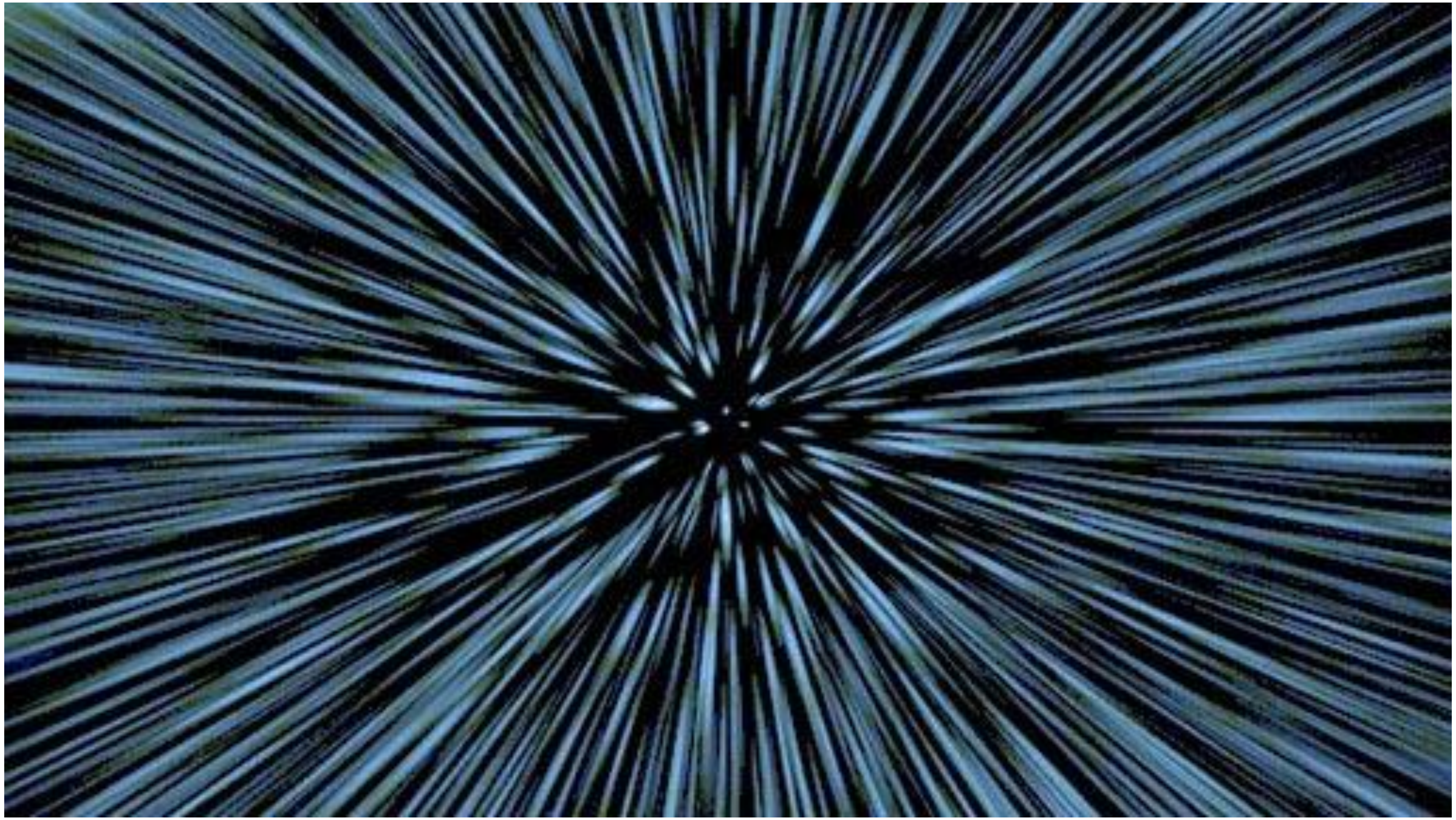
2D transformations (a.k.a. warping)



16-385 Computer Vision
Spring 2019, Lecture 7

http://www.cs.cmu.edu/~16385/

Course announcements

- Homework 2 is posted on the course website.
  - It is due on February 27th at 23:59 pm.
  - Start early because it is much larger and more difficult than homework 1.
  - Note the updated deadline – homeworks are now back in sync with lectures, but homework 4 will be turned into a single-week homework.

