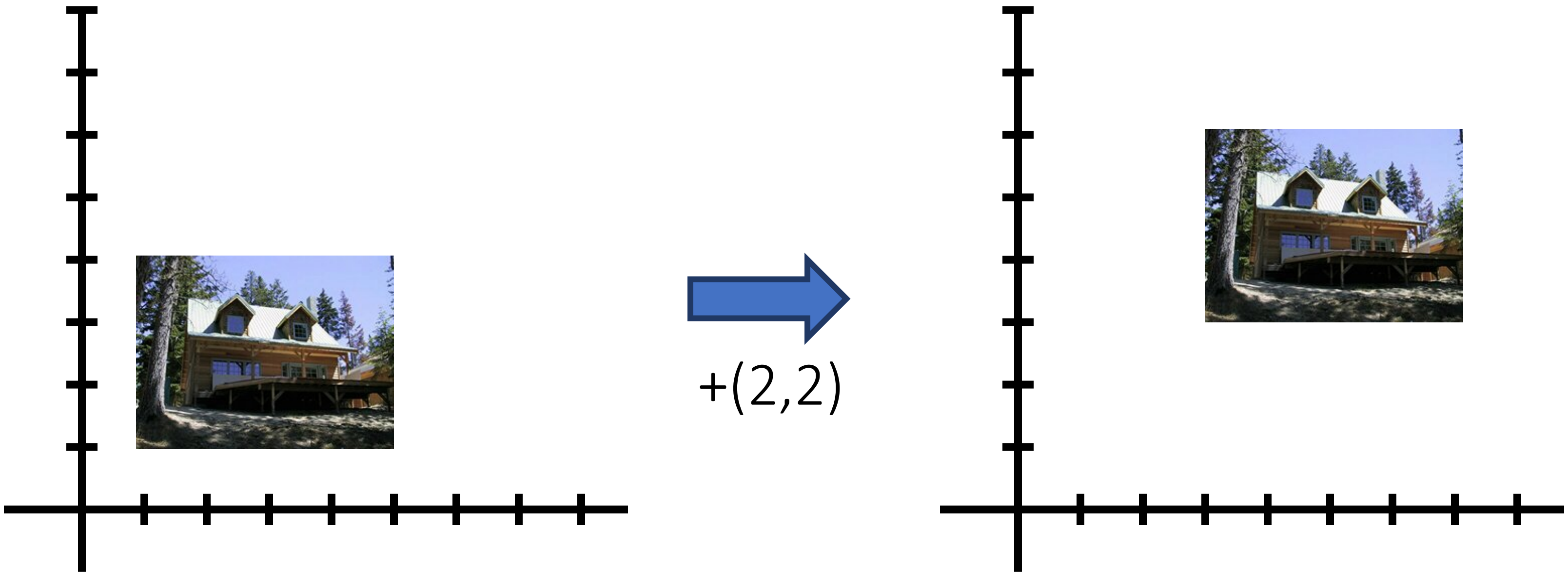
- Origin is origin: **0 = T0**
- Lines are lines
- Parallel lines are parallel

Things You Can't Do With 2x2

What about translation?

$x' = x + t_x,\ y' = y + t_y$

How do we fix it?

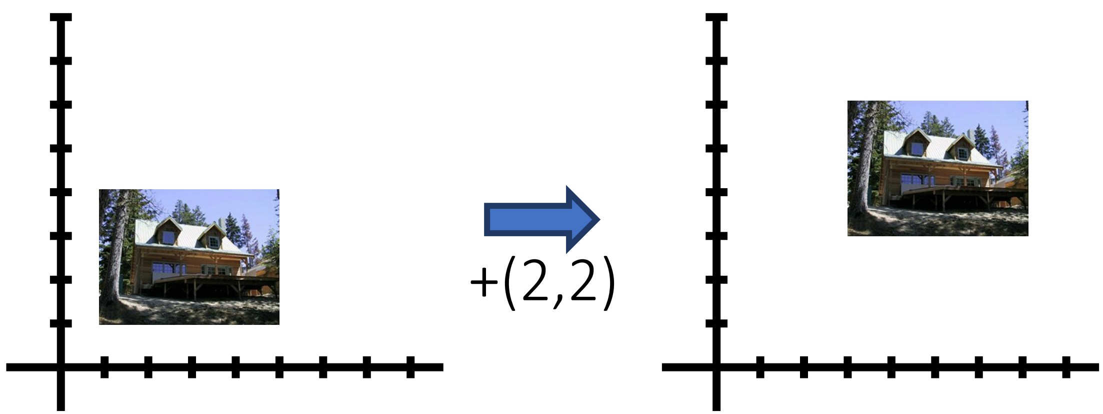


Homogenous Coordinates Again

What about translation?

x' = x + t_x, y' = y+t_y

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



Slide credit: A. Efros

Representing 2D Transformations

How do we represent a 2D transformation?

Let's pick scaling

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \equiv \begin{bmatrix} s_x & 0 & a \\ 0 & s_y & b \\ d & e & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

What's	a	b	d	e	f
	0	0	0	0	1

Affine Transformations

Affine: *linear transformation* plus *translation*

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \equiv \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Will the last coordinate always be 1?

In general (without homogeneous coordinates)

$$\boldsymbol{x}' = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$$

Composing Transforms

We can combine transformations via matrix multiplication.

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \equiv \underbrace{\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}}_{T(t_x, t_y)} \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{R(\theta)} \underbrace{\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{S(s_x, s_y)} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Does order matter?

Slide credit: A. Efros

Affine: What is Preserved

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \equiv \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \boldsymbol{T} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

After multiplication by T (irrespective of T)

- ~~**Origin is origin: 0 = T0**~~
- Lines are lines
- Parallel lines are parallel

Perspective Transformations

Set bottom row to not [0,0,1]

Called a perspective/projective transformation or a ***homography***

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

How many degrees of freedom?

How Many Degrees of Freedom?

Recall: can always scale by non-zero value

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

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

Homography can always be re-scaled by $\lambda \neq 0$

Perspective: What is Preserved

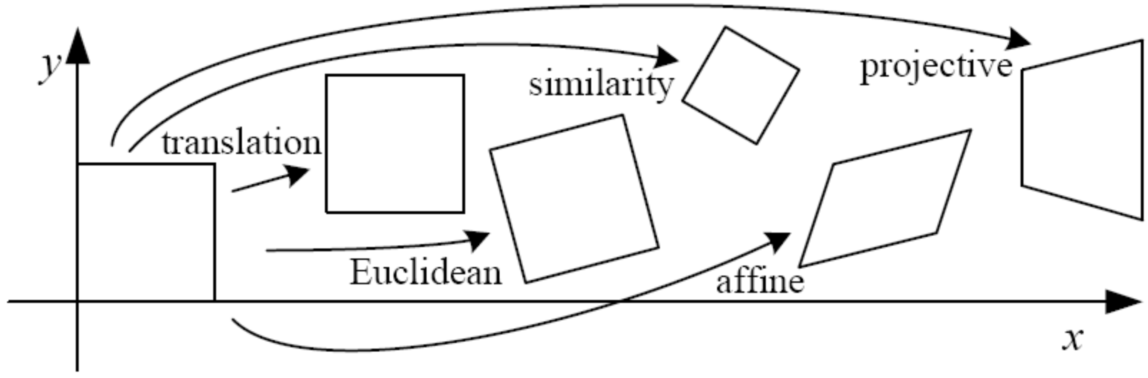
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \equiv \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \equiv \boldsymbol{T} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

After multiplication by T (irrespective of T)

- ~~**Origin is origin: 0 = T0**~~
- Lines are lines
- ~~Parallel lines are parallel~~
- ~~Ratios between distances~~

Transformation Families

In general: transformations are a nested set of groups



Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orientation $+ \cdots$	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	lengths $+ \cdots$	
similarity	$\begin{bmatrix} s\boldsymbol{R} \mid \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	angles $+ \cdots$	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	parallelism $+ \cdots$	
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}_{3\times 3}$	8	straight lines	

Diagram credit: R. Szeliski

What Can Homography Do?

