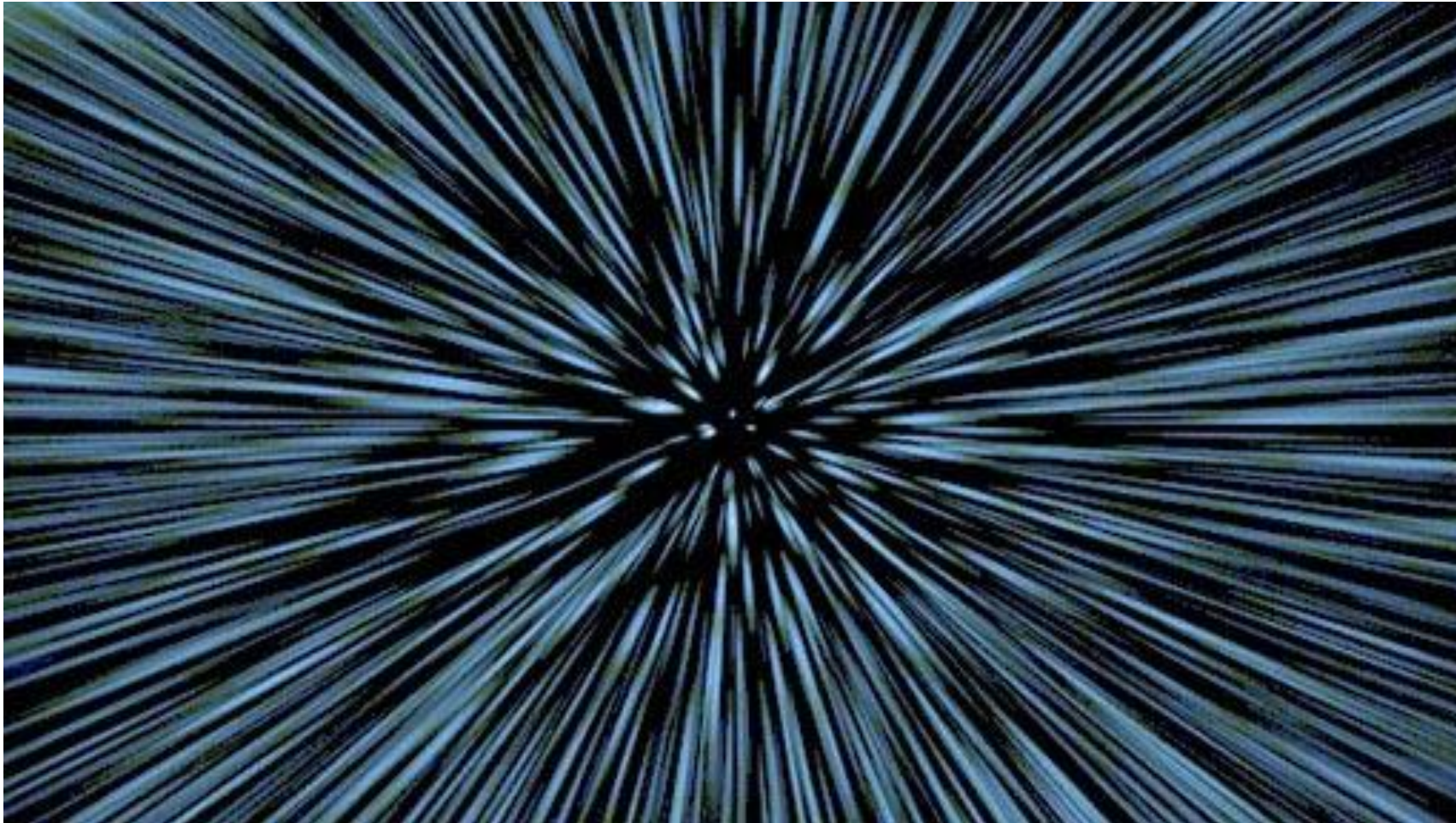


2D transformations (a.k.a. warping)



Course announcements

- Homework 0 and homework 1 are available on course website.
 - Homework 1 is due on February 7th **(tonight!)**.
 - Any questions about the homeworks?
 - How many of you have looked at/started/finished homework 0?
 - How many of you have looked at/started/finished homework 1?
- Homework 2 will be posted tonight.
 - Will be due on February 21st at midnight.
 - Start early cause it is much larger and more difficult than homework 1.

Overview of today's lecture

- Reminder: image transformations.
- 2D transformations.
- Projective geometry 101.
- Transformations in projective geometry.
- Classification of 2D transformations.
- Determining unknown 2D transformations.
- Determining unknown image warps.

Slide credits

Most of these slides were adapted from:

- Kris Kitani (16-385, Spring 2017).

Reminder: image transformations

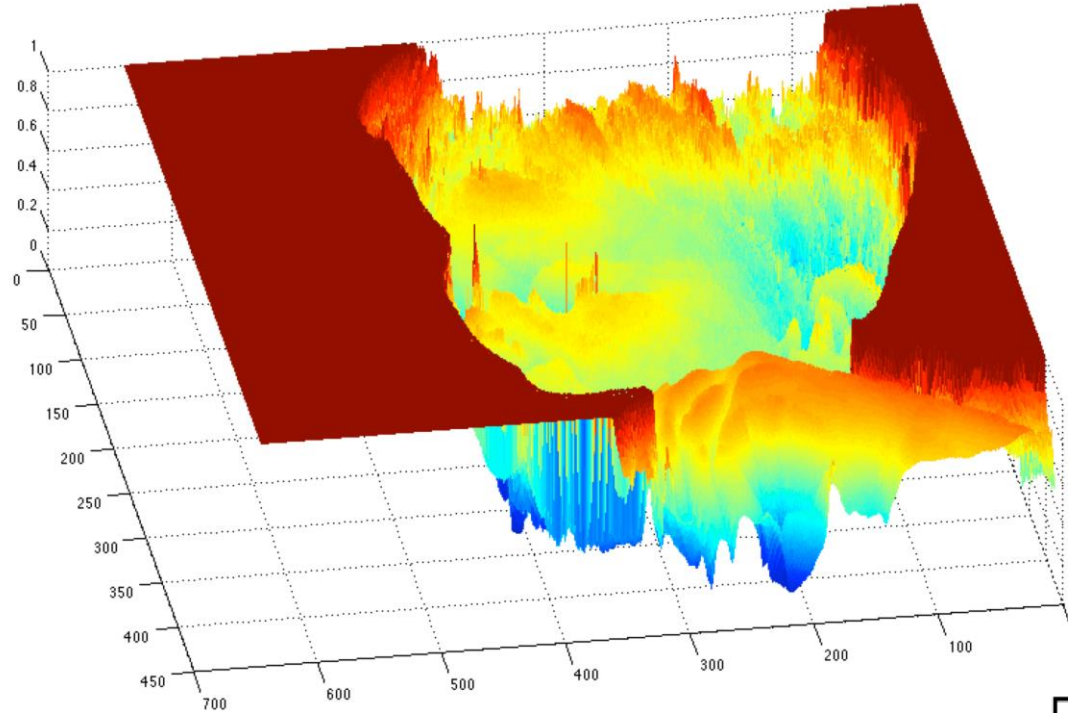
What is an image?



grayscale image

What is the range of the image function f ?

$f(\mathbf{x})$



domain $\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

A (grayscale) image is a 2D function.

What types of image transformations can we do?



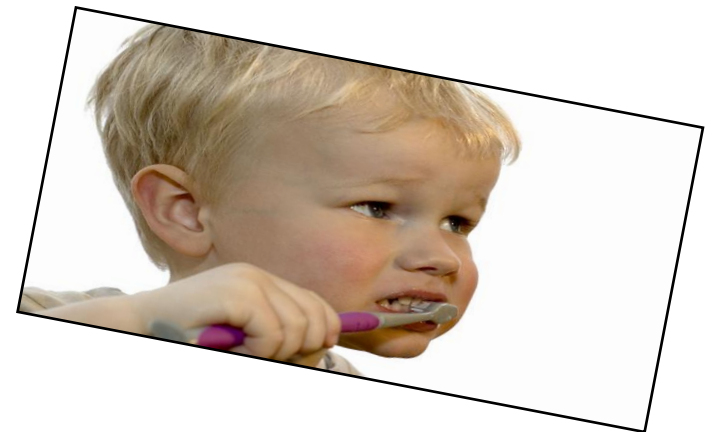
Filtering



changes pixel *values*



Warping



changes pixel *locations*

What types of image transformations can we do?

F



Filtering



$$G(\mathbf{x}) = h\{F(\mathbf{x})\}$$

G



changes *range* of image function

F

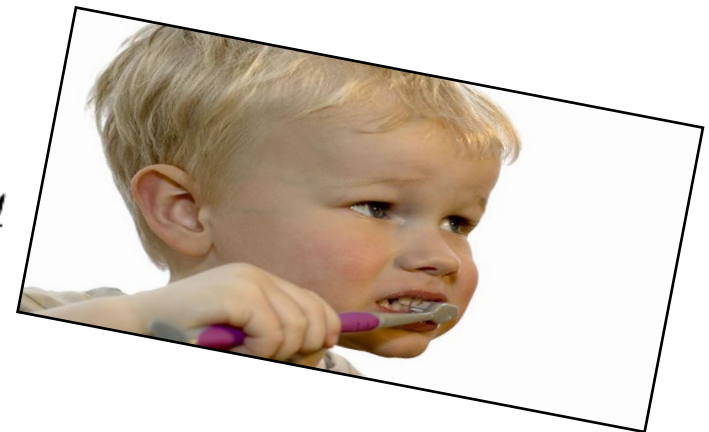


Warping



$$G(\mathbf{x}) = F(h\{\mathbf{x}\})$$

G



changes *domain* of image function

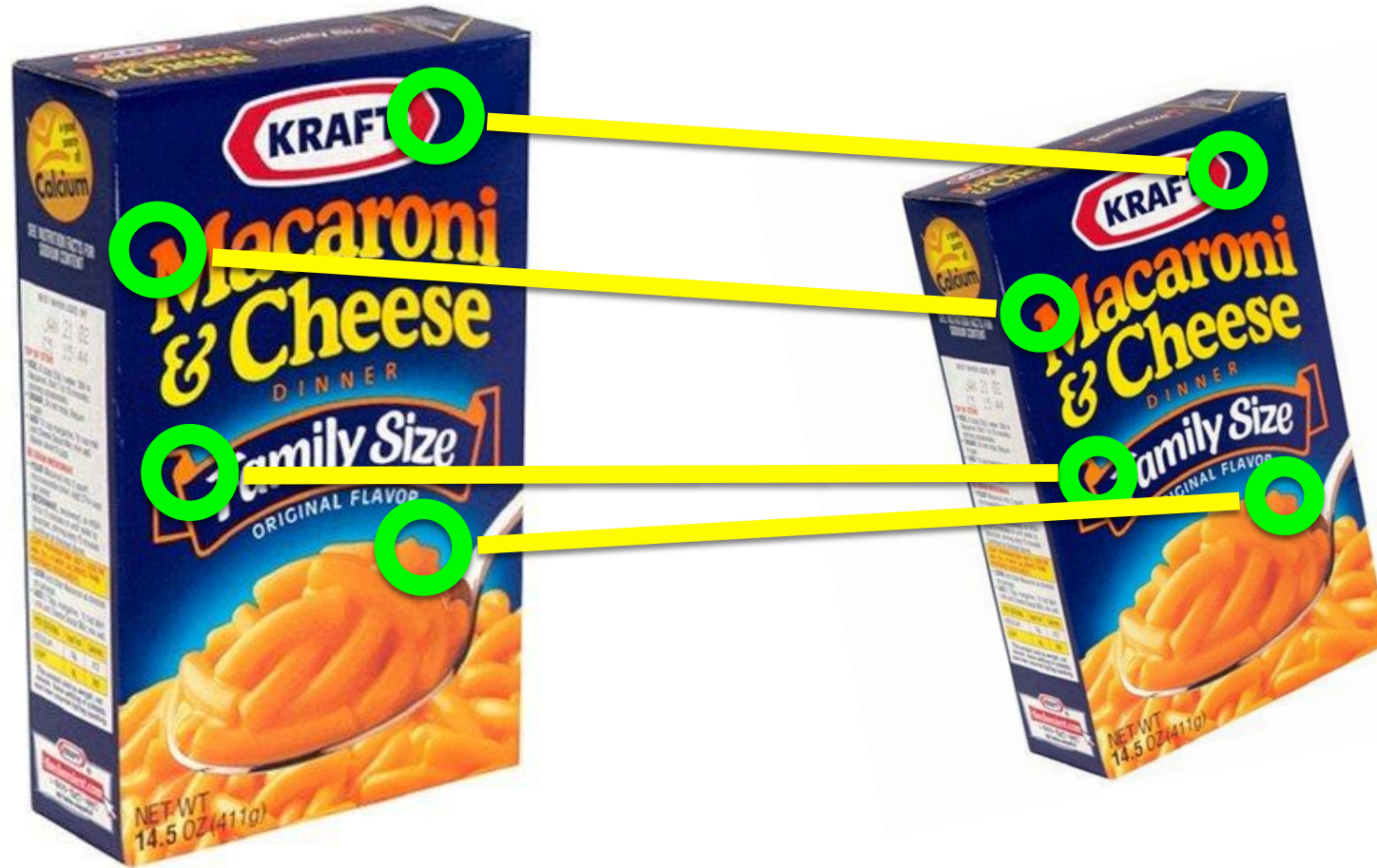
Warping example: feature matching



Warping example: feature matching



Warping example: feature matching

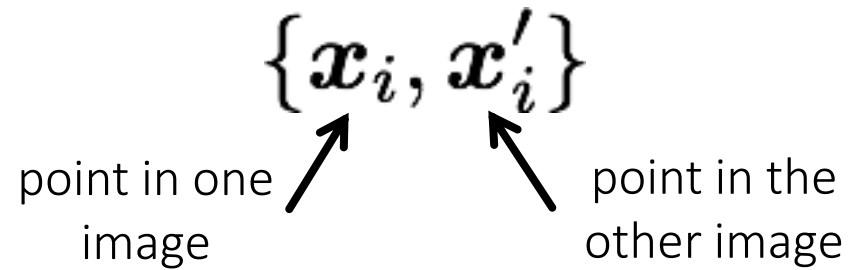


- object recognition
- 3D reconstruction
- augmented reality
- image stitching

How do you compute the transformation?

Warping example: feature matching

Given a set of matched feature points:



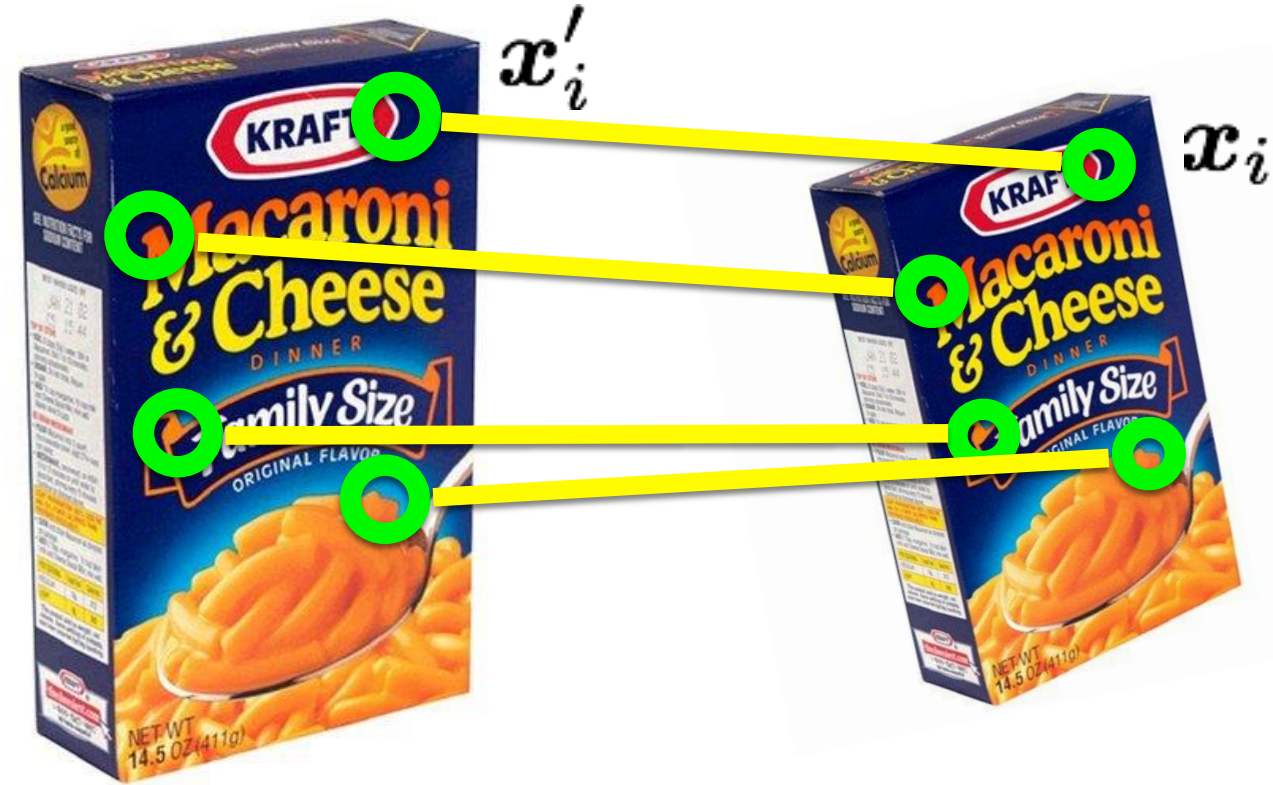
and a transformation:

$$x' = f(x; p)$$

transformation function parameters

find the best estimate of the parameters

p



What kind of transformation functions f are there?

2D transformations

2D transformations



translation



rotation



aspect



affine



perspective



cylindrical

2D planar transformations



2D planar transformations

y



How would you implement scaling?

- Each component multiplied by a scalar
- Uniform scaling - same scalar for each component

x

2D planar transformations

y



$$x' = ax$$

$$y' = by$$

What's the effect of using
different scale factors?