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

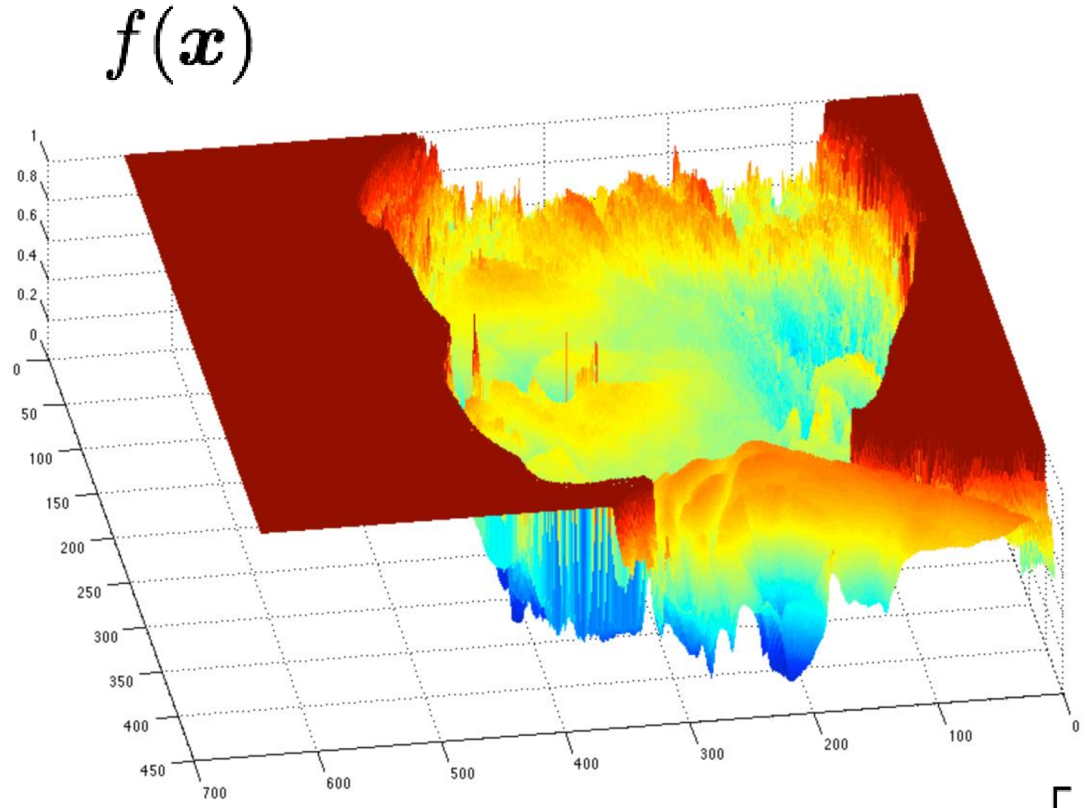
Most of these slides were adapted from:

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

Reminder: image transformations

What is an image?

grayscale image



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

A (grayscale) image is a 2D function.

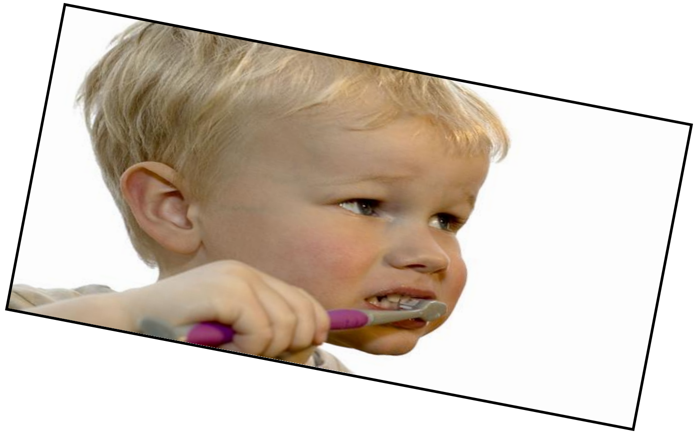
What is the range of the image function f?