Homography example 1: any two views of a *planar* surface

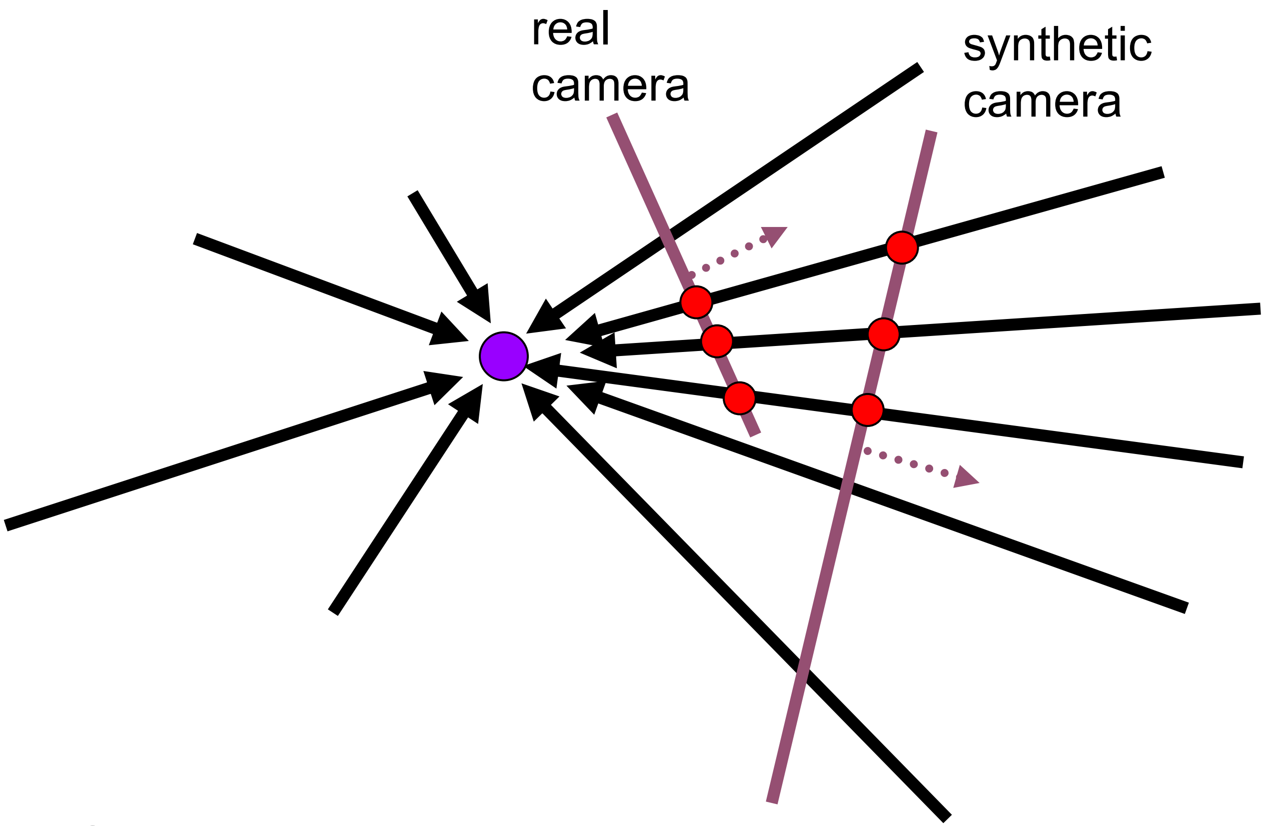
Figure Credit: S. Lazebnik

What Can Homography Do?

Homography example 2: any images from two cameras sharing a camera center

Figure Credit: S. Lazebnik

What Can Homography Do?



Can generate any synthetic camera view
as long as it has **the same center of projection**!

Slide Credit: A. Efros

What Can Homography Do?

Homography sort of example “3”: far away scene that can be approximated by a plane