- Each component multiplied by a scalar
- Uniform scaling - same scalar for each component

x

2D planar transformations

y



$$x' = ax$$

$$y' = by$$

matrix representation of scaling:

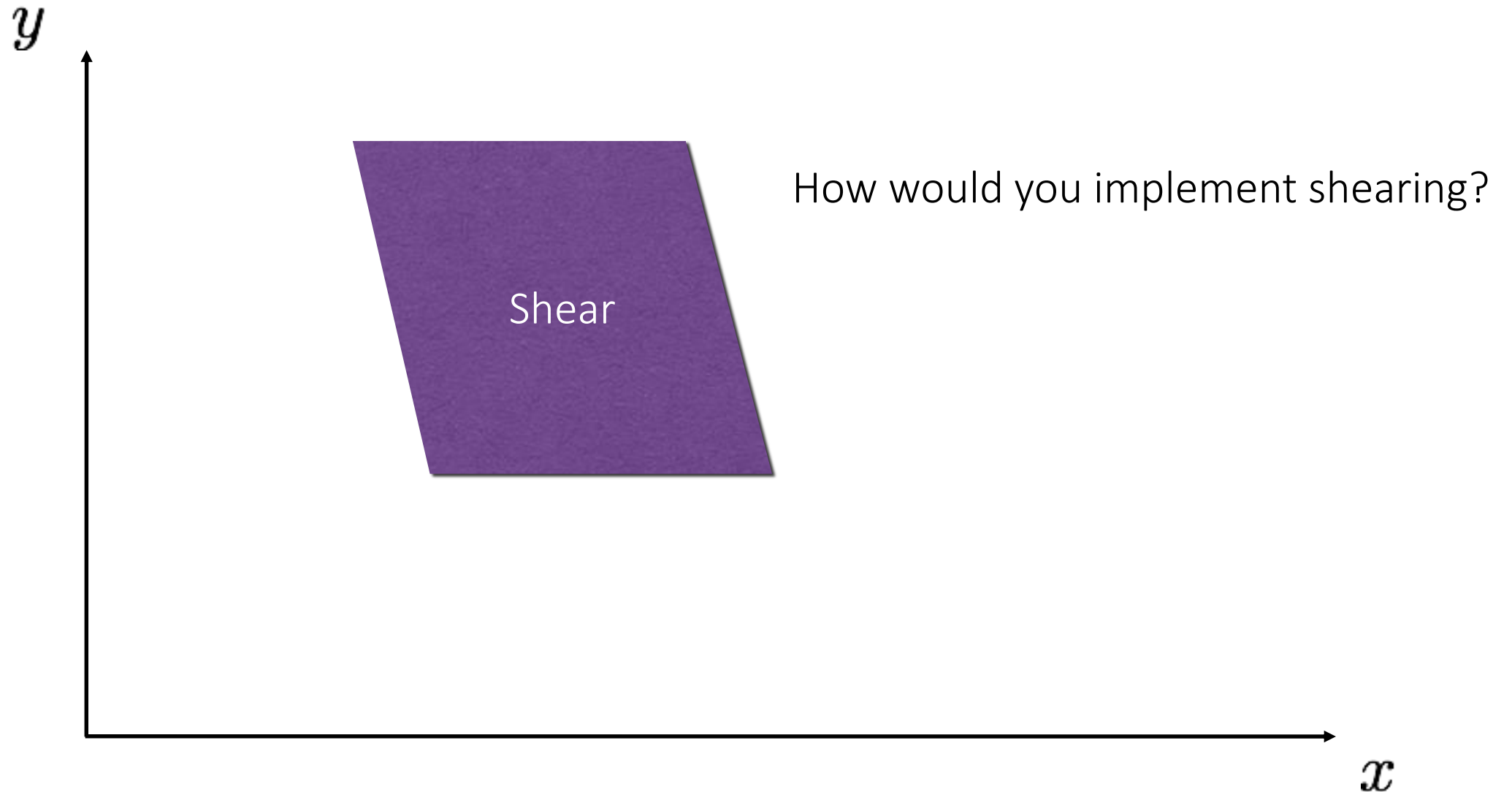
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling matrix } S} \begin{bmatrix} x \\ y \end{bmatrix}$$

scaling matrix S

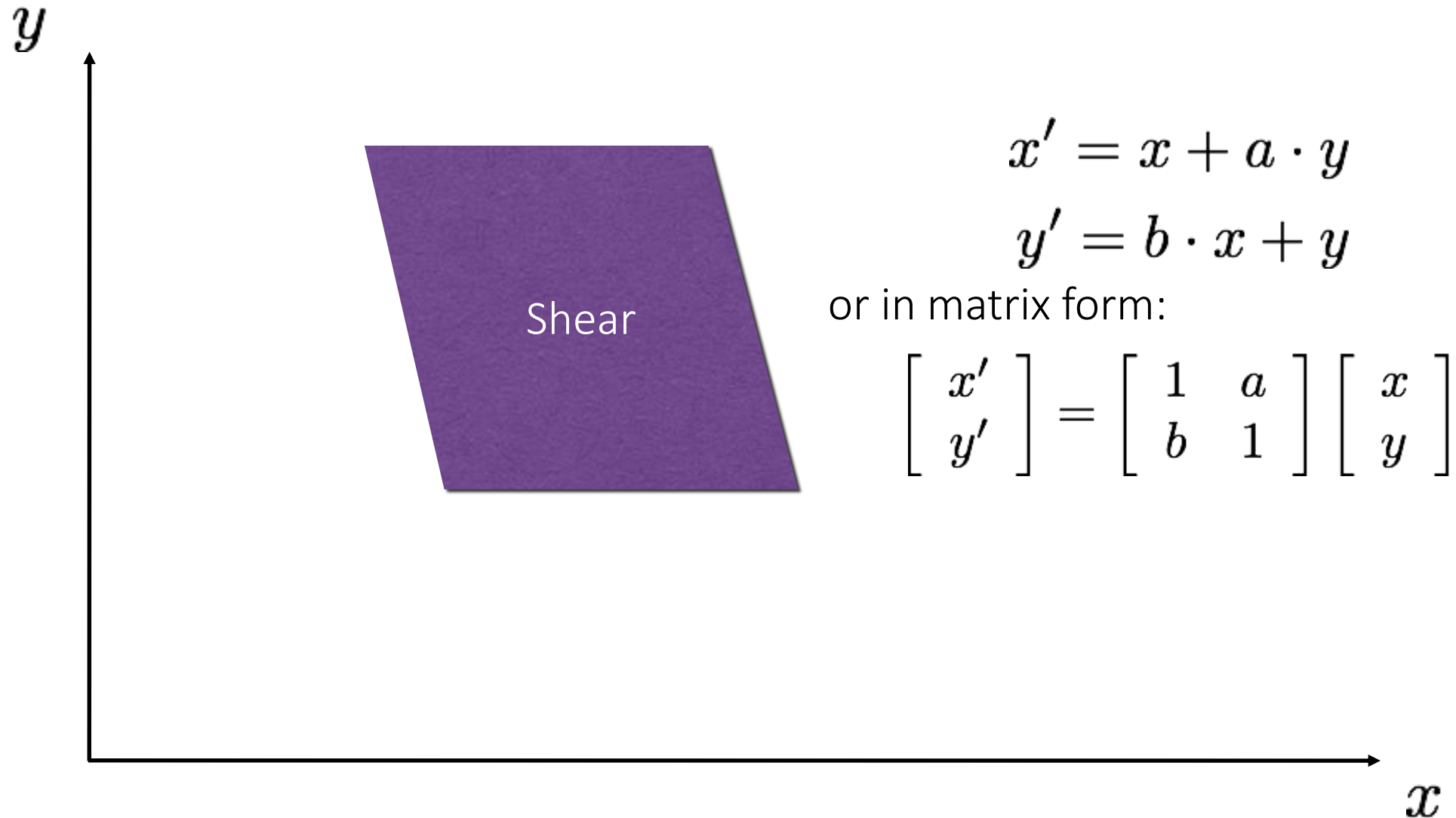
- Each component multiplied by a scalar
- Uniform scaling - same scalar for each component

x

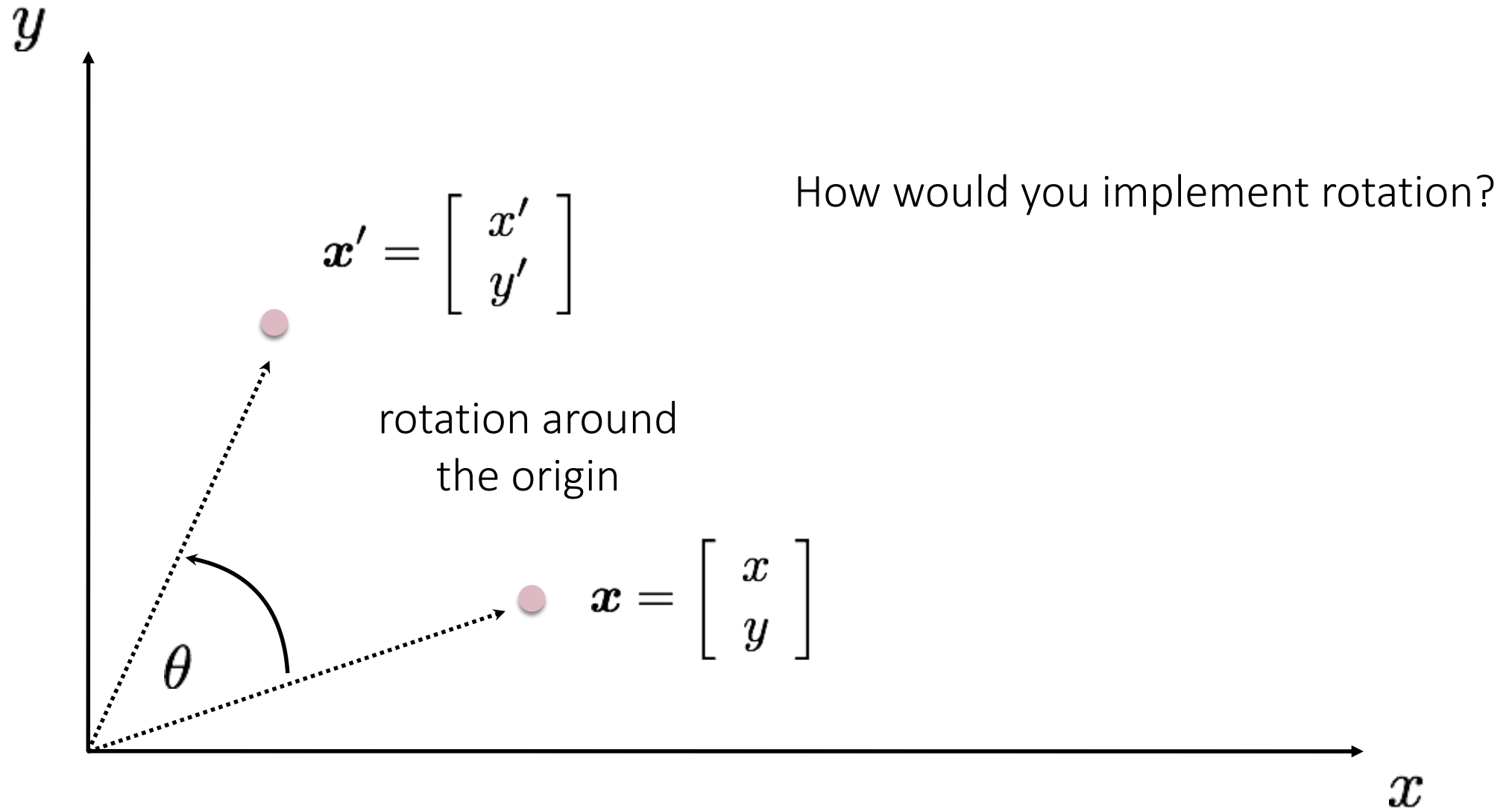
2D planar transformations



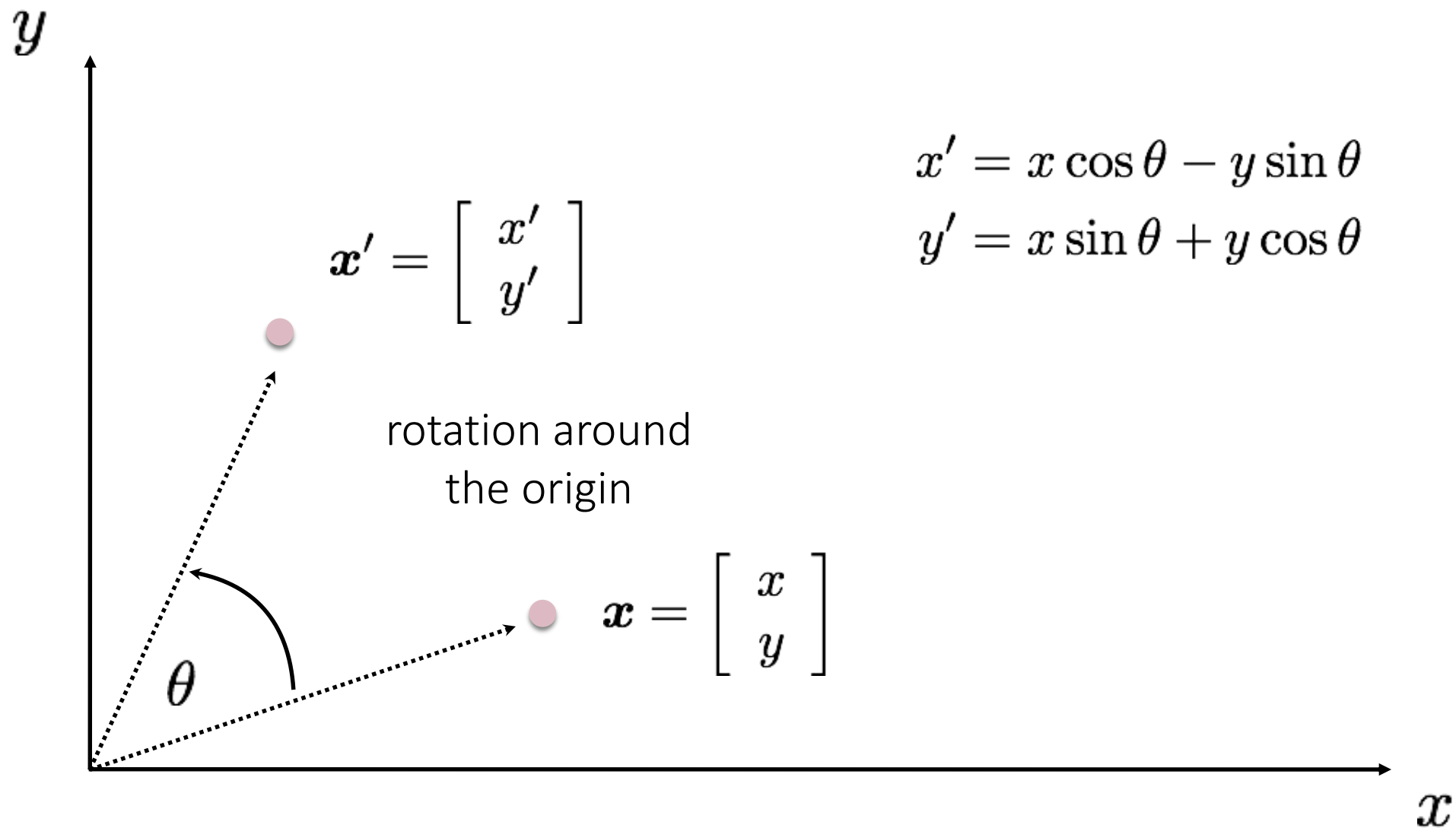
2D planar transformations



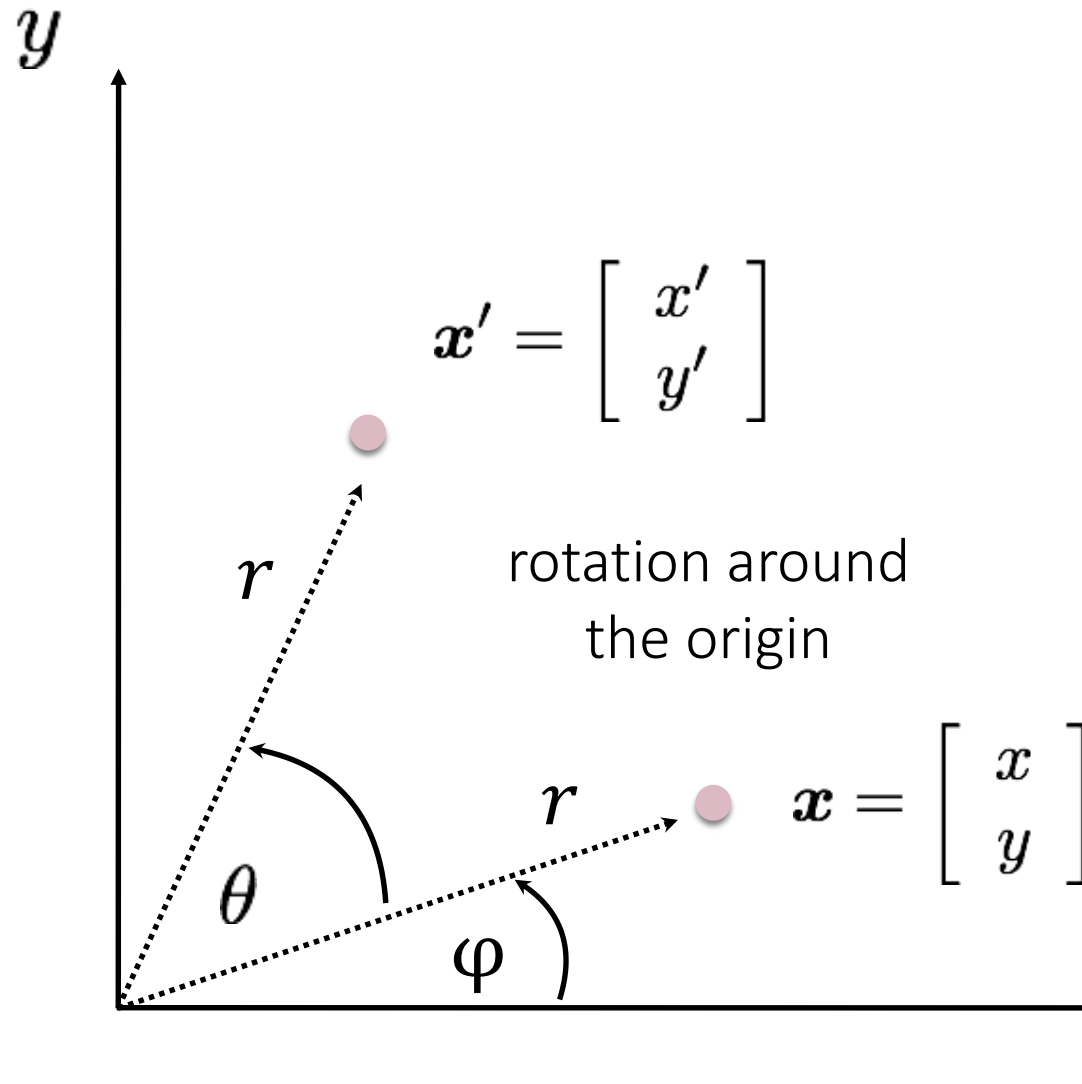
2D planar transformations



2D planar transformations



2D planar transformations



Polar coordinates...

$$x = r \cos(\phi)$$

$$y = r \sin(\phi)$$

$$x' = r \cos(\phi + \theta)$$

$$y' = r \sin(\phi + \theta)$$

Trigonometric Identity...