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

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

G

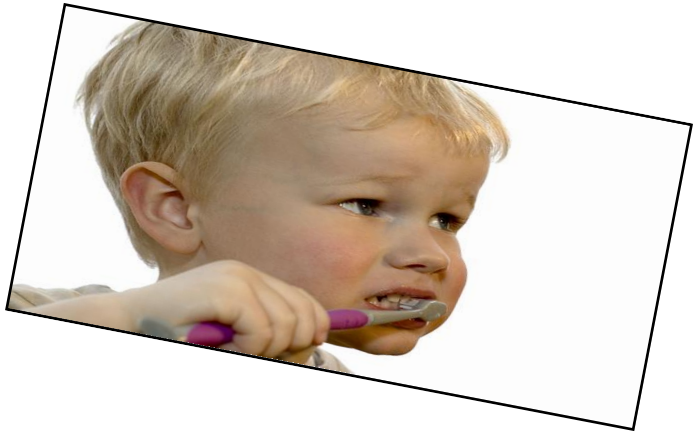
changes *range* of image function

F

Warping

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

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:

$$\{\boldsymbol{x}_i, \boldsymbol{x}'_i\}$$

point in one image

point in the other image

and a transformation:

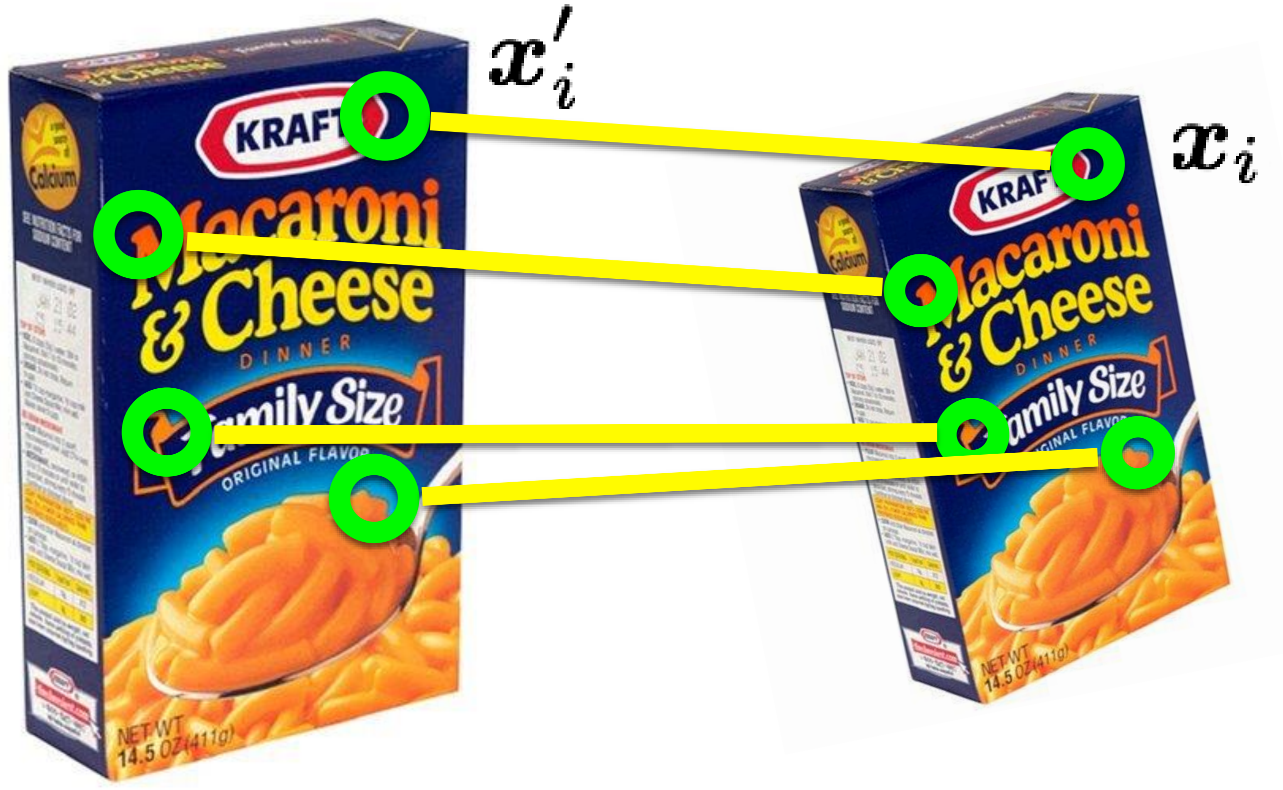
$$\boldsymbol{x}' = \boldsymbol{f}(\boldsymbol{x}; \boldsymbol{p})$$

transformation function

parameters

find the best estimate of the parameters

$$\boldsymbol{p}$$



What kind of transformation functions $\boldsymbol{f}$ are there?