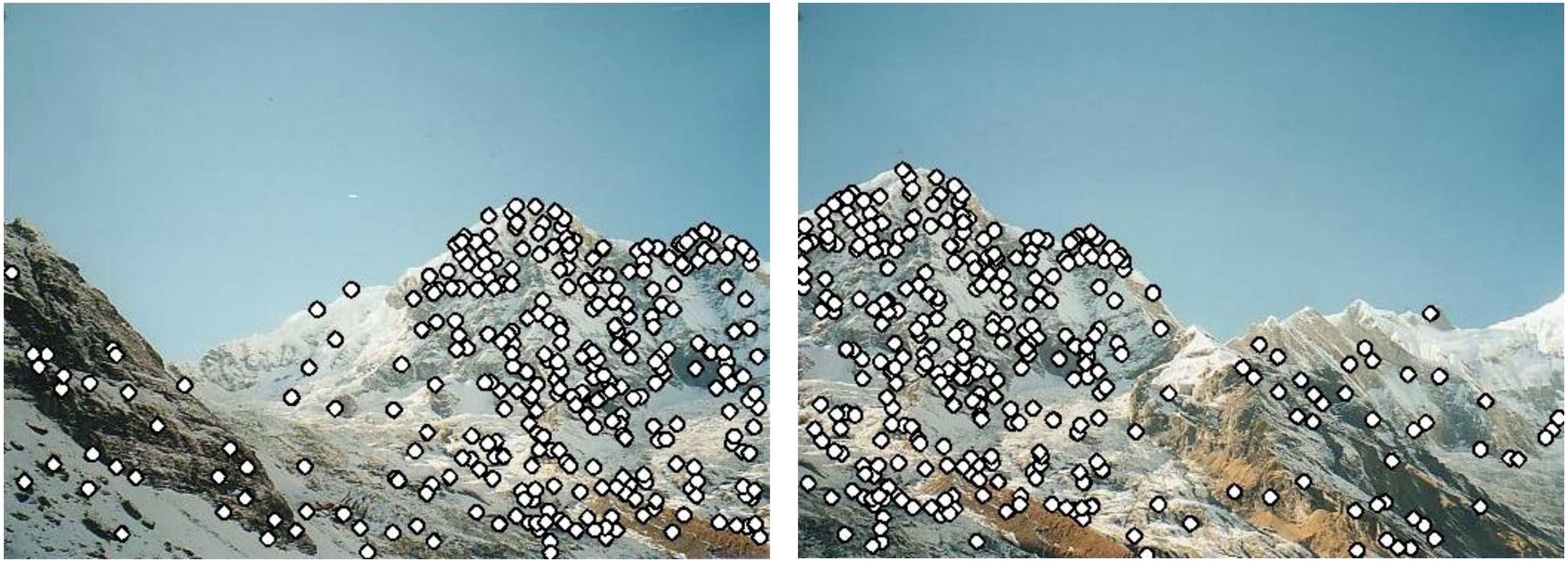


Figure credit: Brown & Lowe

Fun With Homography

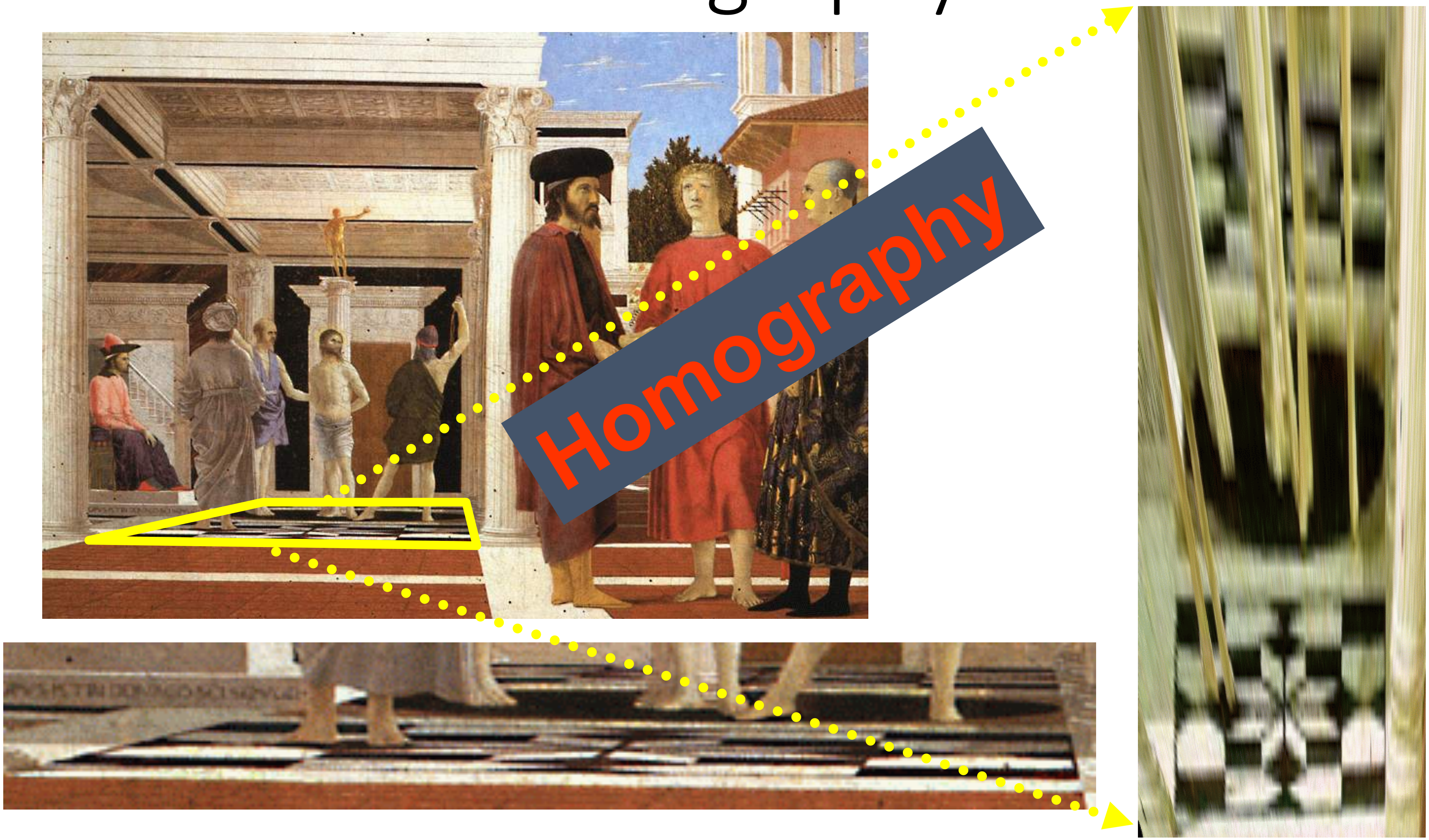
Original image

St. Petersburg
photo by A. Tikhonov

Virtual camera rotations

Slide Credit: A. Efros

Fun With Homography

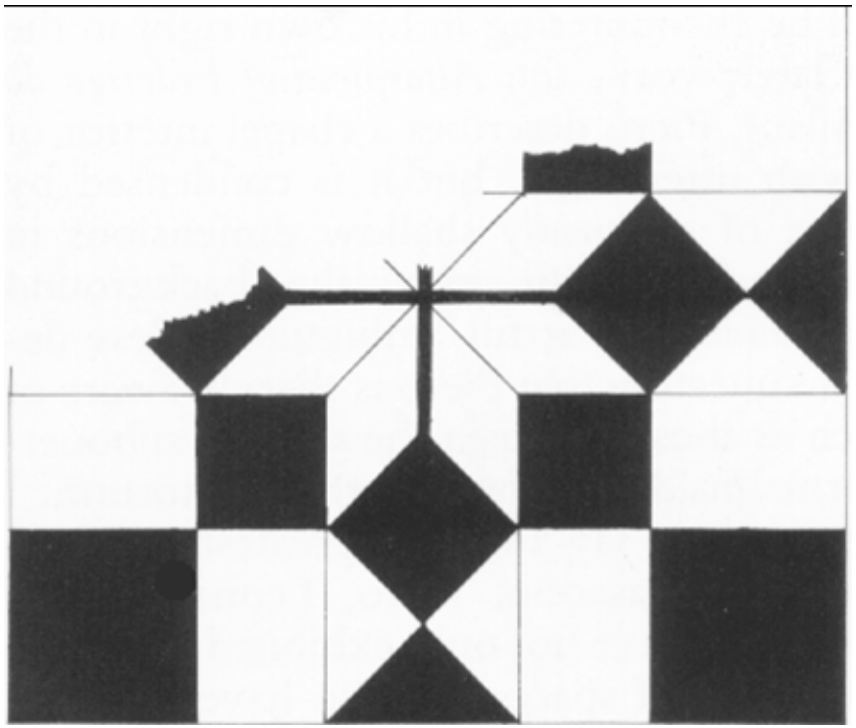


The floor (enlarged)

Automatically rectified floor

Slide from A. Criminisi

Fun With Homography



From Martin Kemp *The Science of Art*
(manual reconstruction)

Automatic rectification

Slide from A. Criminisi

Fun With Homography

St. Lucy Altarpiece, D. Veneziano

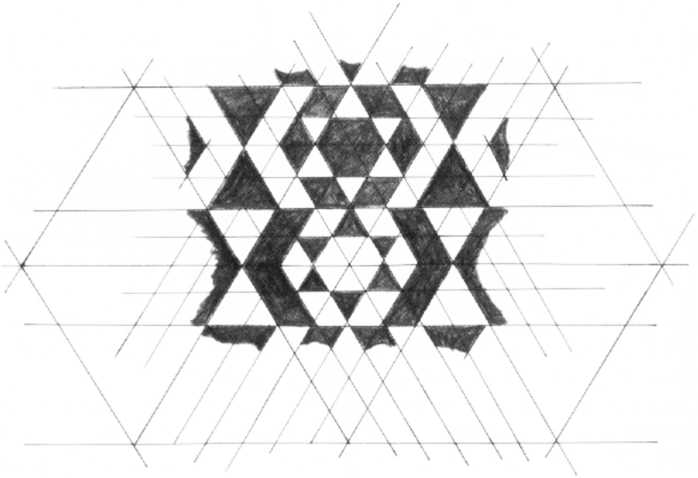
Slide from A. Criminisi

What is the (complicated) shape of the floor pattern?

Automatically rectified floor

Fun With Homography

Automatic rectification



From Martin Kemp, *The Science of Art* *(manual reconstruction)*

Slide from A. Criminisi

Today

Categories of Transformations

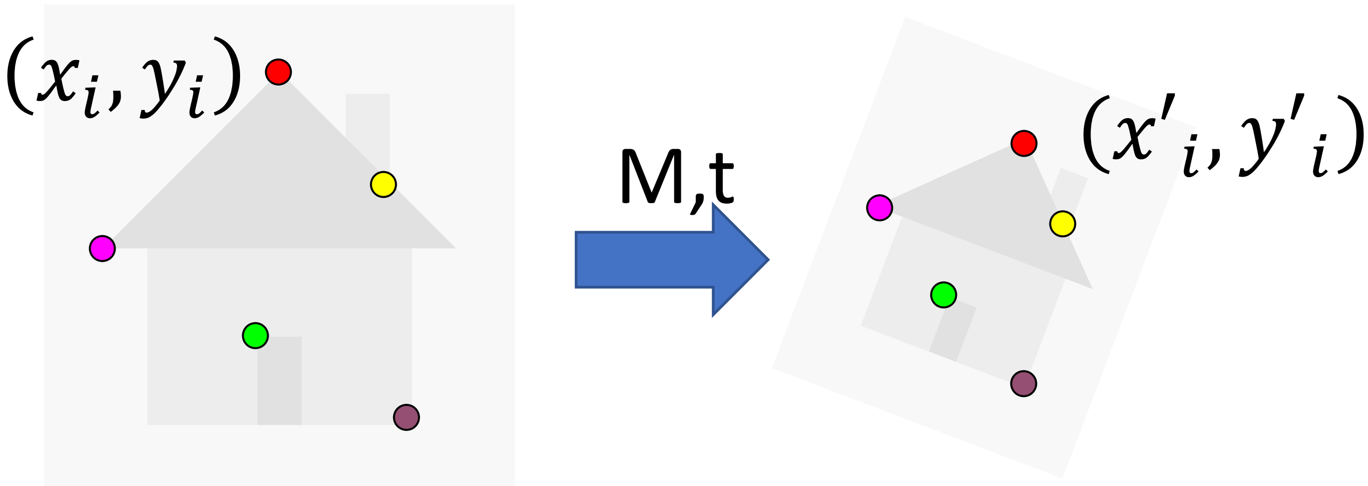
Fitting Transformations

Applying Transformations

Blending Images

Fitting Transformations

Setup: have pairs of correspondences



$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \boldsymbol{M} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \boldsymbol{t}$$