$$x' = r \cos(\phi) \cos(\theta) - r \sin(\phi) \sin(\theta)$$

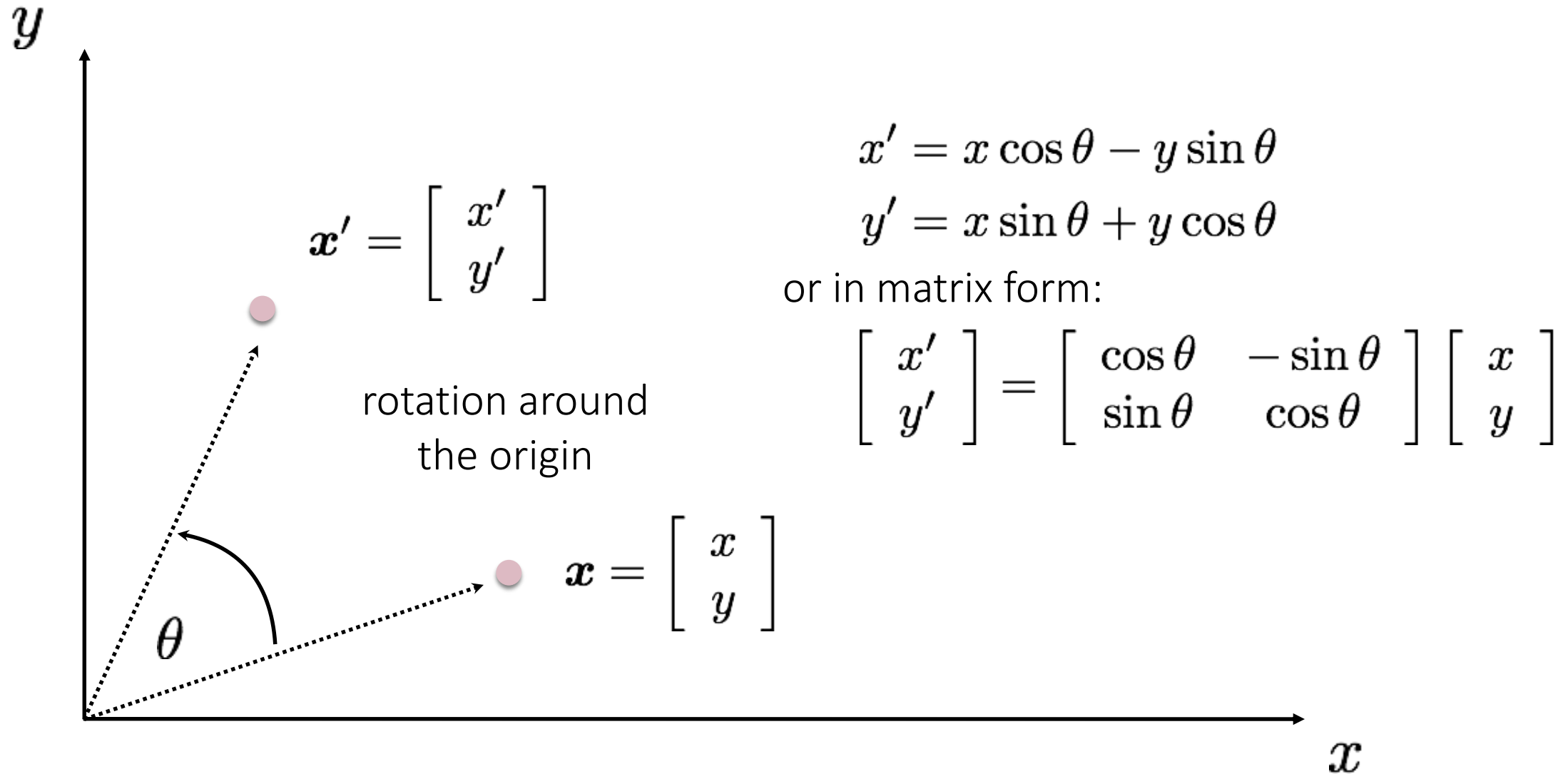
$$y' = r \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta)$$

Substitute...

$$x' = x \cos(\theta) - y \sin(\theta)$$

$$y' = x \sin(\theta) + y \cos(\theta)$$

2D planar transformations



2D planar and linear transformations

$$\boldsymbol{x}' = f(\boldsymbol{x}; p)$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \boldsymbol{M} \begin{bmatrix} x \\ y \end{bmatrix}$$

parameters p

point \boldsymbol{x}

2D planar and linear transformations

Scale

$$\mathbf{M} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

Flip across y

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

Rotate

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

Flip across origin

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

Shear

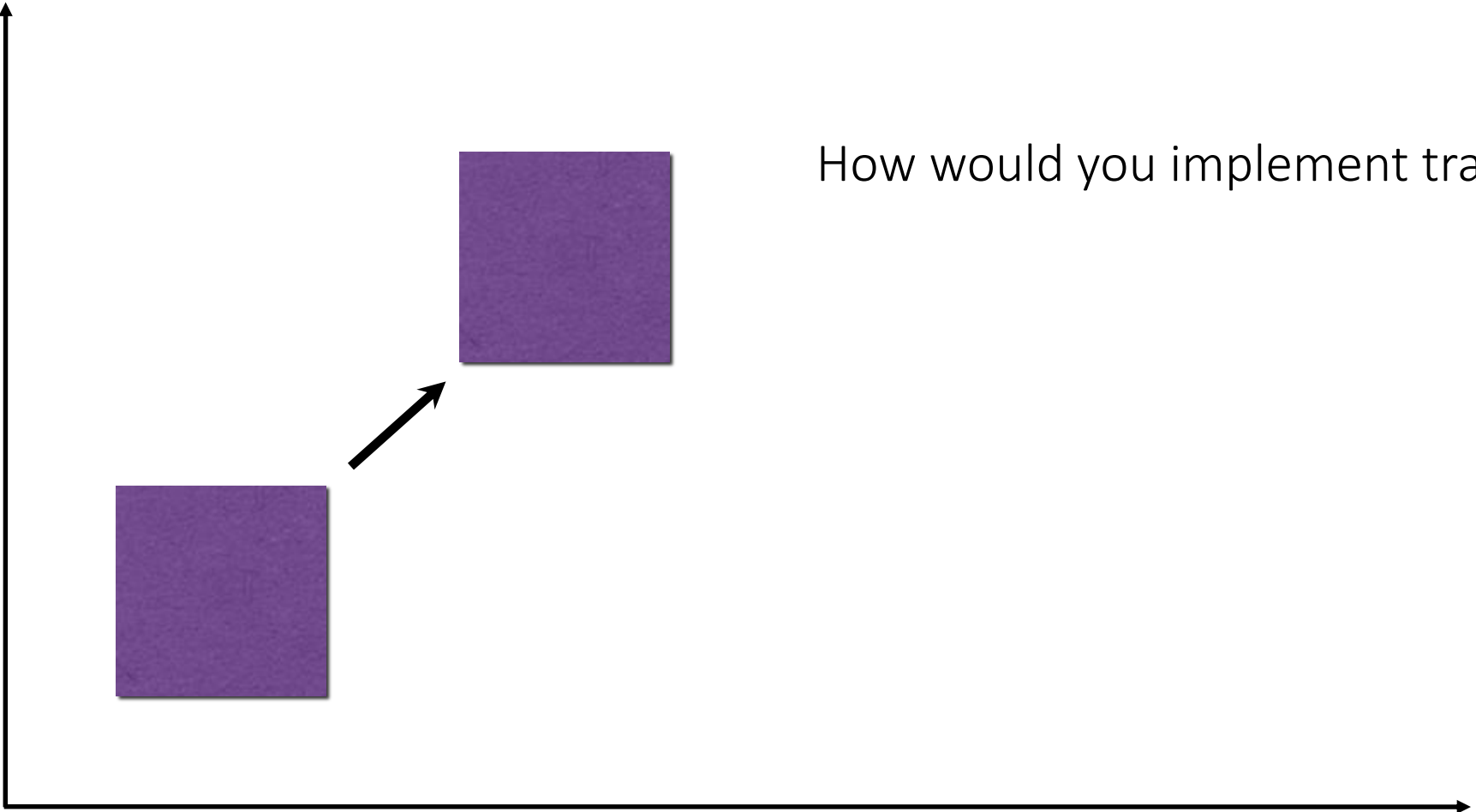
$$\mathbf{M} = \begin{bmatrix} 1 & s_x \\ s_y & 1 \end{bmatrix}$$

Identity

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

2D translation

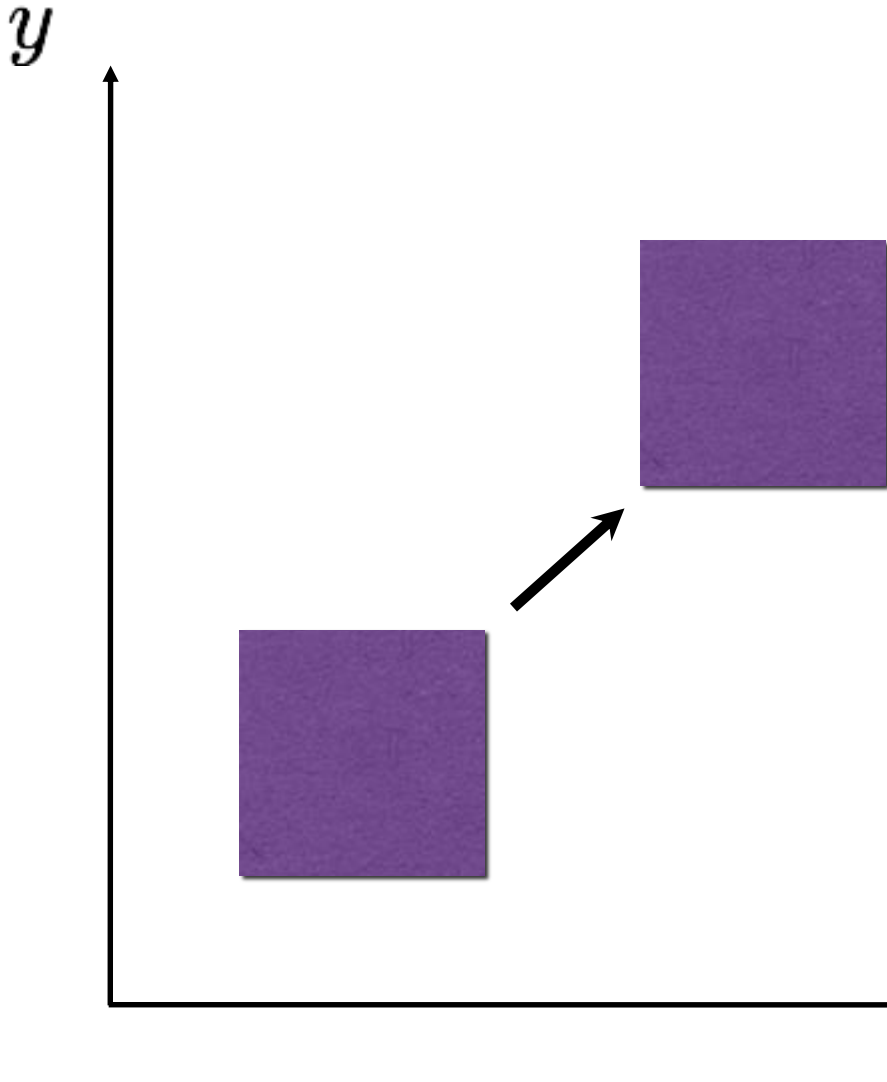
y



How would you implement translation?

x

2D translation



$$x' = x + t_x$$

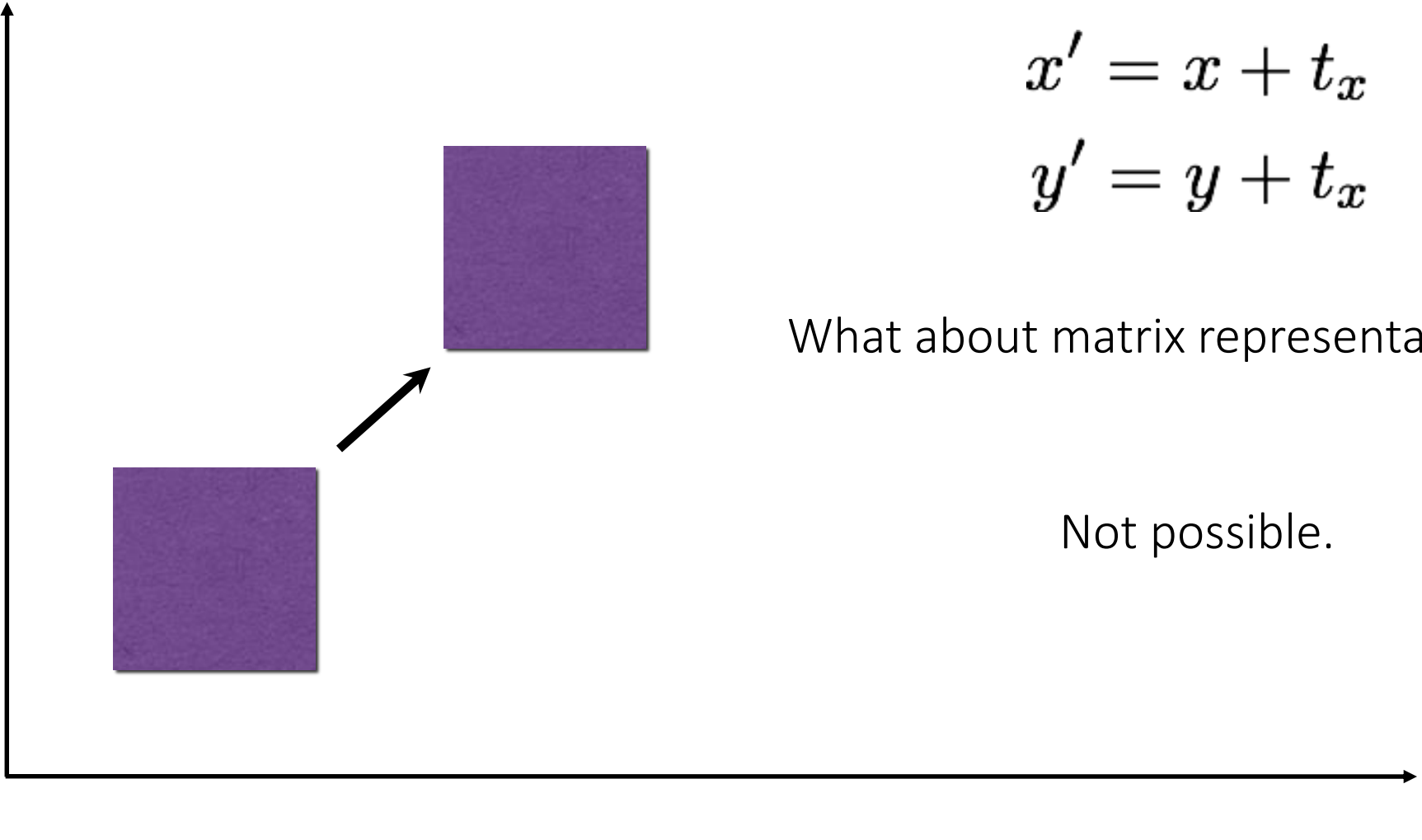
$$y' = y + t_y$$

What about matrix representation?

$$\mathbf{M} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix}$$

2D translation

y



What about matrix representation?

Not possible.

Projective geometry 101

Homogeneous coordinates

heterogeneous
coordinates

homogeneous
coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

← add a 1 here

- Represent 2D point with a 3D vector

Homogeneous coordinates

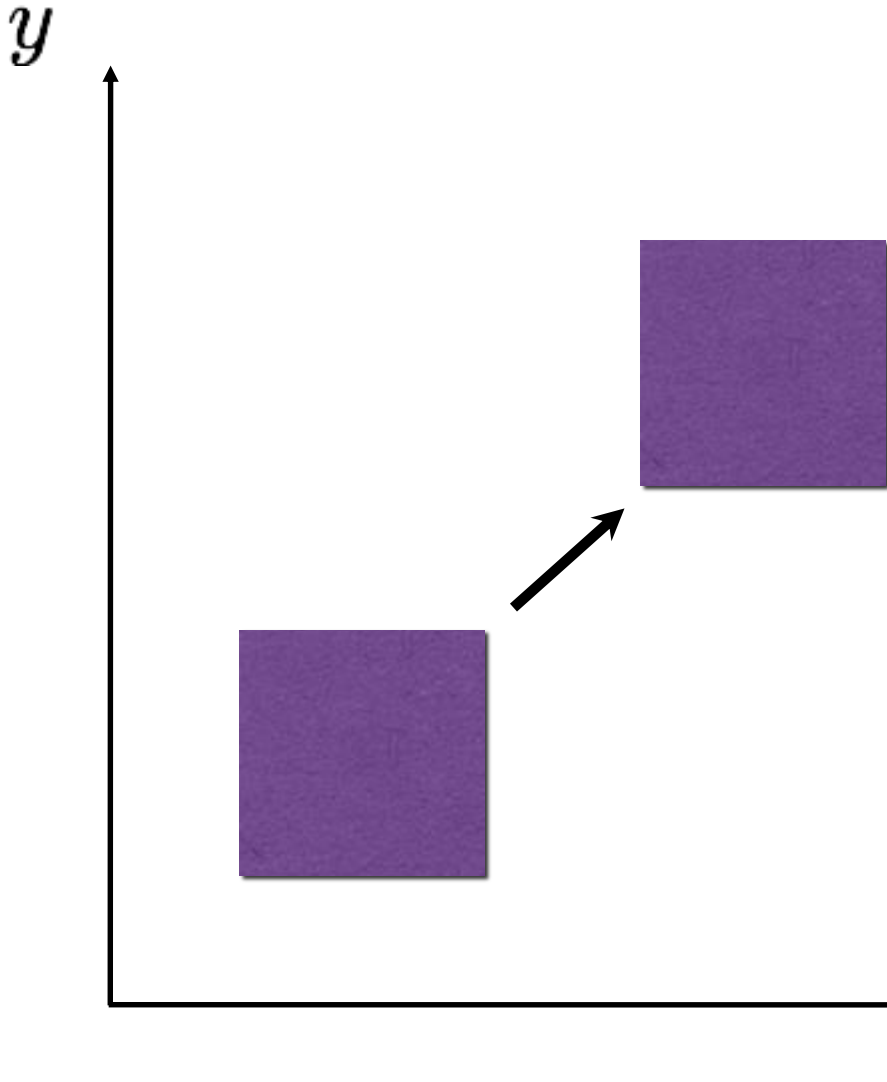
heterogeneous
coordinates

homogeneous
coordinates

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \stackrel{\text{def}}{=} \begin{bmatrix} ax \\ ay \\ a \end{bmatrix}$$

- Represent 2D point with a 3D vector
- 3D vectors are only defined up to scale

2D translation



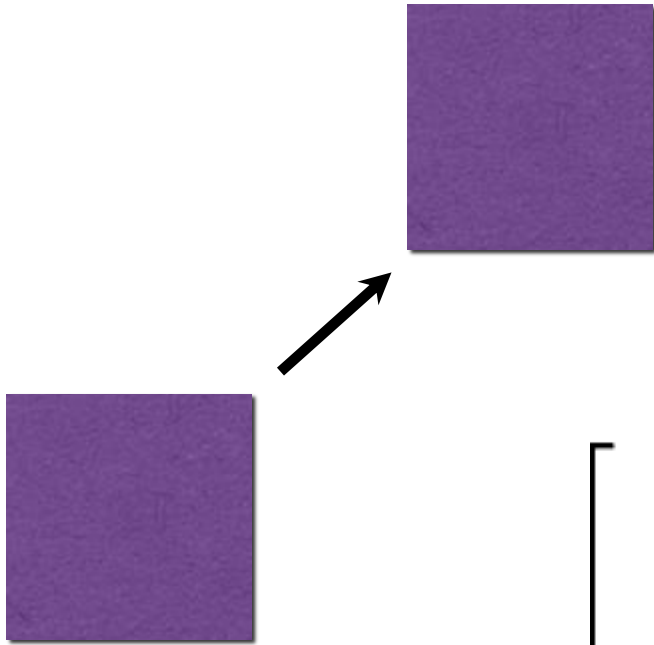
$$x' = x + t_x$$

$$y' = y + t_y$$

What about matrix representation
using homogeneous coordinates?