Slide Credit: S. Lazebnik

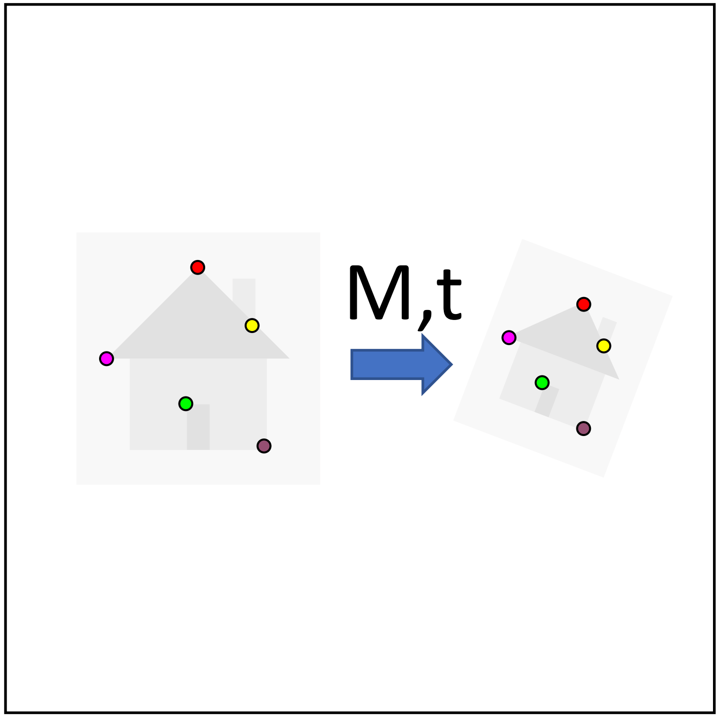
Fitting Transformations: Affine

Affine Transformation: M,t

Data: (x_i, y_i, x'_i, y'_i) for $i=1,\ldots,k$

Model:
$[x'_i, y'_i] = \mathbf{M}[x_i, y_i] + \mathbf{t}$

Objective function:
$||[x'_i, y'_i] - \mathbf{M}[x_i, y_i] + \mathbf{t}||^2$



Fitting Transformations: Affine

Given correspondences: **p**' = [x'_i,y'_i], **p =** [x_i,y_i]

$$\begin{bmatrix} x_i{}' \\ y_i{}' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Set up two equations per point

$$\begin{bmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{bmatrix} = \begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix}$$

Solve with least squares!

Fitting Transformations: Affine

$$\begin{bmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{bmatrix} = \begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix}$$

2 equations per point, 6 unknowns

How many points do we need?

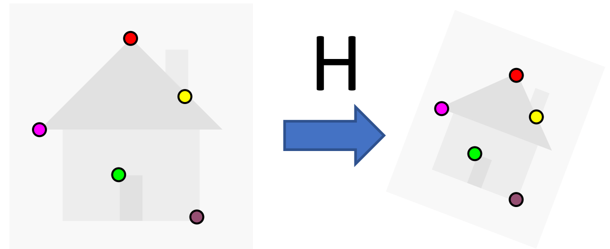
Fitting Transformations: Homography

Homography: H

Data: (x_i, y_i, x'_i, y'_i) for $i=1,\ldots,k$

Model:
$[x'_i, y'_i, 1] \equiv \mathbf{H}[x_i, y_i, 1]$

Objective function:
It's complicated



Fitting Transformations: Homography

Want:

$$\begin{bmatrix} x_i' \\ y_i' \\ w_i' \end{bmatrix} \equiv \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ w_i \end{bmatrix}$$

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ w_i \end{bmatrix} \equiv \boldsymbol{H}\boldsymbol{x_i} \equiv \begin{bmatrix} \boldsymbol{h_1^T} \\ \boldsymbol{h_2^T} \\ \boldsymbol{h_3^T} \end{bmatrix} \boldsymbol{x_i} \equiv \begin{bmatrix} \boldsymbol{h_1^T x_i} \\ \boldsymbol{h_2^T x_i} \\ \boldsymbol{h_3^T x_i} \end{bmatrix}$$

Recall: $\boldsymbol{a} \equiv \boldsymbol{b} \rightarrow \boldsymbol{a} = \lambda \boldsymbol{b} \rightarrow \boldsymbol{a} \times \boldsymbol{b} = \boldsymbol{0}$

Fitting Transformations: Homography

Want:

$$\begin{bmatrix} x_i' \\ y_i' \\ w_i' \end{bmatrix} \equiv \begin{bmatrix} \boldsymbol{h}_1^T \boldsymbol{x}_i \\ \boldsymbol{h}_2^T \boldsymbol{x}_i \\ \boldsymbol{h}_3^T \boldsymbol{x}_i \end{bmatrix} \Leftrightarrow \begin{bmatrix} x_i' \\ y_i' \\ w_i' \end{bmatrix} \times \begin{bmatrix} \boldsymbol{h}_1^T \boldsymbol{x}_i \\ \boldsymbol{h}_2^T \boldsymbol{x}_i \\ \boldsymbol{h}_3^T \boldsymbol{x}_i \end{bmatrix} = \boldsymbol{0}$$

Cross-product

$$\begin{bmatrix} y_i' \boldsymbol{h}_3^T \boldsymbol{x}_i - w_i' \boldsymbol{h}_2^T \boldsymbol{x}_i \\ w_i' \boldsymbol{h}_1^T \boldsymbol{x}_i - x_i' \boldsymbol{h}_3^T \boldsymbol{x}_i \\ x_i' \boldsymbol{h}_2^T \boldsymbol{x}_i - y_i' \boldsymbol{h}_1^T \boldsymbol{x}_i \end{bmatrix} = \boldsymbol{0}$$

Re-arrange and put 0s in

$$\begin{bmatrix} \boldsymbol{h}_1^T \boldsymbol{0} - w_i' \boldsymbol{h}_2^T \boldsymbol{x}_i + y_i' \boldsymbol{h}_3^T \boldsymbol{x}_i \\ w_i' \boldsymbol{h}_1^T \boldsymbol{x}_i + \boldsymbol{h}_2^T \boldsymbol{0} - x_i' \boldsymbol{h}_3^T \boldsymbol{x}_i \\ -y_i' \boldsymbol{h}_1^T \boldsymbol{x}_i + x_i' \boldsymbol{h}_2^T \boldsymbol{x}_i + \boldsymbol{h}_3^T \boldsymbol{0} \end{bmatrix} = \boldsymbol{0}$$

Fitting Transformations: Homography

Equation
$$\begin{bmatrix} \boldsymbol{h}_1^T\boldsymbol{0} - w_i'\boldsymbol{h}_2^T\boldsymbol{x}_i + y_i'\boldsymbol{h}_3^T\boldsymbol{x}_i \\ w_i'\boldsymbol{h}_1^T\boldsymbol{x}_i + \boldsymbol{h}_2^T\boldsymbol{0} - x_i'\boldsymbol{h}_3^T\boldsymbol{x}_i \\ -y_i'\boldsymbol{h}_1^T\boldsymbol{x}_i + x_i'\boldsymbol{h}_2^T\boldsymbol{x}_i + \boldsymbol{h}_3^T\boldsymbol{0} \end{bmatrix} = \boldsymbol{0}$$

Pull out h
$$\begin{bmatrix} \boldsymbol{0}^T & -w'_i\boldsymbol{x}_i^T & y'_i\boldsymbol{x}_i^T \\ w_i'\boldsymbol{x}_i^T & \boldsymbol{0}^T & -x_i'\boldsymbol{x}_i^T \\ -y_i'\boldsymbol{x}_i^T & x_i'\boldsymbol{x}_i^T & \boldsymbol{0}^T \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \boldsymbol{h}_3 \end{bmatrix} = \boldsymbol{0}$$

Only two linearly independent equations

$$\frac{x_i'}{w_i'}[0 \quad -w_i' \quad y_i'] + \frac{y_i'}{w_i'}[w_i' \quad 0 \quad -x_i'] + [-y_i' \quad x_i' \quad 0] = \boldsymbol{0}$$

Fitting Transformations: Homography

N points $\rightarrow$

$$\begin{array}{cc} & 9 \\ 2n & \begin{bmatrix} \mathbf{0}^T & -w_1'\boldsymbol{x}_1^T & y_1'\boldsymbol{x}_1^T \\ w_1'\boldsymbol{x}_1^T & \mathbf{0}^T & -x_1'\boldsymbol{x}_1^T \\ & \vdots & \\ \mathbf{0}^T & -w_n'\boldsymbol{x}_n^T & y_n'\boldsymbol{x}_n^T \\ w_n'\boldsymbol{x}_n^T & \mathbf{0}^T & -x_n'\boldsymbol{x}_n^T \end{bmatrix} \end{array} \begin{bmatrix} \boldsymbol{h}_1 \\ \boldsymbol{h}_2 \\ \boldsymbol{h}_3 \end{bmatrix} = \mathbf{0}$$

$$\boldsymbol{A}\boldsymbol{h} = \mathbf{0}$$

If h is up to scale, what do we use from last time?

$$h^* = \arg \min_{\|h\|=1} \|Ah\|^2 \rightarrow$$ Eigenvector of A^TA with smallest eigenvalue

Fitting Transforms: Small Detail

$||Ah||^2$ doesn't measure model fit (it's called an *algebraic error* that's mainly just convenient to minimize*)*

Really want *geometric error*:

$$\sum_{i=1}^{k} \|[x_i', y_i'] - T([x_i, y_i])\|^2 + \|[x_i, y_i] - T^{-1}([x_i', y_i'])\|^2$$

Fitting Transformations: Small Detail

Solution: initialize with algebraic (min $||Ah||$), optimize with geometric using standard non-linear optimizer

In RANSAC, we always take just enough points to fit. Why might this not make a big difference when fitting a model with RANSAC?

Today

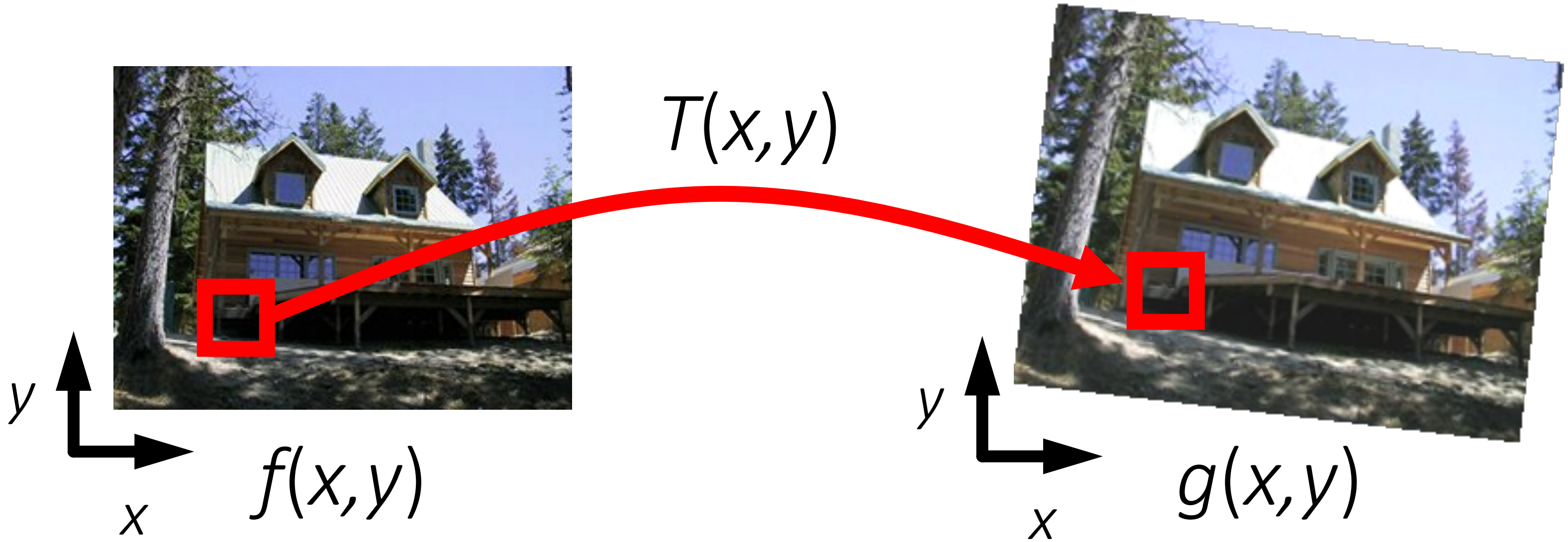
Categories of Transformations

Fitting Transformations

Applying Transformations

Blending Images

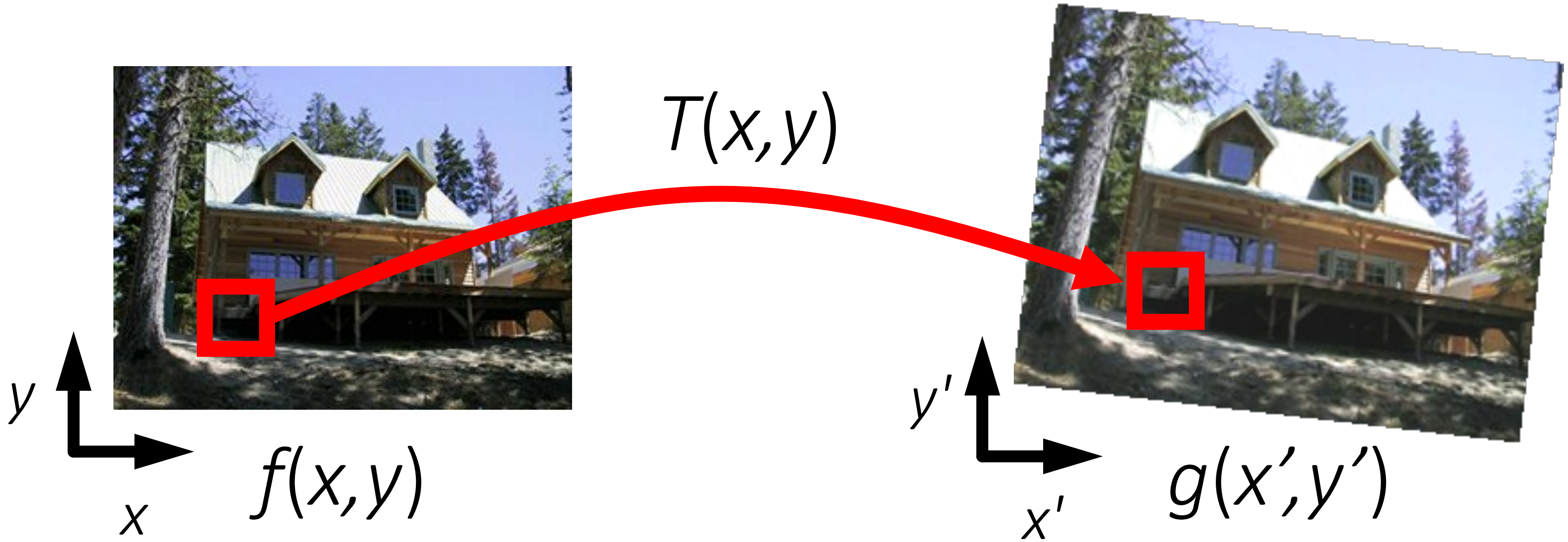
Image Warping



Given a coordinate transform $(x',y') = T(x,y)$ and a source image $f(x,y)$, how do we compute a transformed image $g(x',y') = f(T(x,y))$?