2D translation

y



$$x' = x + t_x$$

$$y' = y + t_y$$

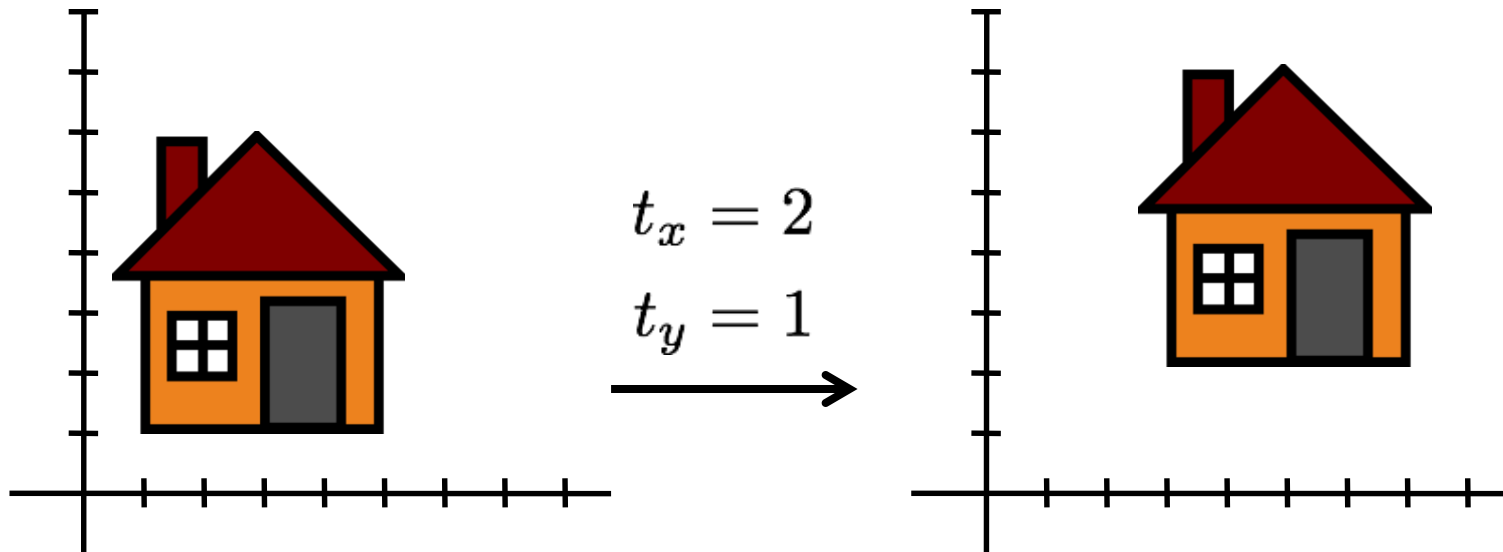
What about matrix representation
using heterogeneous coordinates?

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \mathbf{M} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

x

2D translation using homogeneous coordinates

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



Homogeneous coordinates

Conversion:

- heterogeneous \rightarrow homogeneous

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- homogeneous \rightarrow heterogeneous

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

- scale invariance

$$\begin{bmatrix} x & y & w \end{bmatrix}^{\top} = \lambda \begin{bmatrix} x & y & w \end{bmatrix}^{\top}$$

Special points:

- point at infinity

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

- undefined

$$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

Projective geometry

image point in
pixel coordinates

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

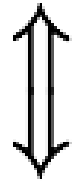
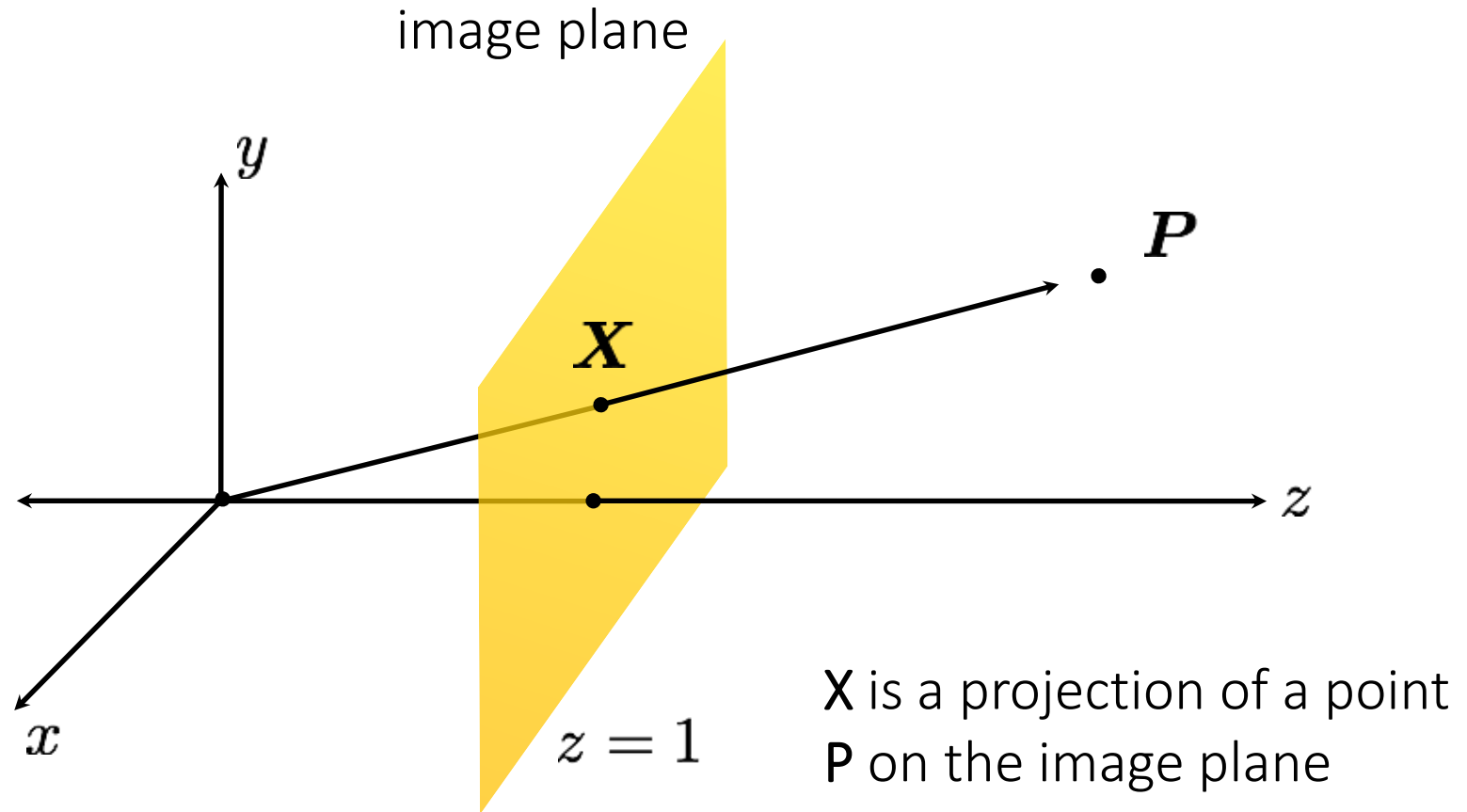


image point in
homogeneous
coordinates

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



What does scaling \mathbf{X} correspond to?

Transformations in projective geometry

2D transformations in heterogeneous coordinates

Re-write these transformations as 3x3 matrices:

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

translation

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

scaling

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

rotation

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

shearing

2D transformations in heterogeneous coordinates

Re-write these transformations as 3x3 matrices:

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

translation

$$\begin{bmatrix} \mathbf{x}' \\ \mathbf{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{s}_x & 0 & 0 \\ 0 & \mathbf{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ 1 \end{bmatrix}$$

scaling

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

rotation

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

shearing

2D transformations in heterogeneous coordinates

Re-write these transformations as 3x3 matrices:

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

translation

$$\begin{bmatrix} \mathbf{x}' \\ \mathbf{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{s}_x & 0 & 0 \\ 0 & \mathbf{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ 1 \end{bmatrix}$$

scaling

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

rotation

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

shearing

2D transformations in heterogeneous coordinates

Re-write these transformations as 3x3 matrices:

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

translation

$$\begin{bmatrix} \mathbf{x}' \\ \mathbf{y}' \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{s}_x & 0 & 0 \\ 0 & \mathbf{s}_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ 1 \end{bmatrix}$$

scaling

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

rotation

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

shearing

Matrix composition

Transformations can be combined by matrix multiplication:

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \Theta & -\sin \Theta & 0 \\ \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$\mathbf{p}' = \quad ? \quad ? \quad ? \quad \mathbf{p}$

Matrix composition

Transformations can be combined by matrix multiplication:

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \Theta & -\sin \Theta & 0 \\ \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

\mathbf{p}' = translation(t_x, t_y)

rotation(θ)

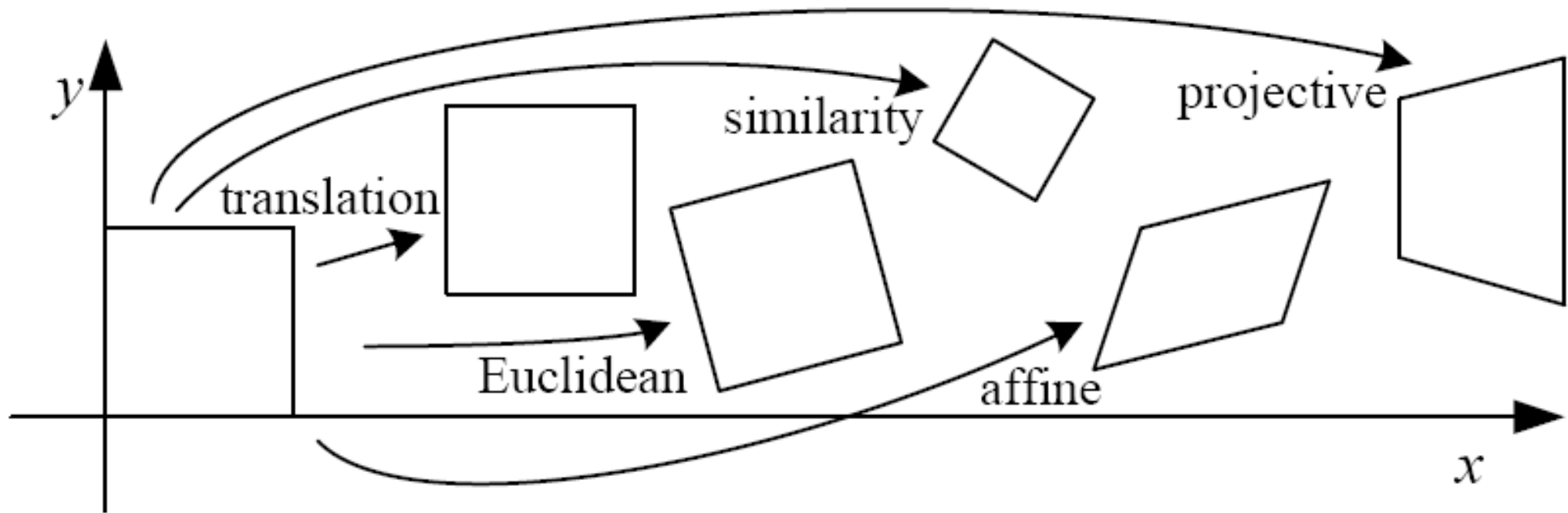
scale(s, s)

\mathbf{p}

Does the multiplication order matter?

Classification of 2D transformations

Classification of 2D transformations



Classification of 2D transformations

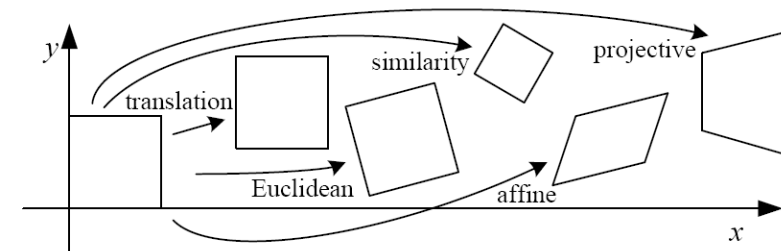
Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \mathbf{I} & & \mathbf{t} \end{bmatrix}$?
rigid (Euclidean)	$\begin{bmatrix} \mathbf{R} & & \mathbf{t} \end{bmatrix}$?
similarity	$\begin{bmatrix} s\mathbf{R} & & \mathbf{t} \end{bmatrix}$?
affine	$\begin{bmatrix} \mathbf{A} \end{bmatrix}$?
projective	$\begin{bmatrix} \tilde{\mathbf{H}} \end{bmatrix}$?

Classification of 2D transformations

Translation:

$$\begin{bmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom?

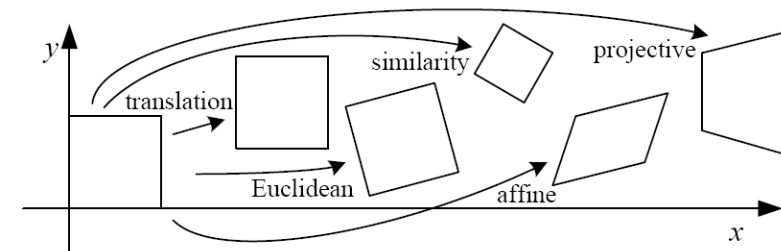


Classification of 2D transformations

Euclidean (rigid):
rotation + translation

$$\begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Are there any values that are related?

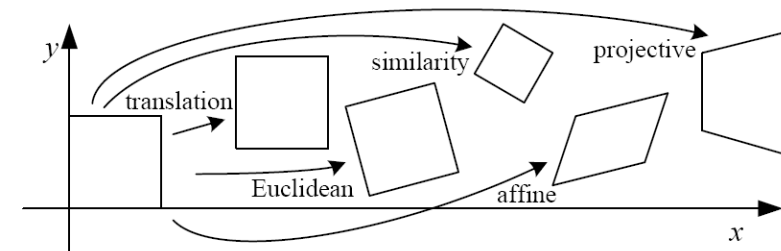


Classification of 2D transformations

Euclidean (rigid):
rotation + translation

$$\begin{bmatrix} \cos \theta & -\sin \theta & r_3 \\ \sin \theta & \cos \theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom?

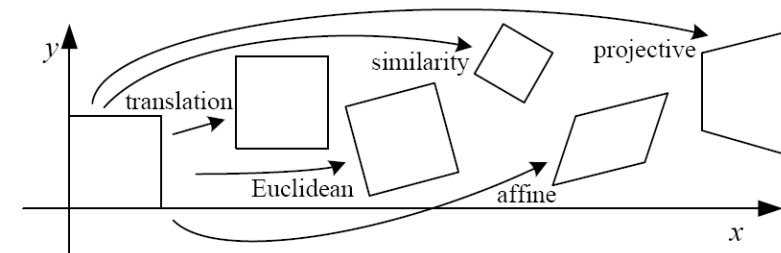


Classification of 2D transformations

which other matrix values will
change if this increases?

Euclidean (rigid):
rotation + translation

$$\begin{bmatrix} \cos \theta & -\sin \theta & r_3 \\ \sin \theta & \cos \theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$



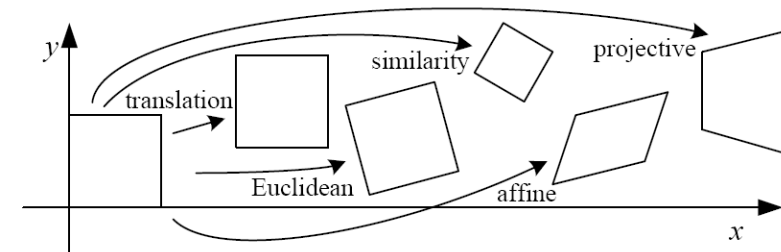
Classification of 2D transformations

what will happen to the
image if this increases?

Euclidean (rigid):
rotation + translation

↓

$$\begin{bmatrix} \cos \theta & -\sin \theta & r_3 \\ \sin \theta & \cos \theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$



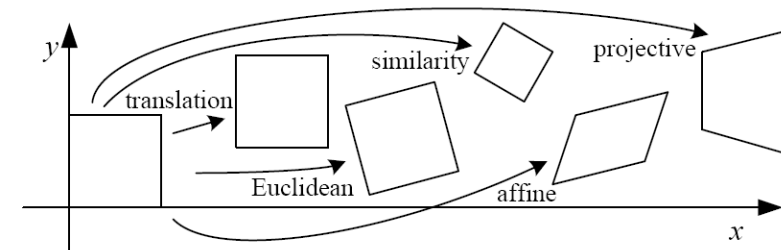
Classification of 2D transformations

what will happen to the
image if this increases?



Euclidean (rigid):
rotation + translation

$$\begin{bmatrix} \cos \theta & -\sin \theta & r_3 \\ \sin \theta & \cos \theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

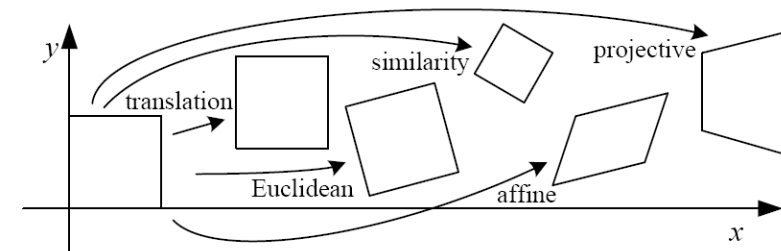


Classification of 2D transformations

Similarity:
uniform scaling + rotation
+ translation

$$\begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Are there any values that are related?



Classification of 2D transformations

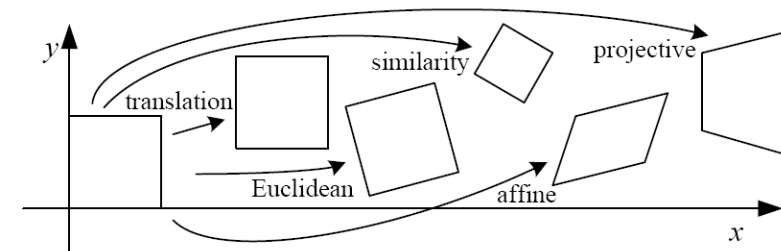
multiply these four by scale s



Similarity:
uniform scaling + rotation
+ translation

$$\begin{bmatrix} \cos \theta & -\sin \theta & r_3 \\ \sin \theta & \cos \theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom?