Slide Credit: A. Efros

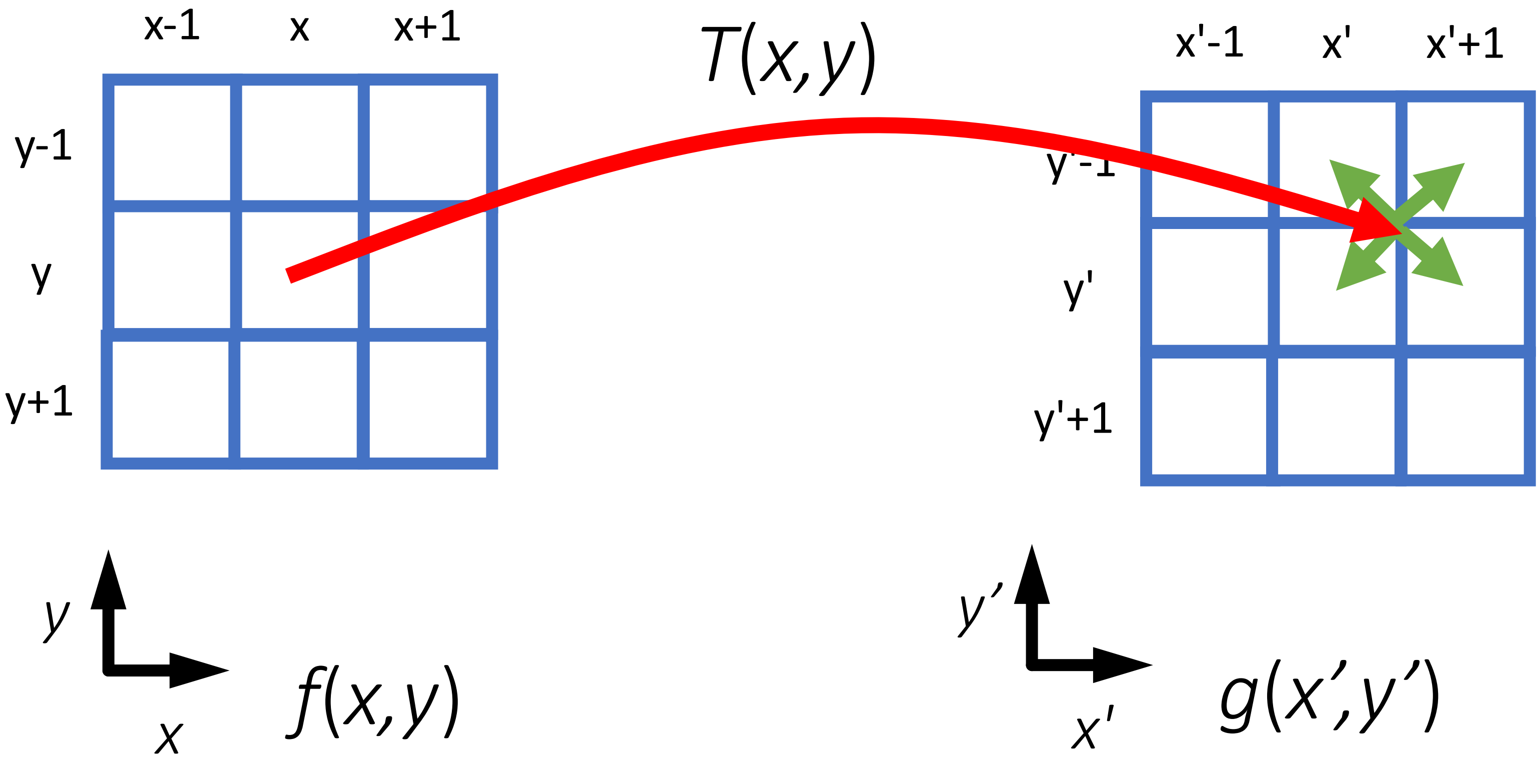
Forward Warping



Send the value at each pixel (x,y) to the new pixel (x',y') = T([x,y])

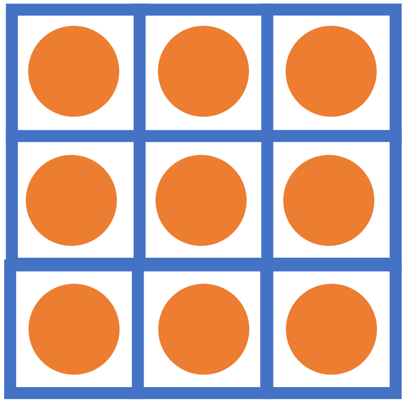
Slide Credit: A. Efros

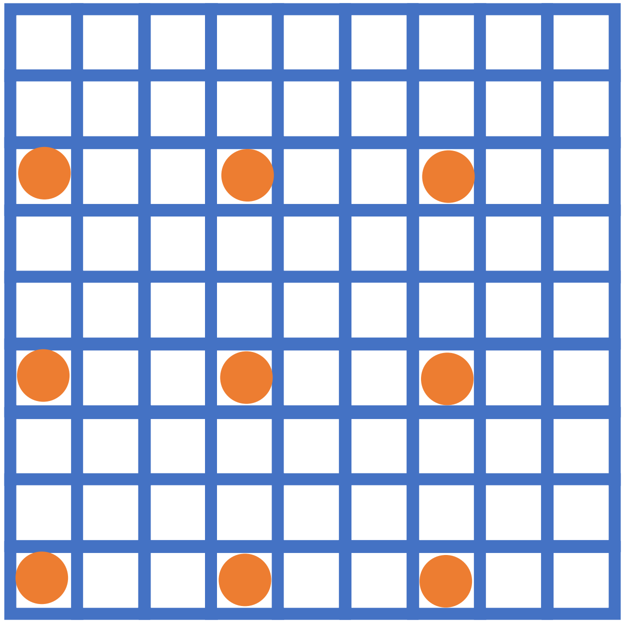
Forward Warping



If you don't hit an exact pixel, give the value to each of the neighboring pixels ("splatting").

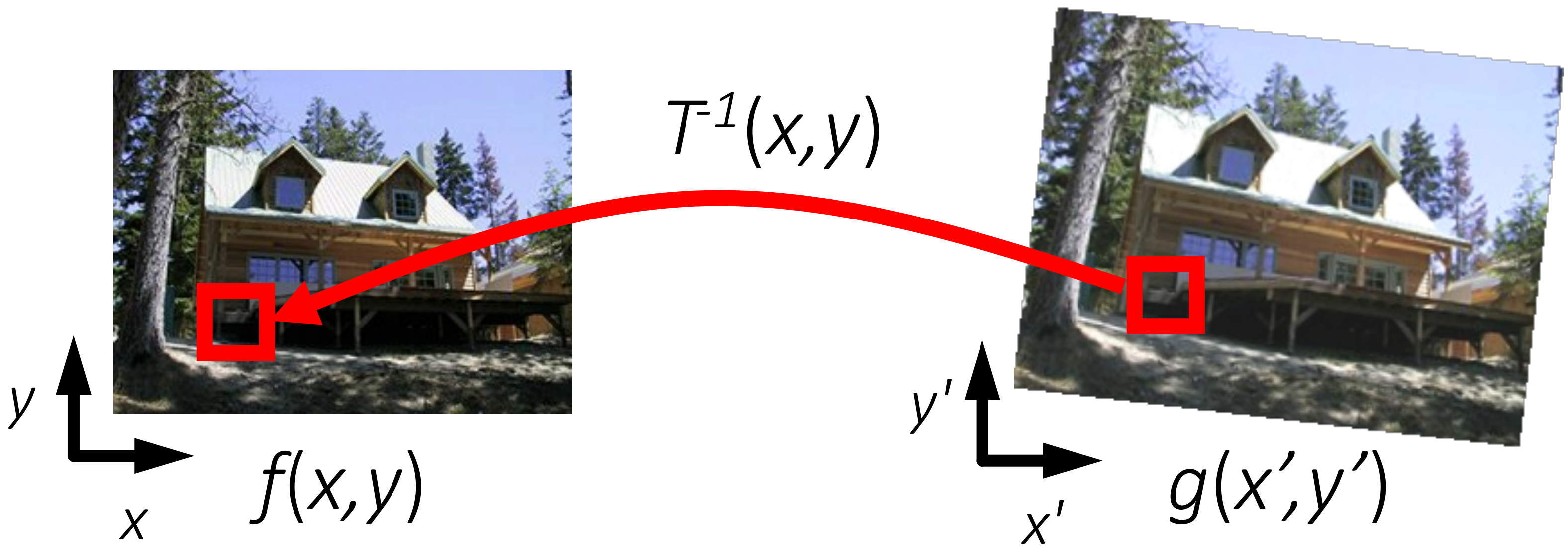
Forward Warping





Suppose T(x,y) scales by a factor of 3.
Hmmmm.