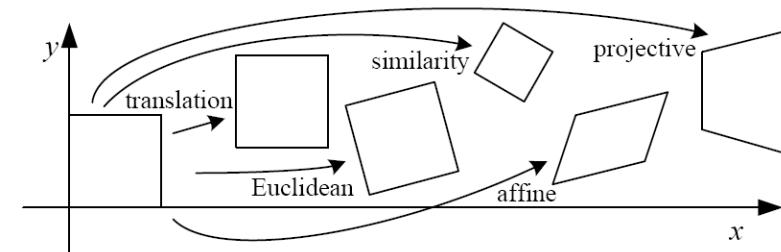


Classification of 2D transformations

what will happen to the
image if this increases?

Similarity:
uniform scaling + rotation
+ translation

$$\begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

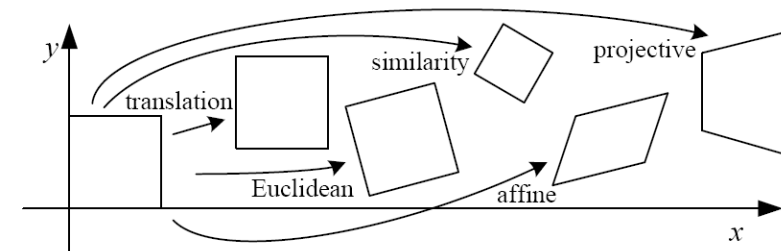


Classification of 2D transformations

Affine transform:
uniform scaling + shearing
+ rotation + translation

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Are there any values that are related?



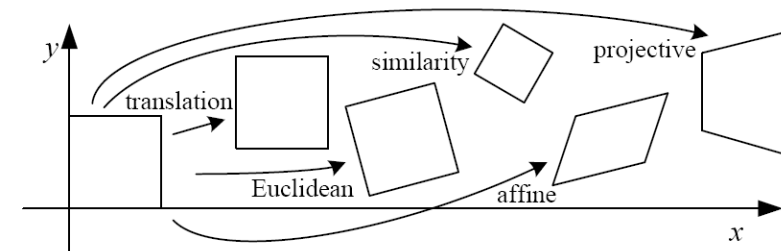
Classification of 2D transformations

Affine transform:
uniform scaling + shearing
+ rotation + translation

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Are there any values that are related?

$$\begin{array}{cc} \text{similarity} & \text{shear} \\ \begin{bmatrix} sr_1 & sr_2 \\ sr_3 & sr_4 \end{bmatrix} & \begin{bmatrix} 1 & h_1 \\ h_2 & 1 \end{bmatrix} \end{array} = \begin{bmatrix} sr_1 + h_2 sr_2 & sr_2 + h_1 sr_1 \\ sr_3 + h_2 sr_4 & sr_4 + h_1 sr_3 \end{bmatrix}$$



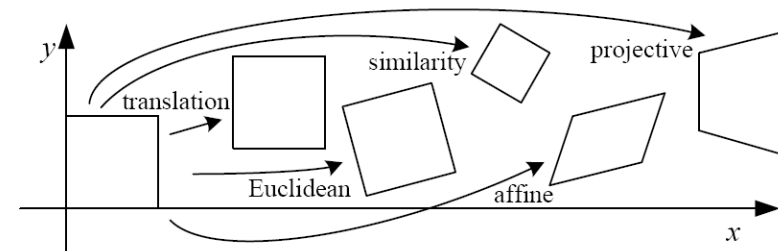
Classification of 2D transformations

Affine transform:
uniform scaling + shearing
+ rotation + translation

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom?

$$\begin{array}{cc} \text{similarity} & \text{shear} \\ \begin{bmatrix} sr_1 & sr_2 \\ sr_3 & sr_4 \end{bmatrix} & \begin{bmatrix} 1 & h_1 \\ h_2 & 1 \end{bmatrix} \end{array} = \begin{bmatrix} sr_1 + h_2 sr_2 & sr_2 + h_1 sr_1 \\ sr_3 + h_2 sr_4 & sr_4 + h_1 sr_3 \end{bmatrix}$$



Affine transformations

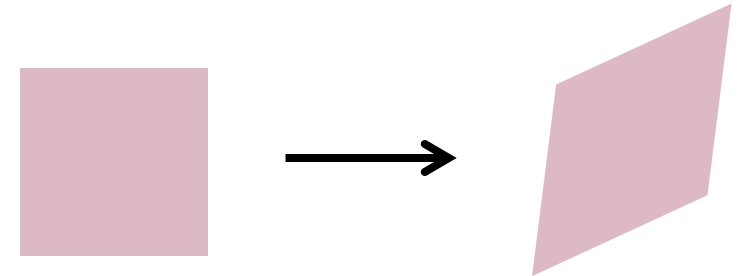
Affine transformations are combinations of

- arbitrary (4-DOF) 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 map to parallel lines
- ratios are preserved
- compositions of affine transforms are also affine transforms



Does the last coordinate w ever change?

Affine transformations

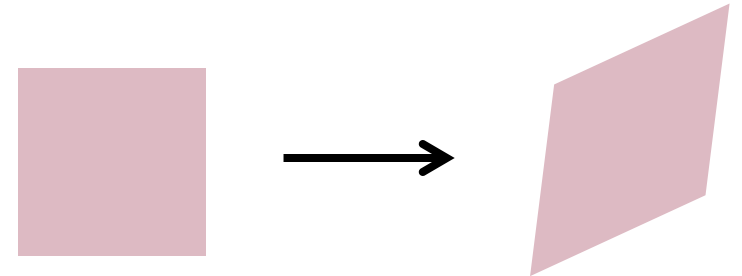
Affine transformations are combinations of

- arbitrary (4-DOF) 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 map to parallel lines
- ratios are preserved
- compositions of affine transforms are also affine transforms



Nope! But what does that mean?

How to interpret affine transformations here?

image point in
pixel coordinates

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

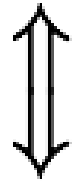
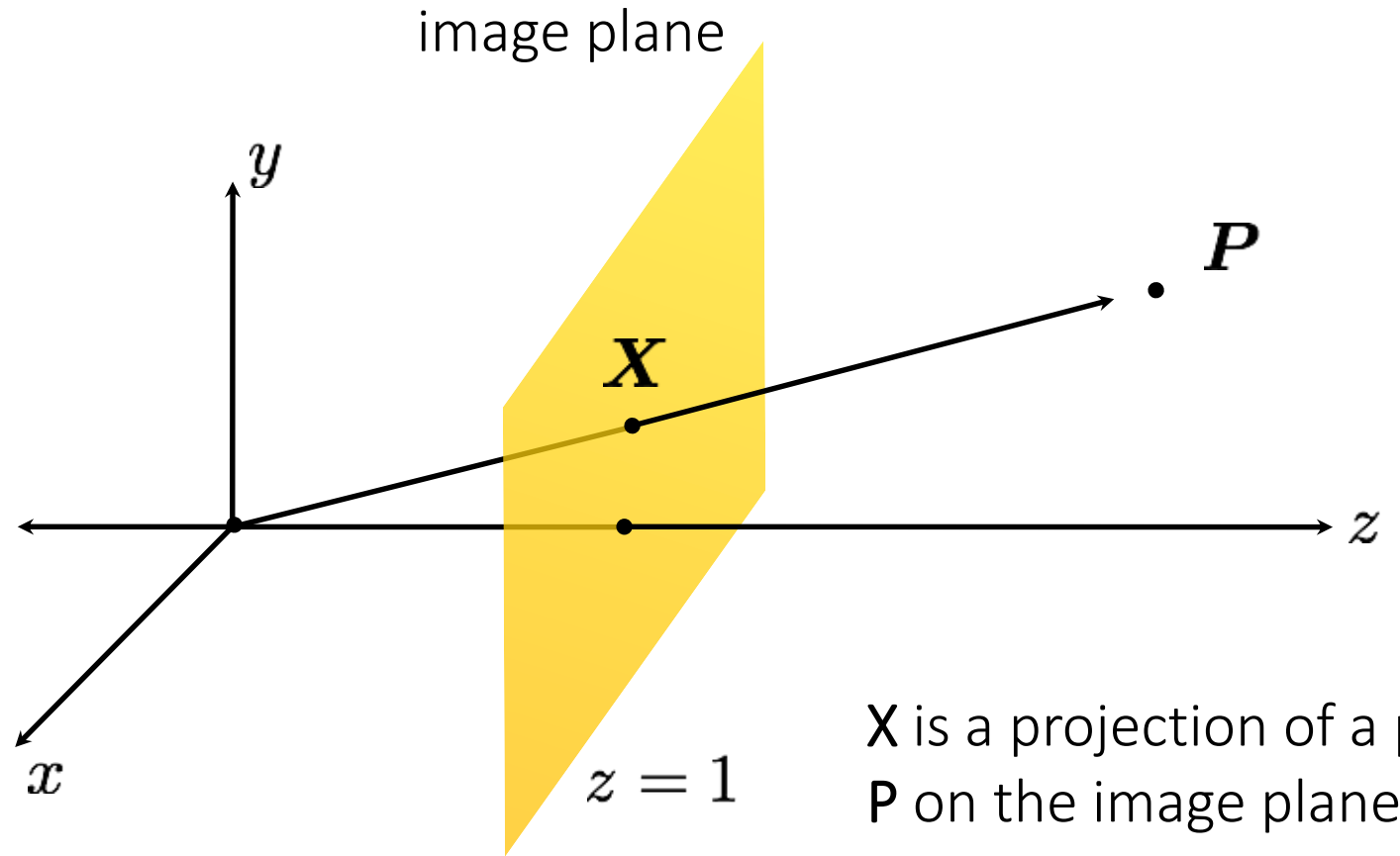


image point in
heterogeneous
coordinates

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



\mathbf{X} is a projection of a point
 \mathbf{P} on the image plane

Projective transformations

Projective transformations are combinations of

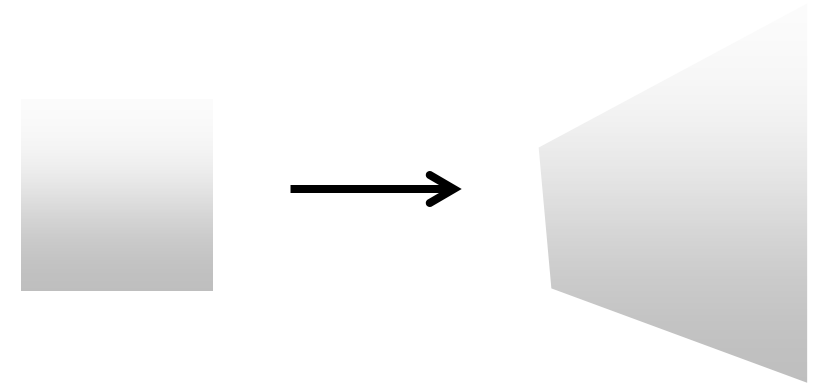
- affine transformations; and
- projective wraps

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

How many degrees of freedom?

Properties of projective transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines do not necessarily map to parallel lines
- ratios are not necessarily preserved
- compositions of projective transforms are also projective transforms



Projective transformations

Projective transformations are combinations of

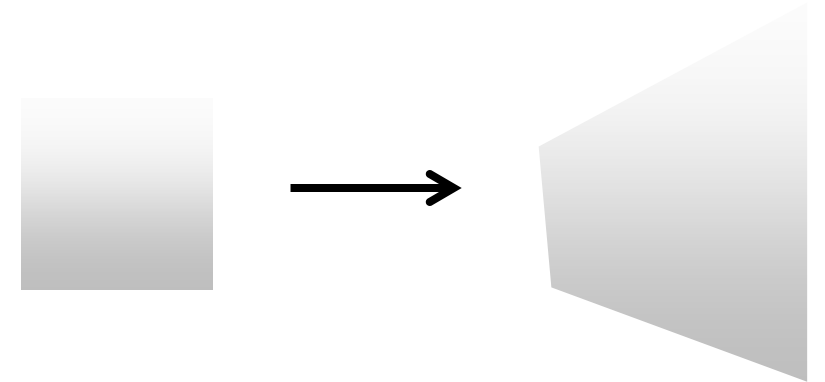
- affine transformations; and
- projective wraps

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

Properties of projective transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines do not necessarily map to parallel lines
- ratios are not necessarily preserved
- compositions of projective transforms are also projective transforms

8 DOF: vectors (and therefore matrices) are defined up to scale)



How to interpret projective transformations here?

image point in
pixel coordinates

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

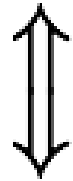
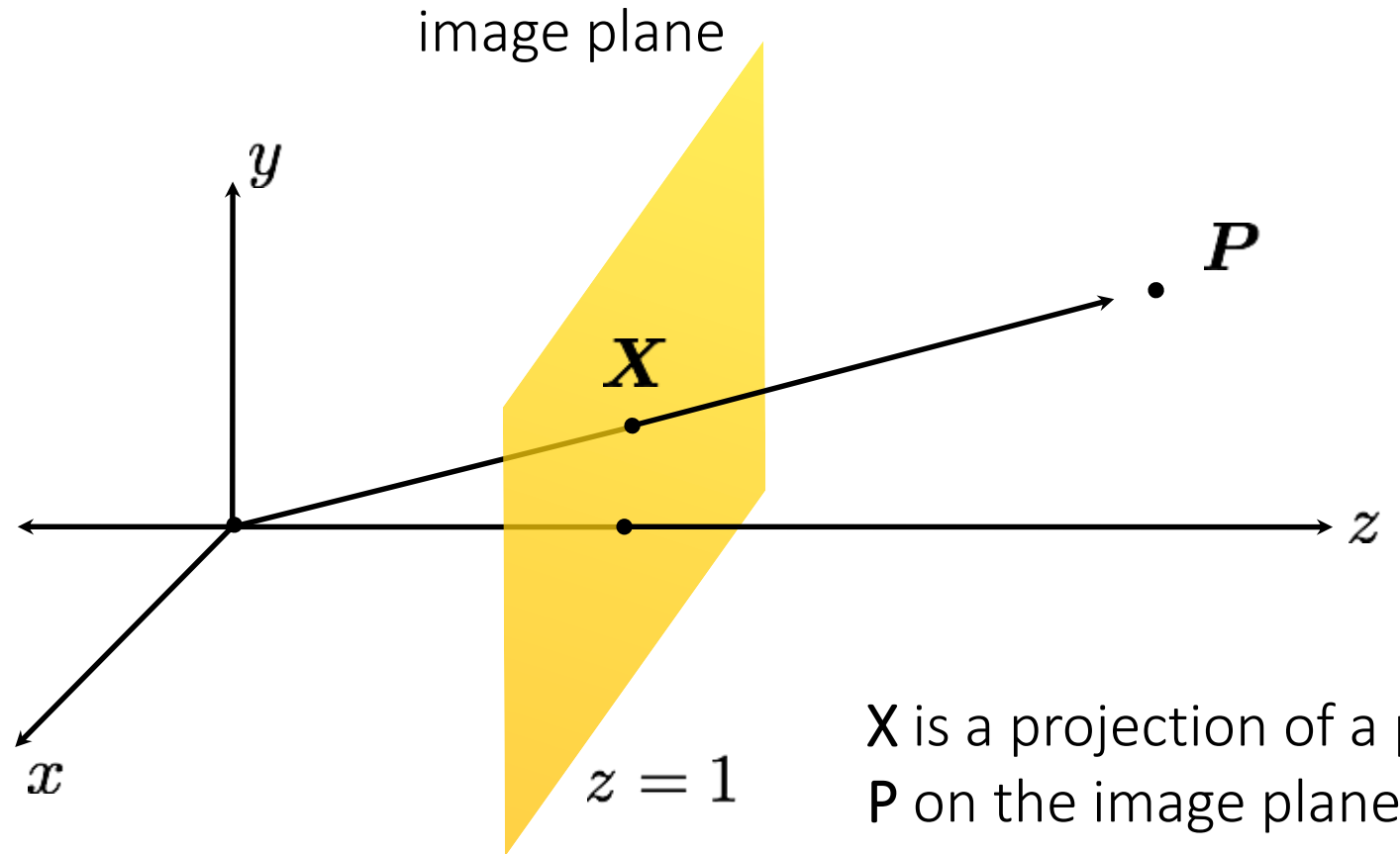


image point in
heterogeneous
coordinates

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

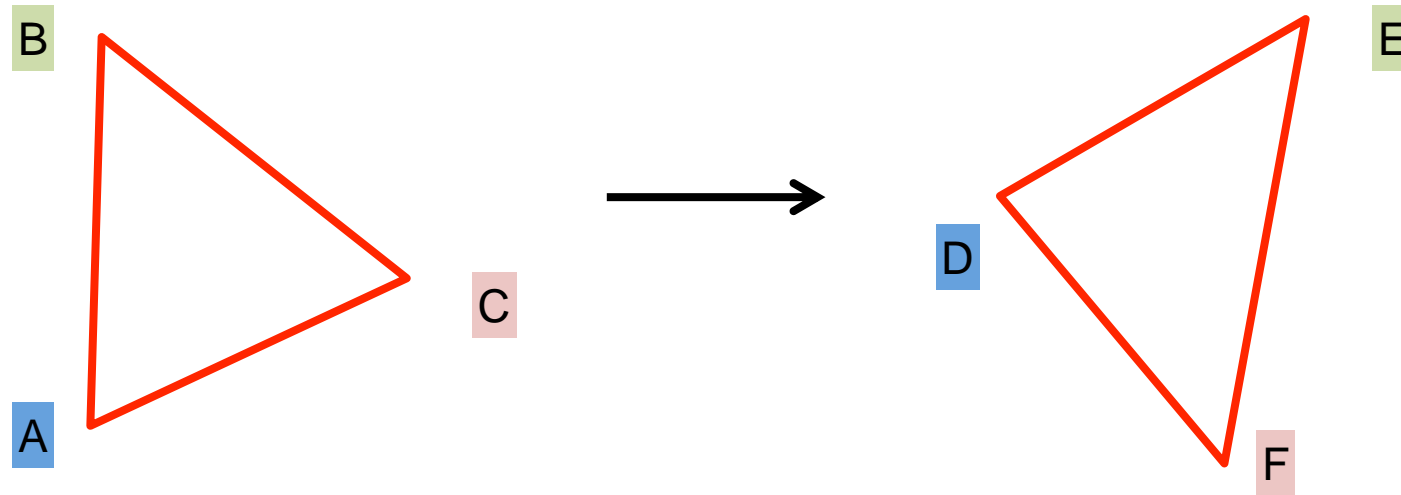


\mathbf{X} is a projection of a point
 \mathbf{P} on the image plane

Determining unknown 2D transformations

Determining unknown transformations

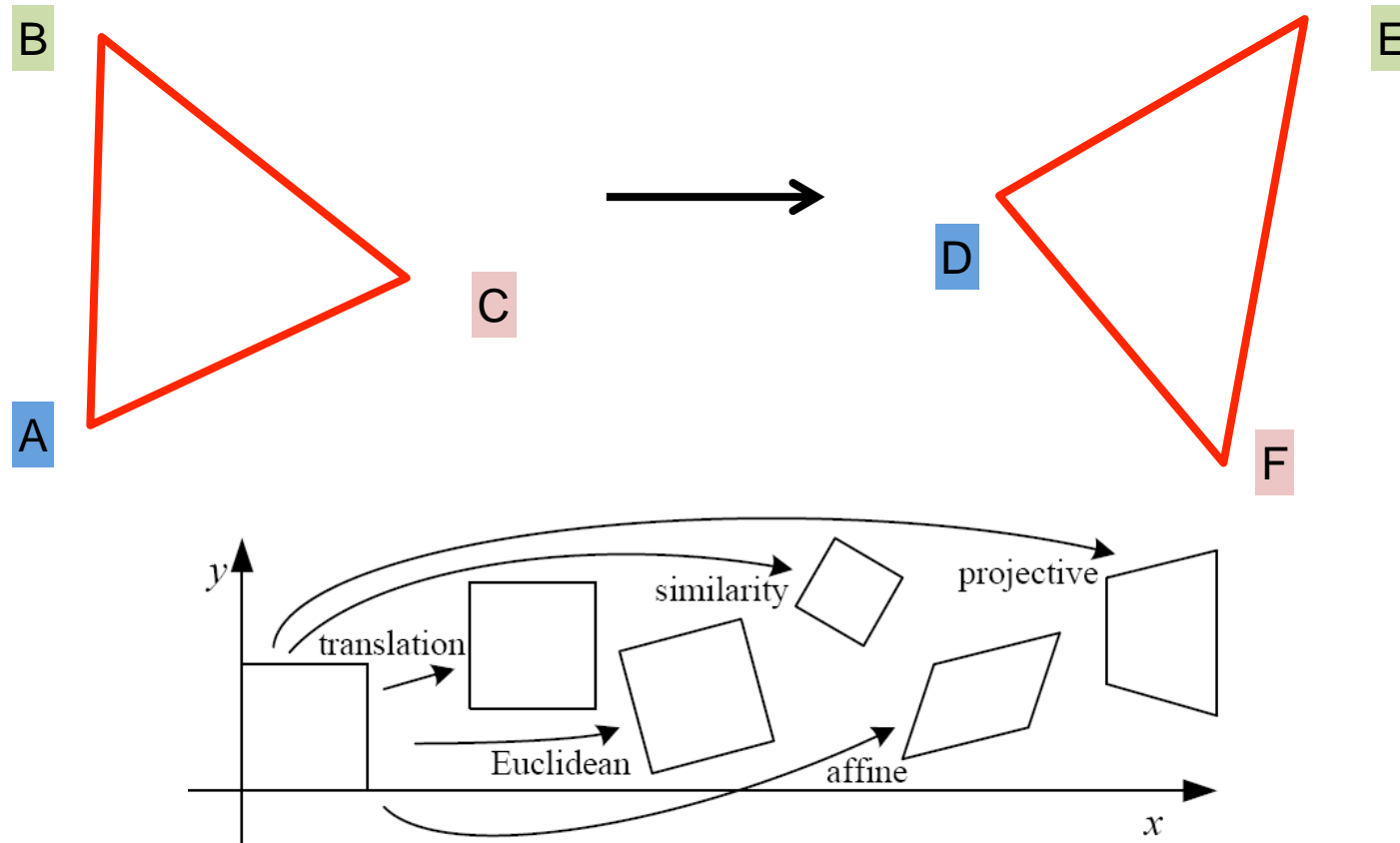
Suppose we have two triangles: ABC and DEF.



Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

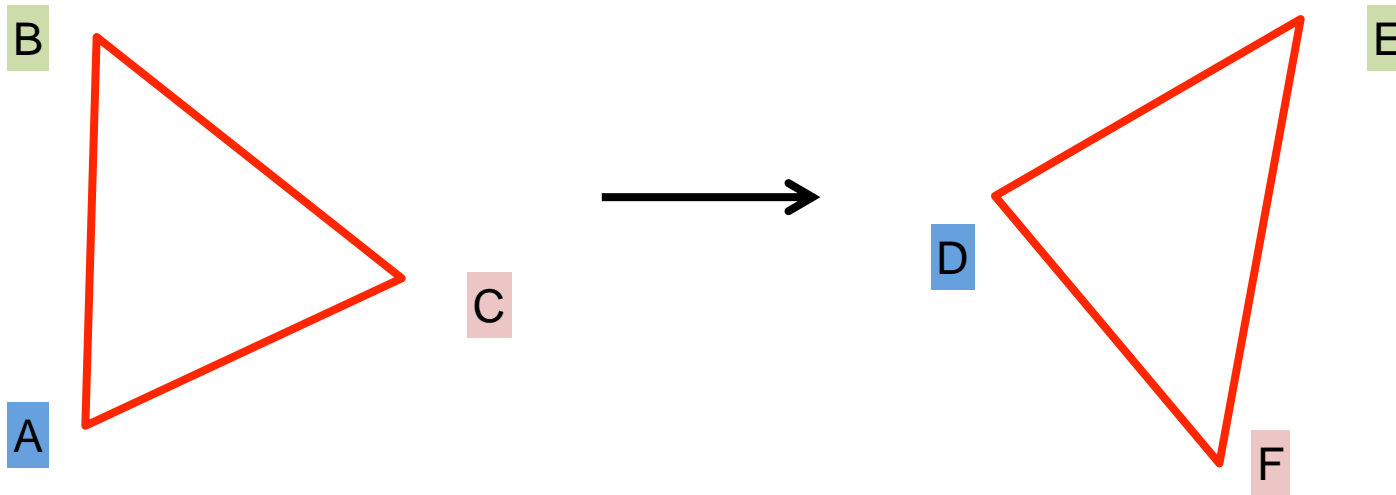
- What type of transformation will map A to D, B to E, and C to F?



Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?
- How do we determine the unknown parameters?



Affine transform:
uniform scaling + shearing
+ rotation + translation

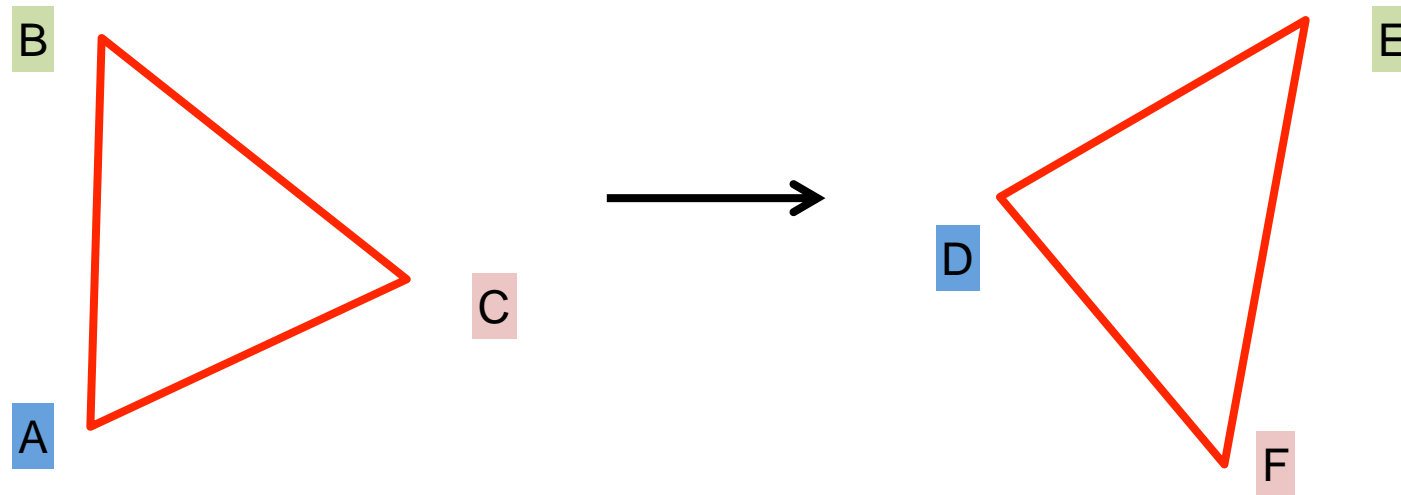
$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom do we have?

Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?
- How do we determine the unknown parameters?



unknowns

$$x' = Mx$$

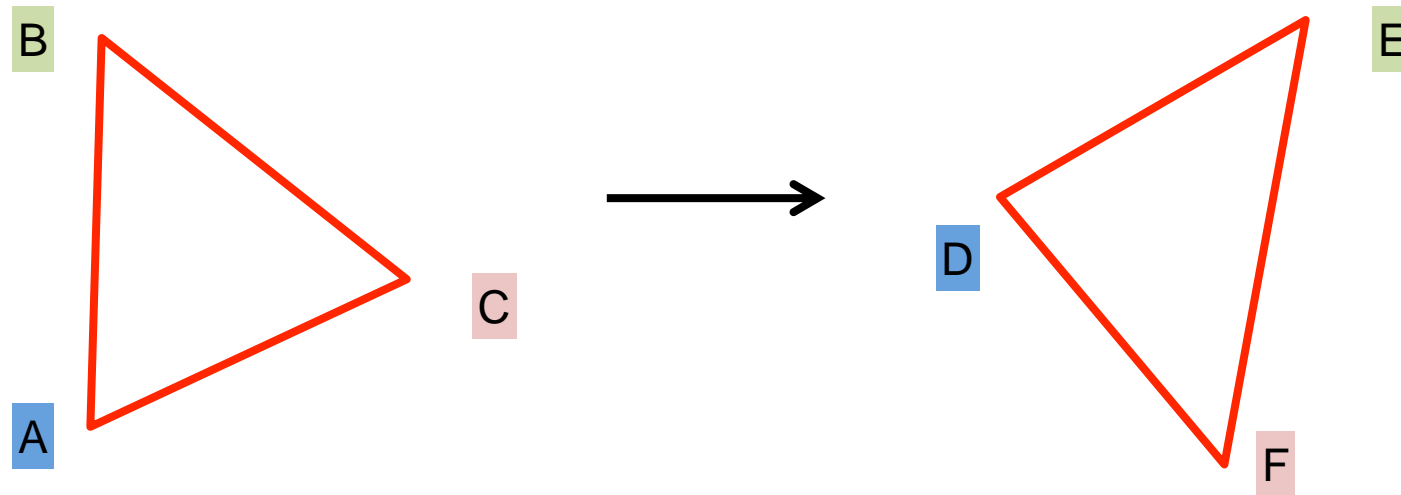
point correspondences

- One point correspondence gives how many equations?
- How many point correspondences do we need?

Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

- What type of transformation will map A to D, B to E, and C to F?
- How do we determine the unknown parameters?

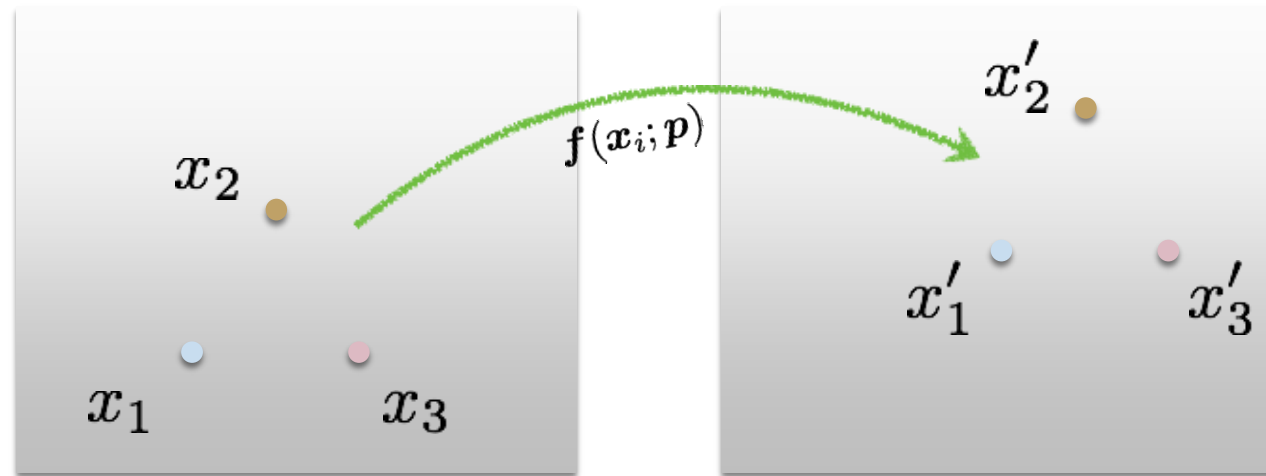


unknowns

$$x' = Mx$$

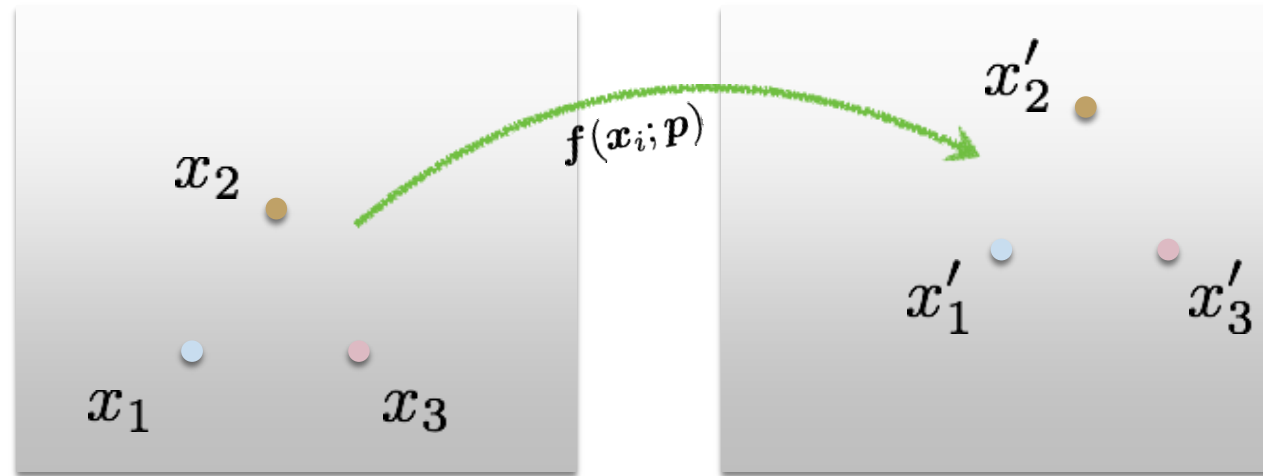
point correspondences

How do we solve this for **M**?



Least Squares Error

$$E_{\text{LS}} = \sum_i \| \mathbf{f}(\mathbf{x}_i; \mathbf{p}) - \mathbf{x}'_i \|^2$$



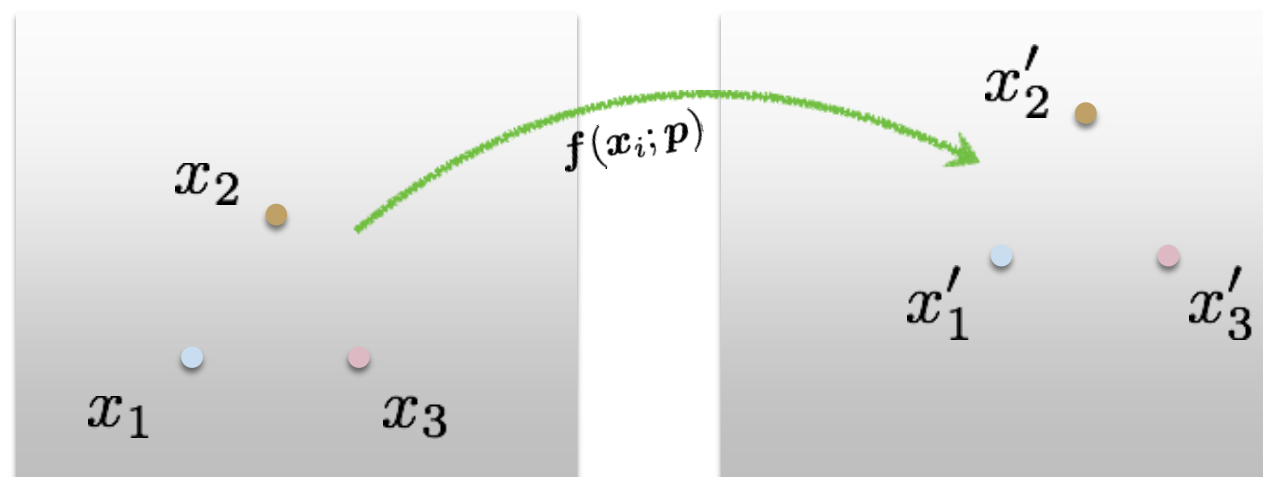
Least Squares Error

$$E_{\text{LS}} = \sum_i \left\| \mathbf{f}(\mathbf{x}_i; \mathbf{p}) - \mathbf{x}'_i \right\|^2$$

What is this?

What is this?