Backward Warping

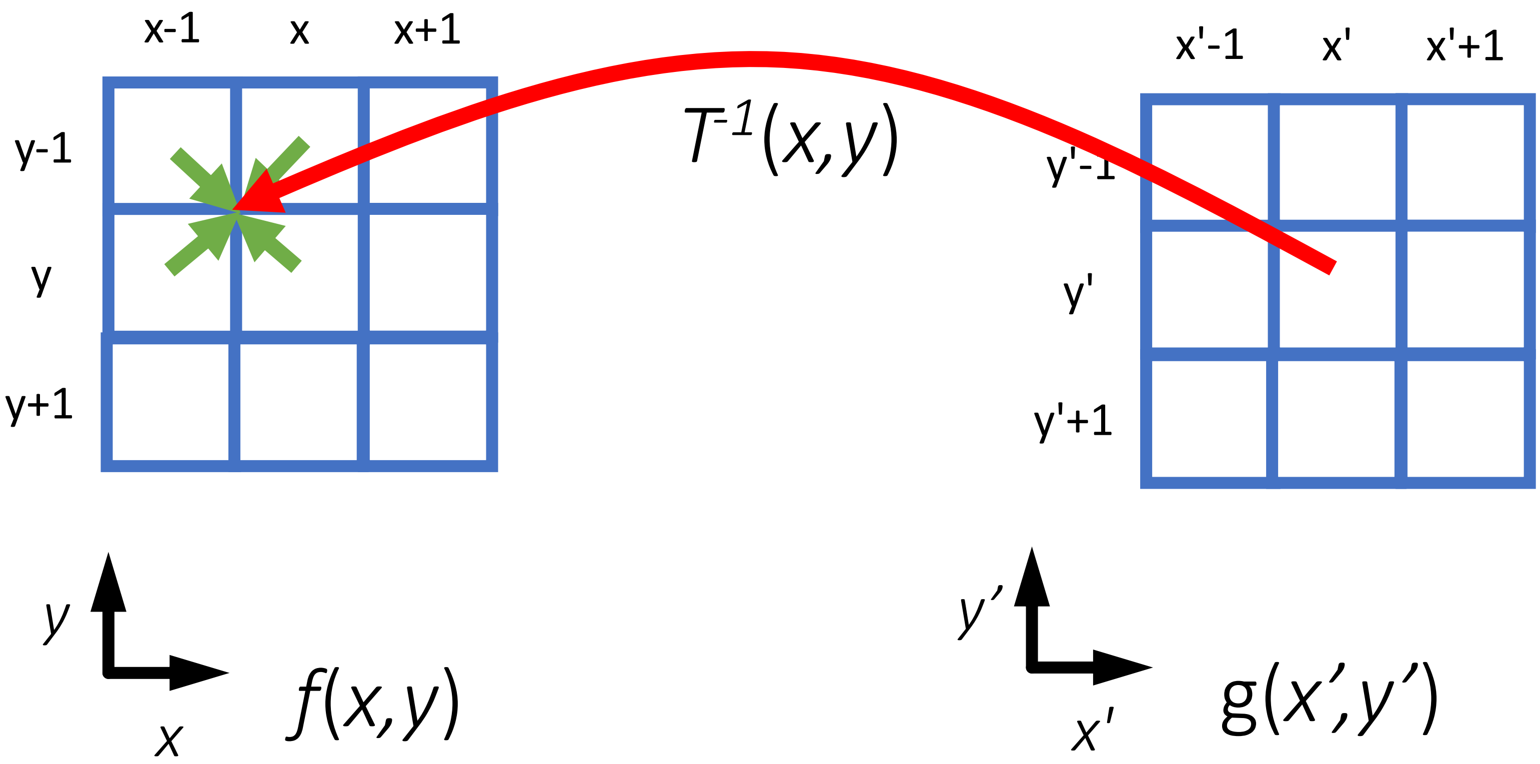


Find out where each pixel g(x',y') should get its value from, and steal it.

Note: requires ability to invert T

Slide Credit: A. Efros

Backward Warping



If you don't hit an exact pixel, figure out how to take it from the neighbors.

Today

Categories of Transformations

Fitting Transformations

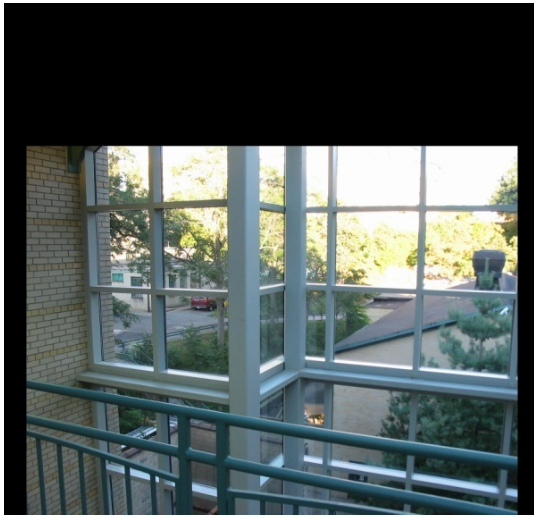
Applying Transformations

Blending Images

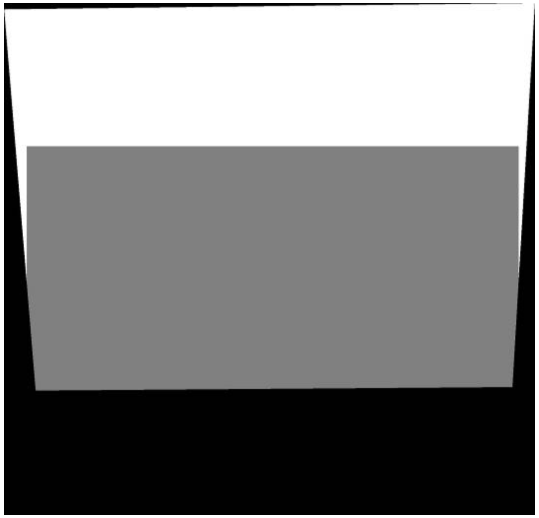
Blending Images

Warped Input 1 I_1

Warped Input 2 I_2



α



$\alpha I_1 + (1-\alpha) I_2$

Slide Credit: A. Efros

Simple Approach: Two-Band Blending

- Brown & Lowe, 2003
  - Break up each image into high frequency + low frequency
  - Linearly blend low-frequency information
  - No blending for high-frequency: at each pixel take from one image or the other

Figure Credit: Brown & Lowe

Simple Approach: Two-Band Blending

Low frequency ($\lambda > 2$ pixels)

High frequency ($\lambda < 2$ pixels)

Linear Blending

2-band Blending

Today

Categories of Transformations

Fitting Transformations

Applying Transformations

Blending Images

Putting It All Together

How do you make a panorama?