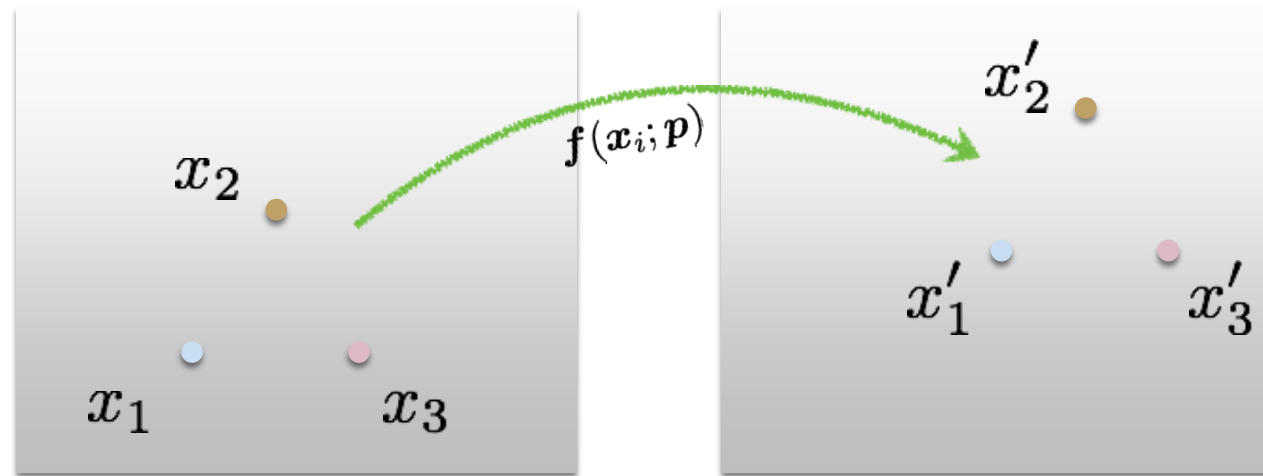
What is this?



$$\|x\| = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

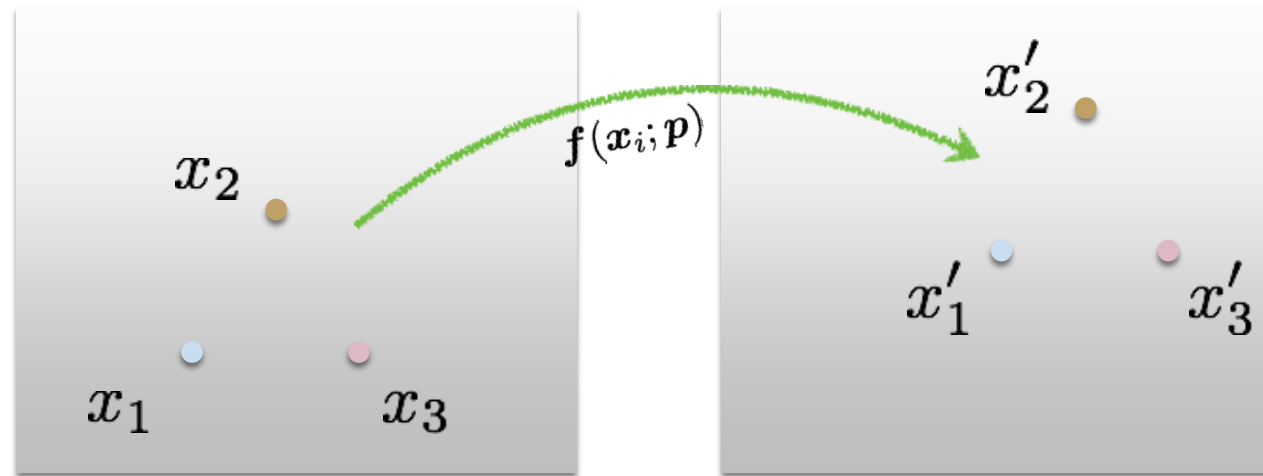
Least Squares Error

$$E_{\text{LS}} = \sum_i \left\| \underset{\substack{\uparrow \\ \text{predicted} \\ \text{location}}}{f(x_i; p)} - \underset{\substack{\uparrow \\ \text{measured} \\ \text{location}}}{x'_i} \right\| \overset{\substack{\text{Euclidean} \\ \text{(L2) norm}}}{\text{squared!}}^2$$



Least Squares Error

$$E_{\text{LS}} = \sum_i \underbrace{\|f(x_i; p) - x'_i\|}_{\text{Residual (projection error)}}^2$$

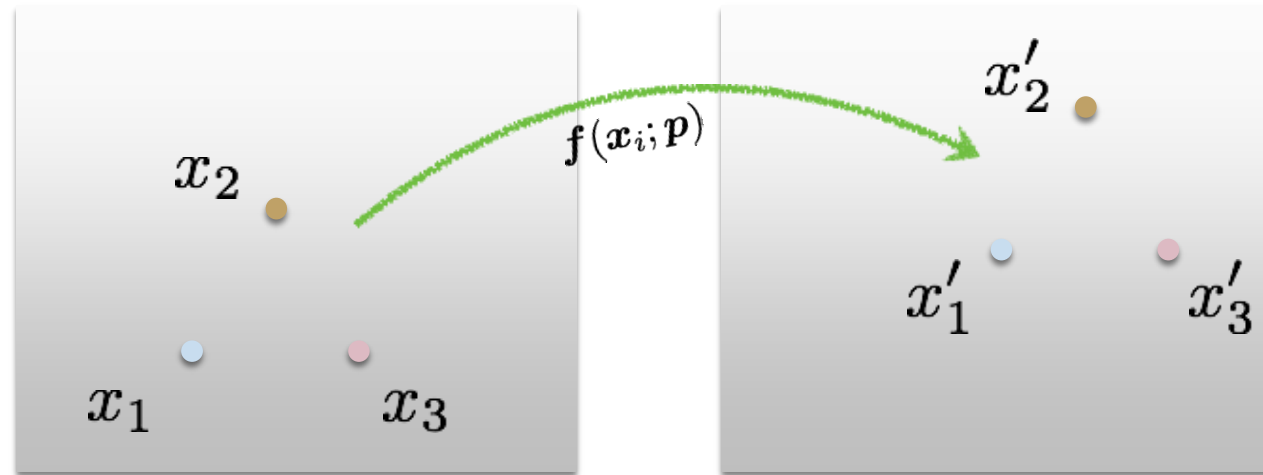


Least Squares Error

$$E_{\text{LS}} = \sum_i \|\mathbf{f}(\mathbf{x}_i; \mathbf{p}) - \mathbf{x}'_i\|^2$$

What is the free variable?

What do we want to optimize?



Find parameters that minimize squared error

$$\hat{\mathbf{p}} = \arg \min_{\mathbf{p}} \sum_i \|\mathbf{f}(\mathbf{x}_i; \mathbf{p}) - \mathbf{x}'_i\|^2$$

General form of linear least squares

(**Warning:** change of notation. \mathbf{x} is a vector of parameters!)

$$\begin{aligned} E_{\text{LLS}} &= \sum_i |\mathbf{a}_i \mathbf{x} - \mathbf{b}_i|^2 \\ &= \|\mathbf{A} \mathbf{x} - \mathbf{b}\|^2 \quad (\text{matrix form}) \end{aligned}$$

Determining unknown transformations

Affine transformation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Why can we drop
the last line?

Vectorize transformation
parameters:

$$\begin{bmatrix} x' \\ y' \\ x' \\ y' \\ \vdots \\ x' \\ y' \end{bmatrix} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \\ \vdots & & & \vdots & & \\ x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{bmatrix}$$

Stack equations from point
correspondences:

$$\underbrace{\begin{bmatrix} x' \\ y' \end{bmatrix}}_{\mathbf{b}} = \underbrace{\begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \\ p_6 \end{bmatrix}}_{\mathbf{x}}$$

Notation in system form:

\mathbf{b}

\mathbf{A}

\mathbf{x}

General form of linear least squares

(**Warning:** change of notation. \mathbf{x} is a vector of parameters!)

$$\begin{aligned} E_{\text{LLS}} &= \sum_i |\mathbf{a}_i \mathbf{x} - \mathbf{b}_i|^2 \\ &= \|\mathbf{A} \mathbf{x} - \mathbf{b}\|^2 \quad (\text{matrix form}) \end{aligned}$$

This function is quadratic.

How do you find the root of a quadratic?

Solving the linear system

Convert the system to a linear least-squares problem:

$$E_{\text{LLS}} = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$$

Expand the error:

$$E_{\text{LLS}} = \mathbf{x}^\top (\mathbf{A}^\top \mathbf{A}) \mathbf{x} - 2\mathbf{x}^\top (\mathbf{A}^\top \mathbf{b}) + \|\mathbf{b}\|^2$$

Minimize the error:

Set derivative to 0 $(\mathbf{A}^\top \mathbf{A})\mathbf{x} = \mathbf{A}^\top \mathbf{b}$

Solve for \mathbf{x} $\mathbf{x} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}$ ←

In Matlab:

$$\mathbf{x} = \mathbf{A} \setminus \mathbf{b}$$

Note: You almost never want to compute the inverse of a matrix.

Linear least squares estimation only works when the transform function is ?

Linear least squares estimation only works when the transform function is **linear!** (duh)

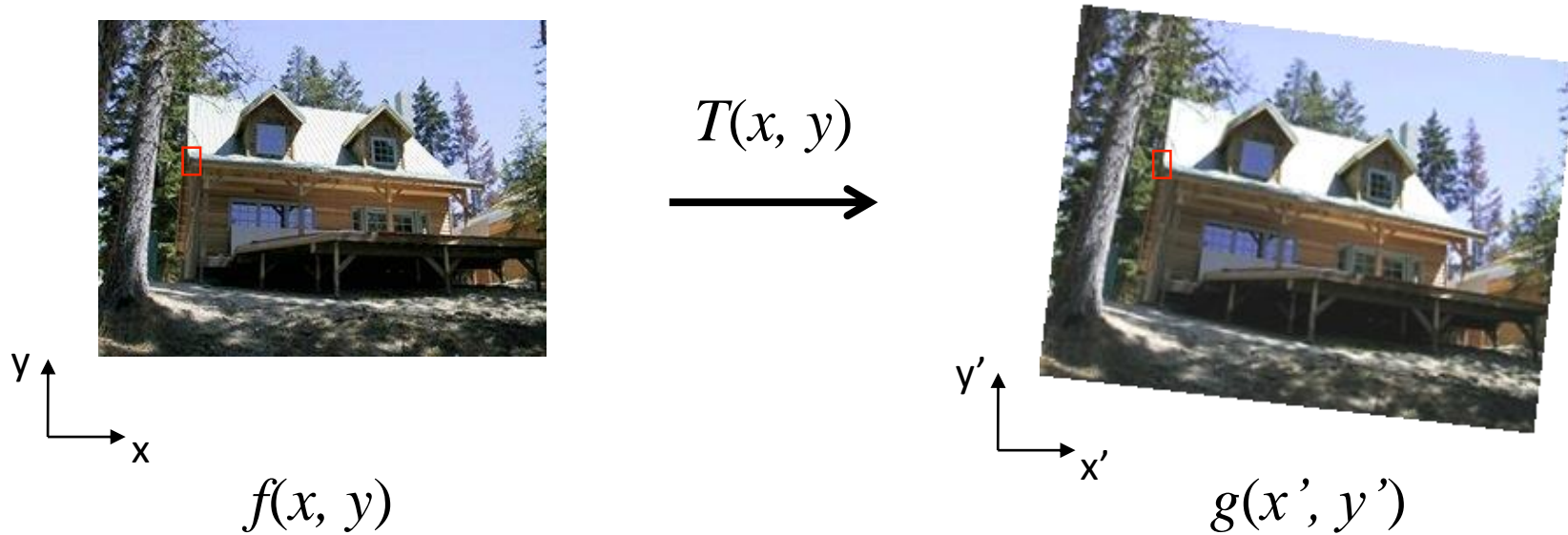
Also doesn't deal well with outliers

Determining unknown image warps

Determining unknown image warps

Suppose we have two images.

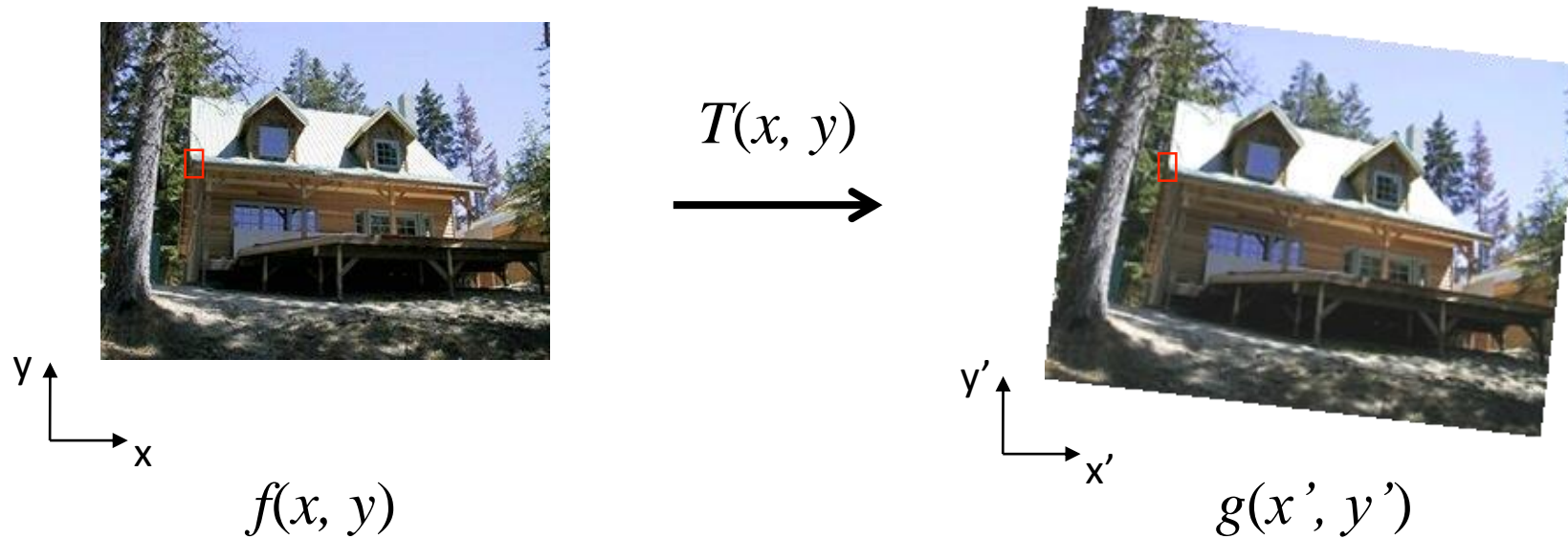
- How do we compute the transform that takes one to the other?



Forward warping

Suppose we have two images.

- How do we compute the transform that takes one to the other?

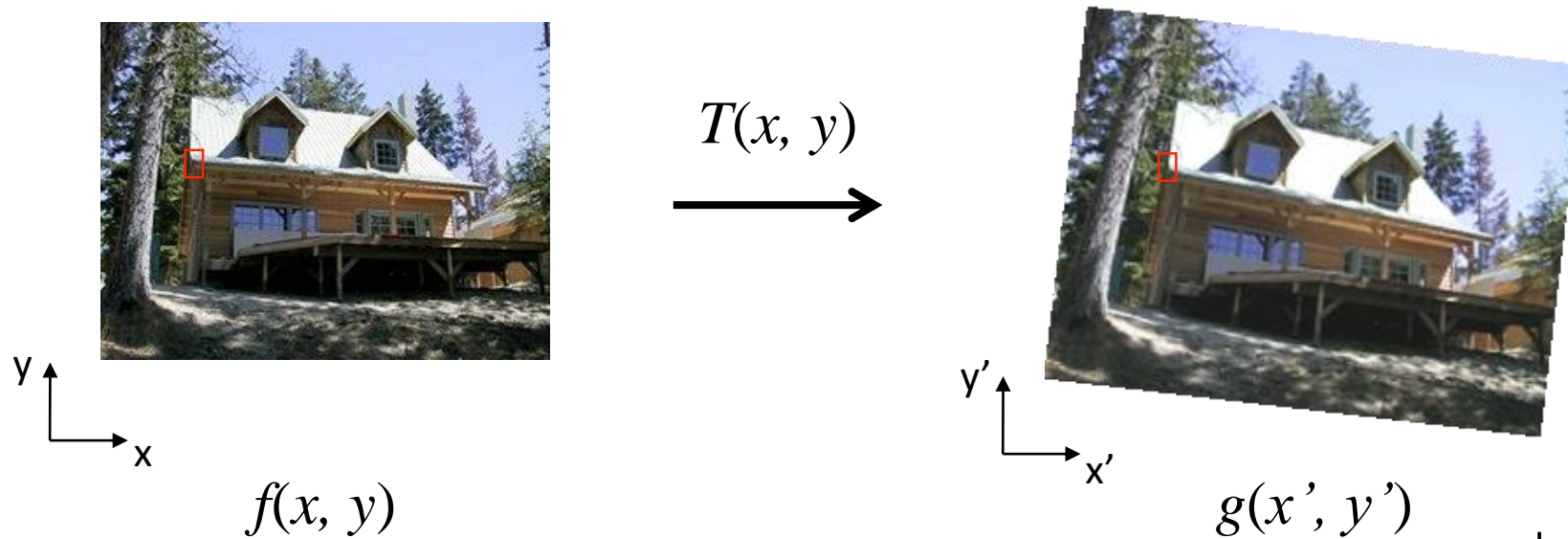


1. Form enough pixel-to-pixel correspondences between two images ← later lecture
2. Solve for linear transform parameters as before
3. Send intensities $f(x, y)$ in first image to their corresponding location in the second image

Forward warping

Suppose we have two images.

- How do we compute the transform that takes one to the other?



what is the problem
with this?

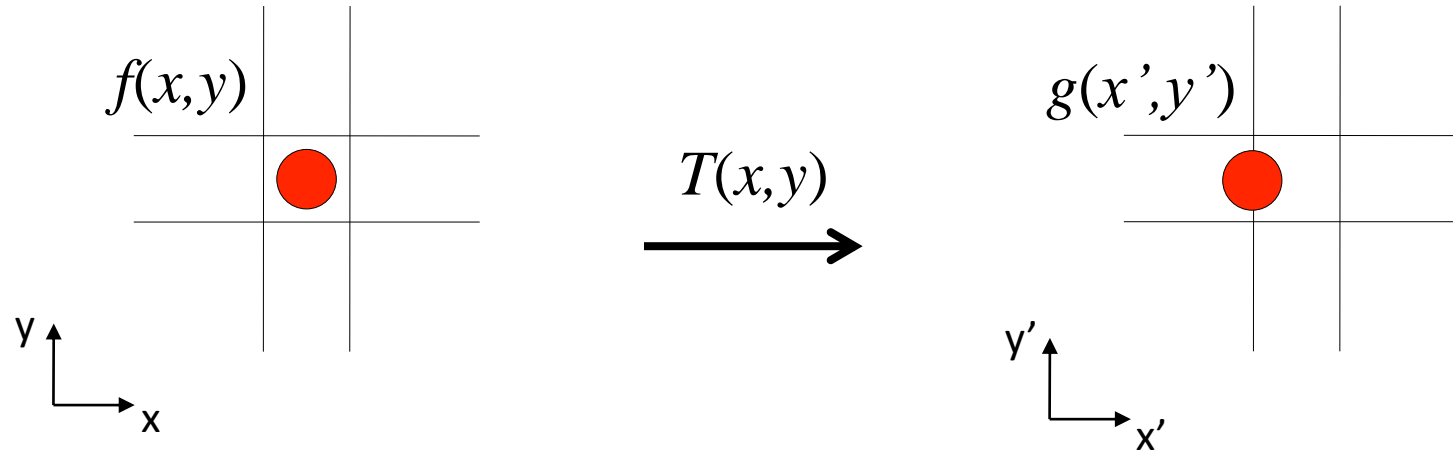
1. Form enough pixel-to-pixel correspondences between two images
2. Solve for linear transform parameters as before
3. Send intensities $f(x, y)$ in first image to their corresponding location in the second image



Forward warping

Pixels may end up between two points

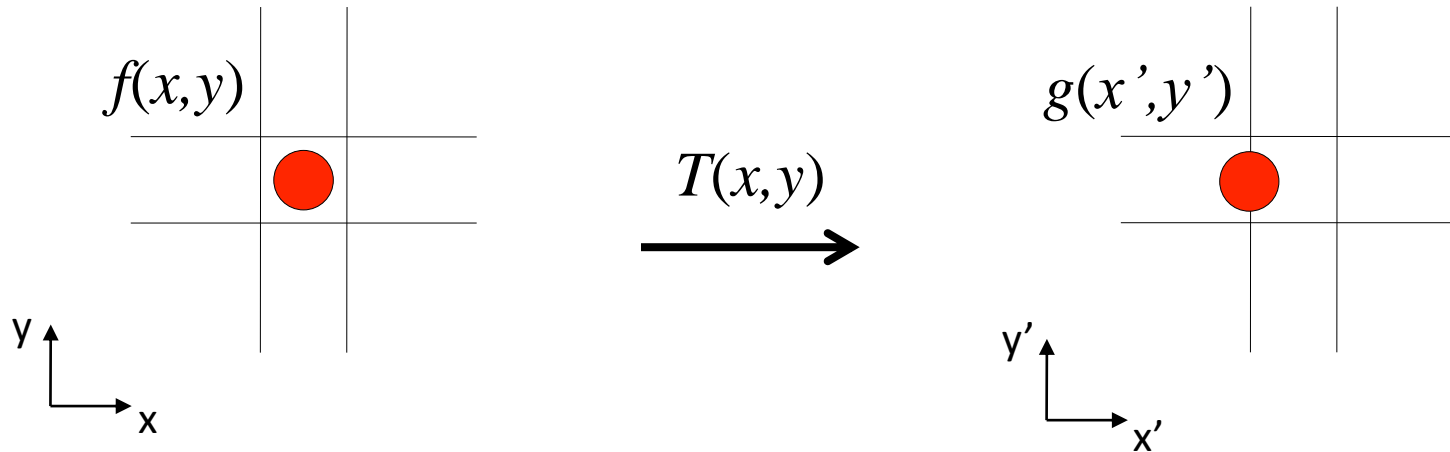
- How do we determine the intensity of each point?