Step 1: Find “features” to match

Step 2: Describe Features

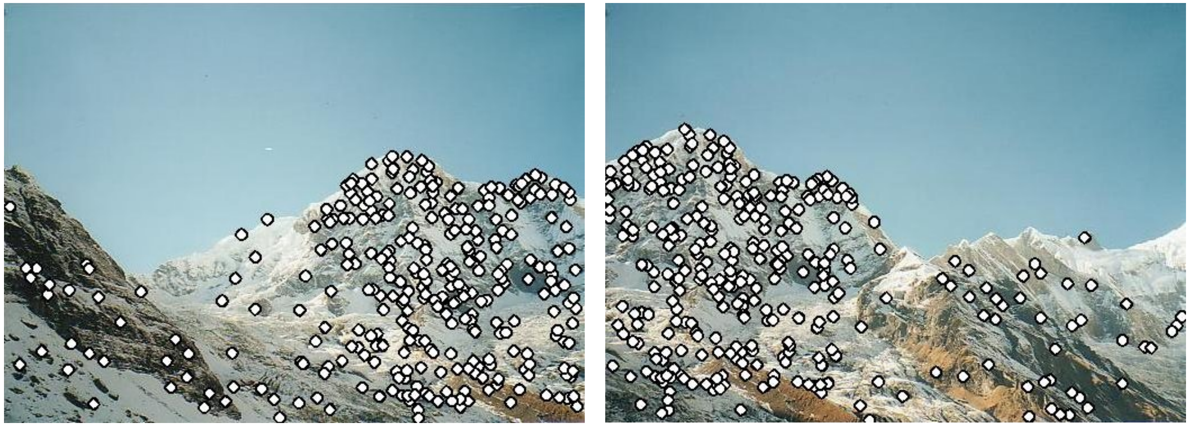
Step 3: Match by Nearest Neighbor

Step 4: Fit H via RANSAC

Step 5: Blend Images

Putting It All Together: Step 1

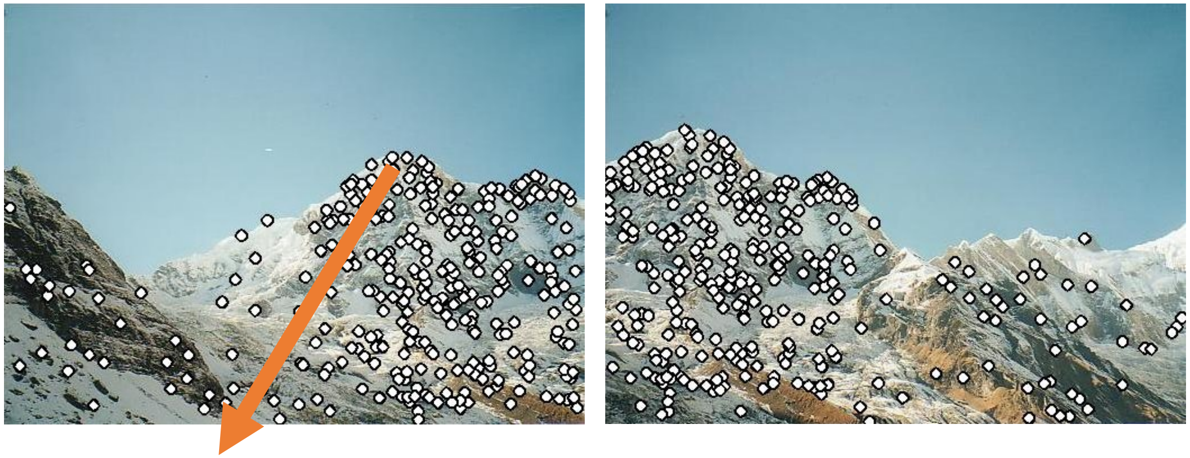
Find corners/blobs



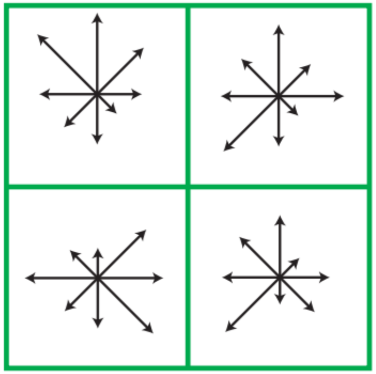
- (Multi-scale) Harris; or
- Laplacian of Gaussian

Putting It All Together: Step 2

Describe Regions Near Features



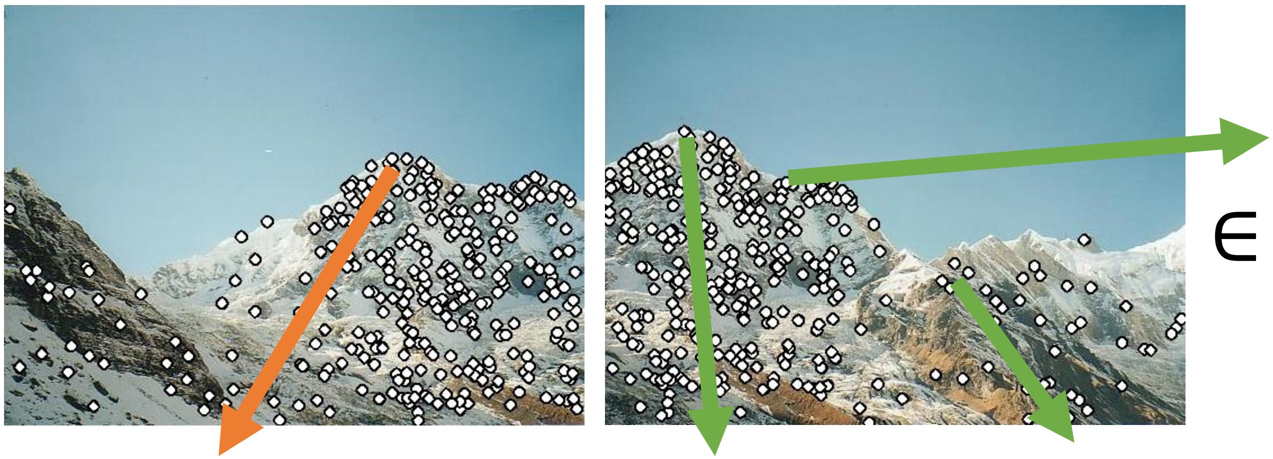
$x_q \in R^{128}$



Build histogram of gradient orientations (SIFT)

Putting It All Together: Step 3

Match Features Based On Region



$x_3 \in R^{128}$

$x_q \in R^{128}$ $\quad$ $x_1 \in R^{128}$ $\quad$ $x_2 \in R^{128}$

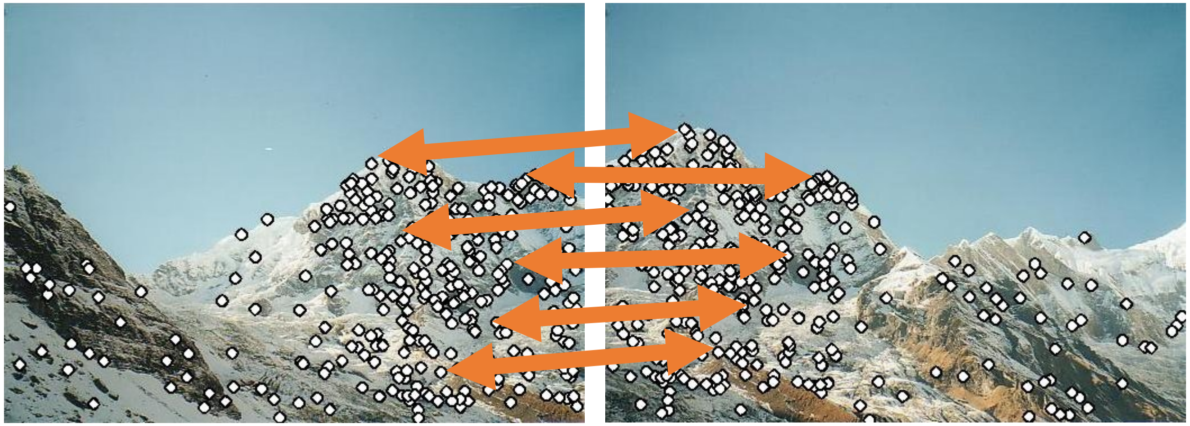
Sort by distance to: x_q $\quad \|x_q - x_1\| < \|x_q - x_2\| < \|x_q - x_3\|$

Accept match if: $\quad \|x_q - x_1\| / \|x_q - x_2\|$

Nearest neighbor is far closer than 2nd nearest neighbor

Putting It All Together: Step 4

Fit transformation H via RANSAC



for trial in range(Ntrials):

 Pick sample

 Fit model

 Check if more inliers

Re-fit model with most inliers

$$\arg\min_{\|\boldsymbol{h}\|=1} \|\boldsymbol{A}\boldsymbol{h}\|^2$$

Putting It All Together: Step 5

Warp images together

Resample images with inverse warping and blend

So far:
Filtering and Matching

Next up:
Recognition
Linear Models
Neural Networks