Forward warping

Pixels may end up between two points

- How do we determine the intensity of each point?
- ✓ We distribute color among neighboring pixels (x',y') (“splatting”)

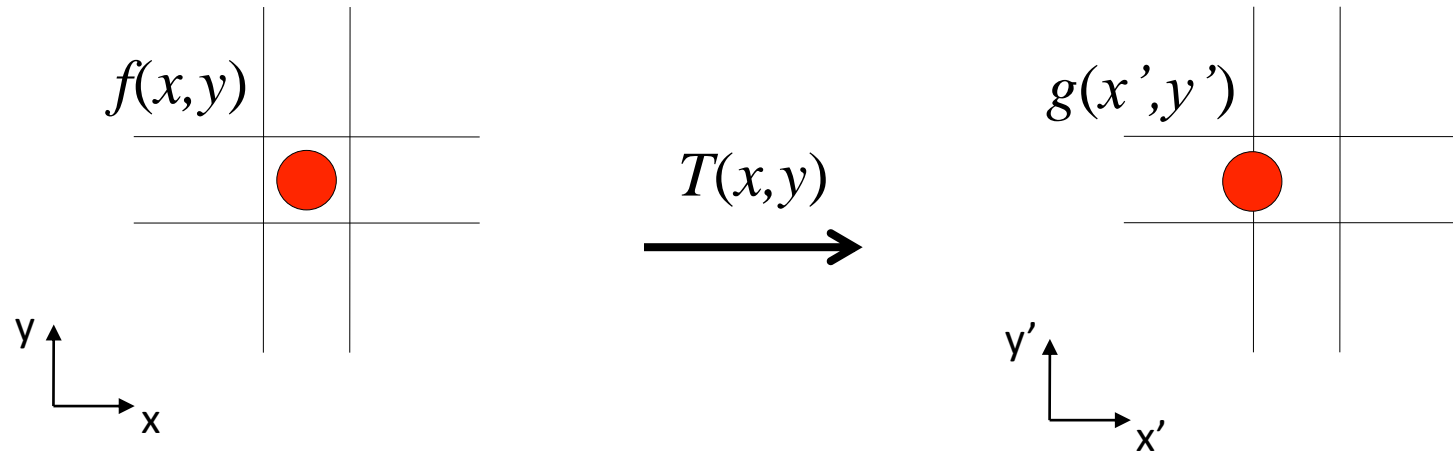


- What if a pixel (x',y') receives intensity from more than one pixels (x,y) ?

Forward warping

Pixels may end up between two points

- How do we determine the intensity of each point?
- ✓ We distribute color among neighboring pixels (x',y') (“splatting”)

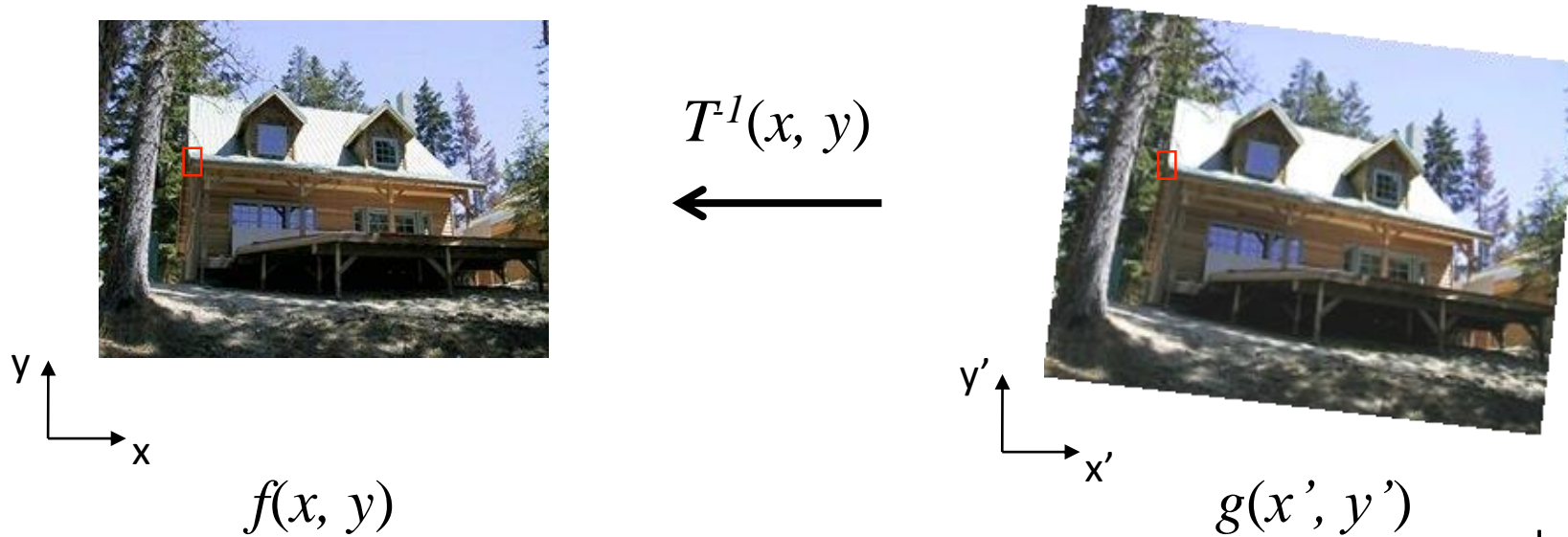


- What if a pixel (x',y') receives intensity from more than one pixels (x,y) ?
- ✓ We average their intensity contributions.

Inverse warping

Suppose we have two images.

- How do we compute the transform that takes one to the other?



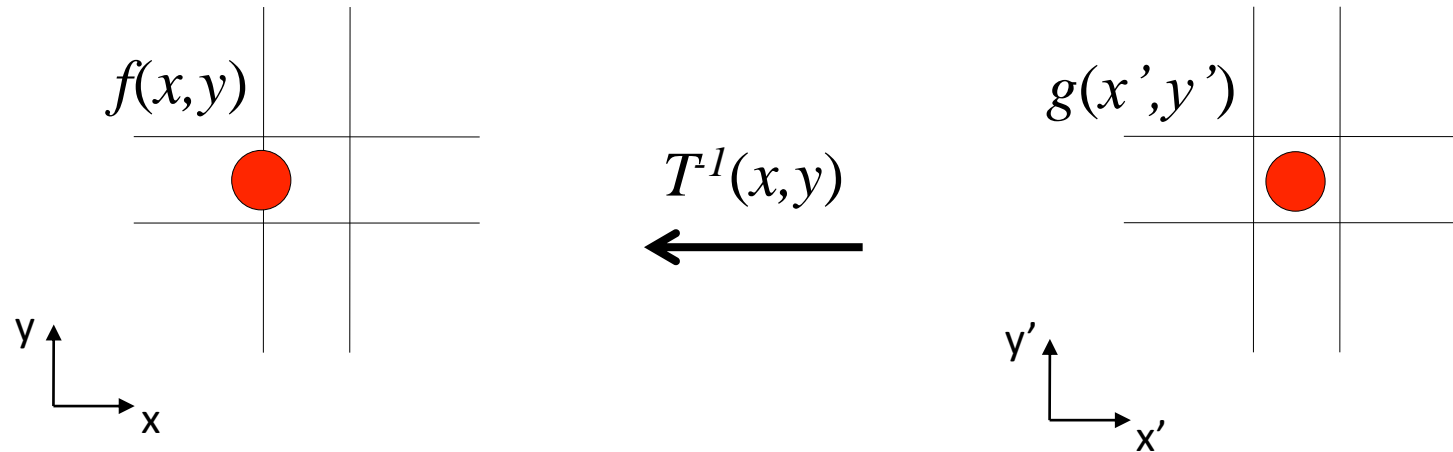
what is the problem
with this?

1. Form enough pixel-to-pixel correspondences between two images
2. Solve for linear transform parameters as before, then compute its inverse
3. Get intensities $g(x', y')$ in the second image from point $(x, y) = T^{-1}(x', y')$ in first image

Inverse warping

Pixel may come from between two points

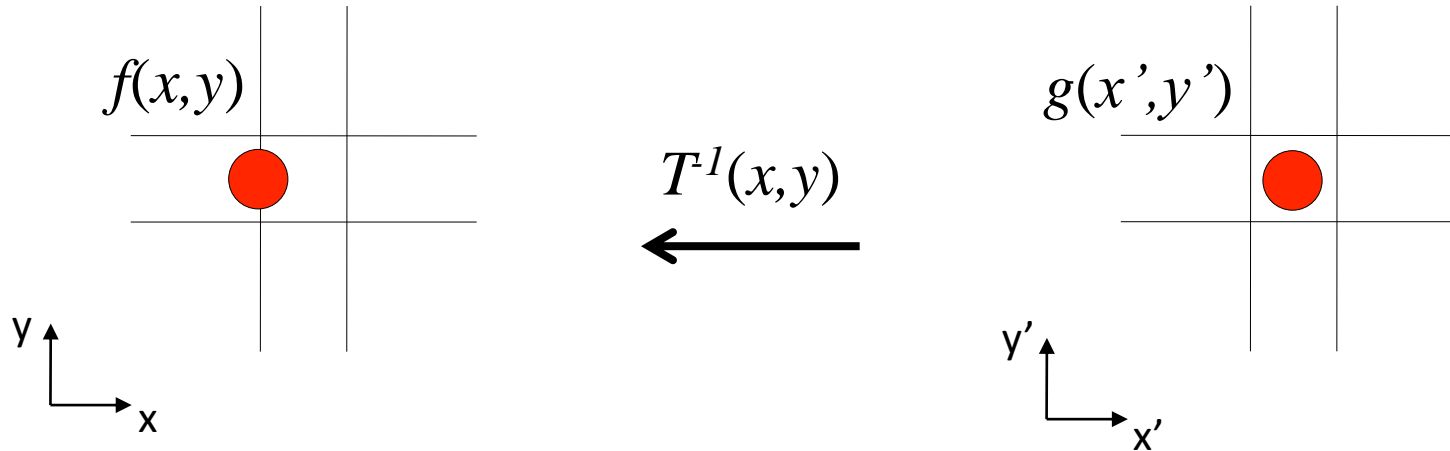
- How do we determine its intensity?



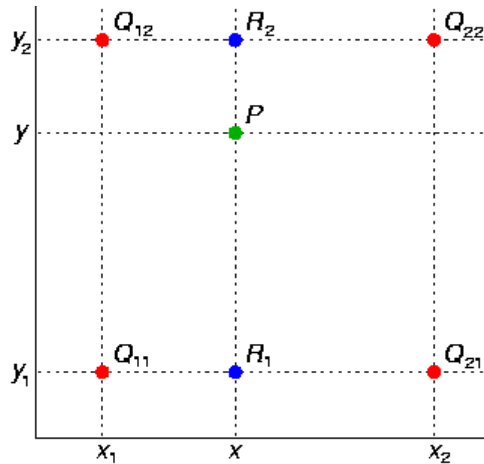
Inverse warping

Pixel may come from between two points

- How do we determine its intensity?
- ✓ Use interpolation

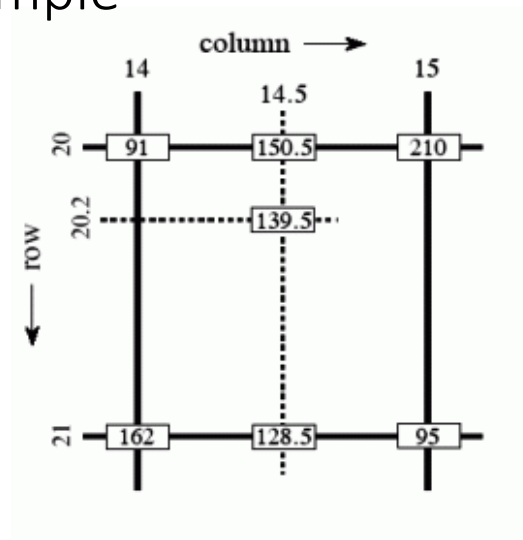


Bilinear interpolation



1. Interpolate to find R_2
2. Interpolate to find R_1
3. Interpolate to find P

Grayscale example



In matrix form (with adjusted coordinates)

$$f(x, y) \approx \begin{bmatrix} 1-x & x \end{bmatrix} \begin{bmatrix} f(0,0) & f(0,1) \\ f(1,0) & f(1,1) \end{bmatrix} \begin{bmatrix} 1-y \\ y \end{bmatrix}.$$

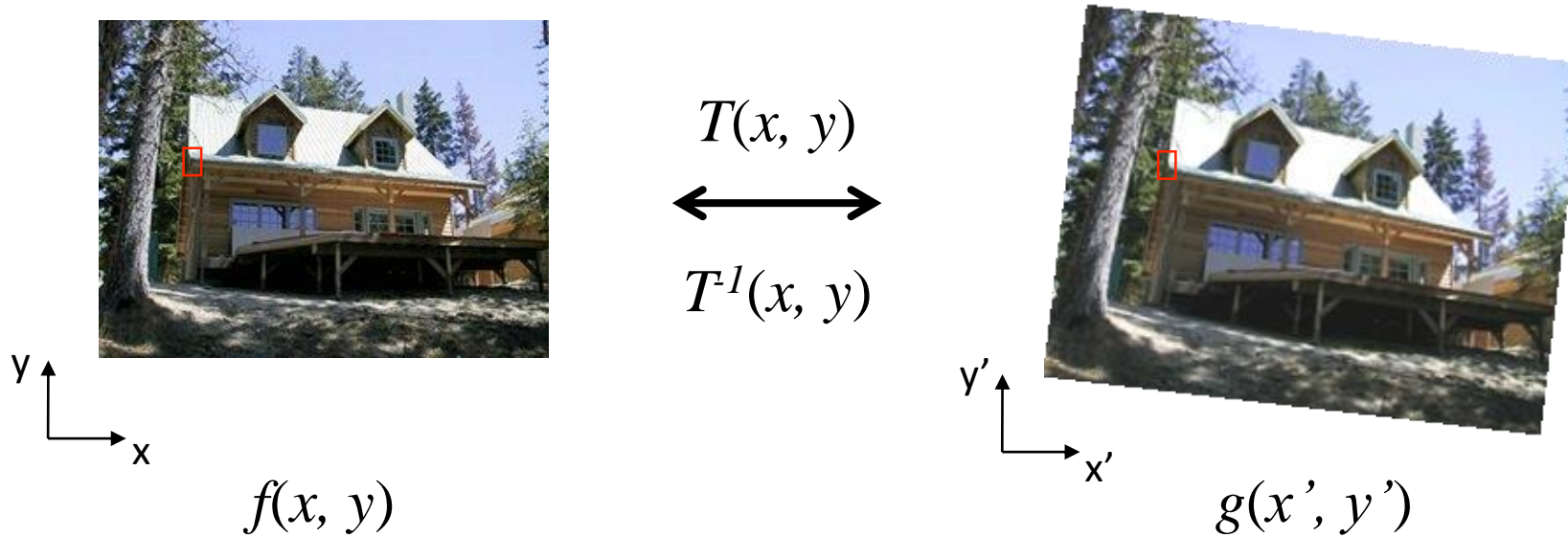
In Matlab:

call `interp2`

Forward vs inverse warping

Suppose we have two images.

- How do we compute the transform that takes one to the other?

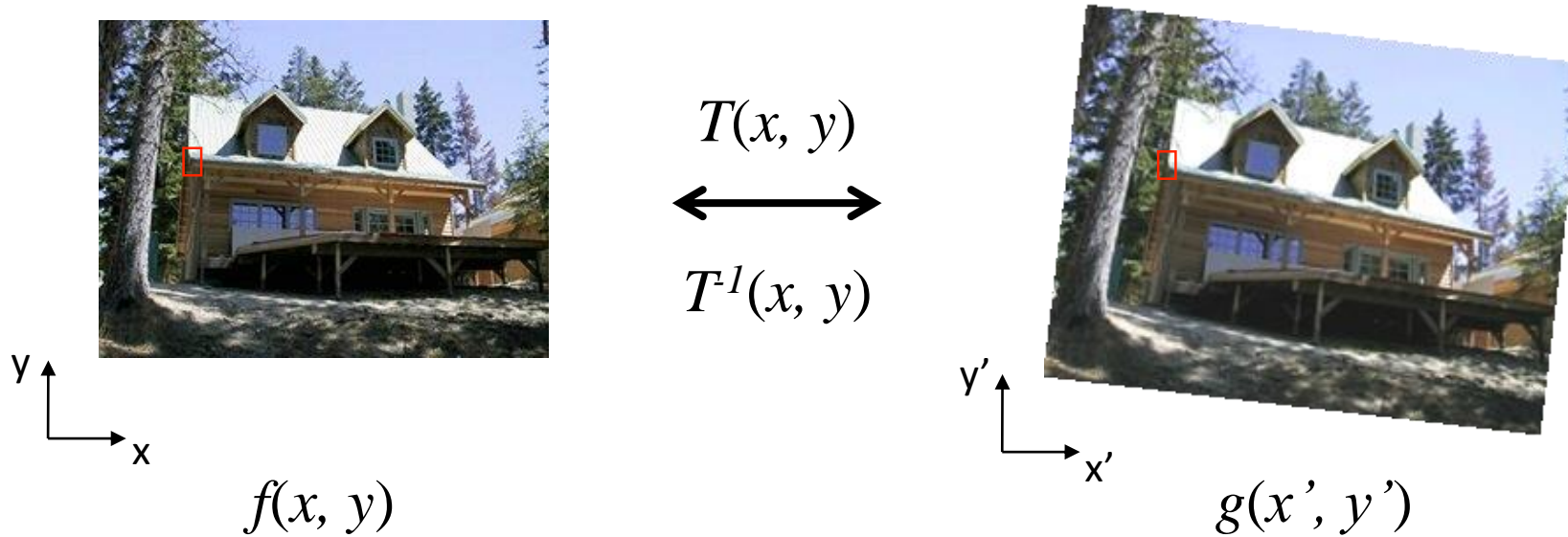


Pros and cons of each?

Forward vs inverse warping

Suppose we have two images.

- How do we compute the transform that takes one to the other?



- Inverse warping eliminates holes in target image
- Forward warping does not require existence of inverse transform

References

Basic reading:

- Szeliski textbook, Section 3.6.

Additional reading:

- Hartley and Zisserman, “Multiple View Geometry in Computer Vision,” Cambridge University Press 2004.
a comprehensive treatment of all aspects of projective geometry relating to computer vision, and also a very useful reference for the second part of the class.
- Richter-Gebert, “Perspectives on projective geometry,” Springer 2011.
a beautiful, thorough, and very accessible mathematics textbook on projective geometry (available online for free from CMU’s library).