2D transformations

2D transformations

translation

rotation

aspect

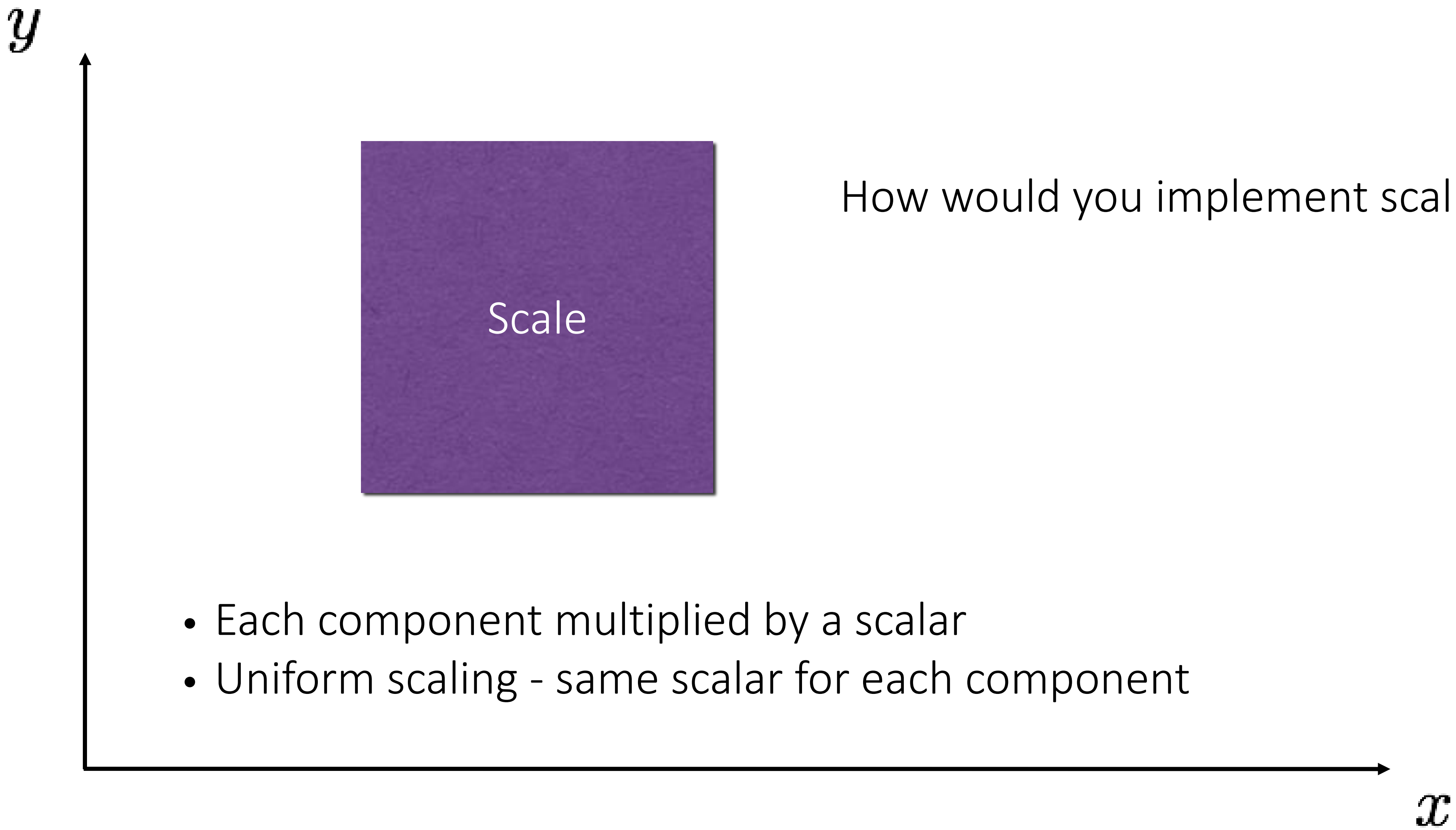
affine

perspective

cylindrical

2D planar transformations

2D planar transformations



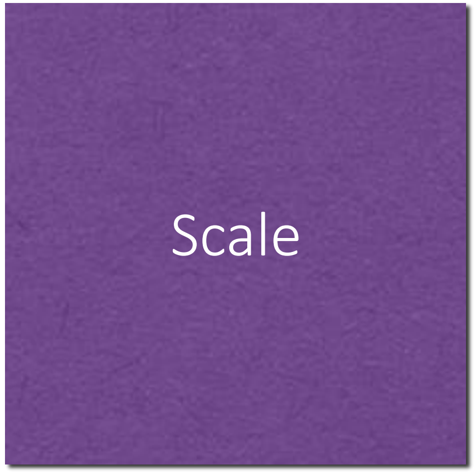
How would you implement scaling?

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

2D planar transformations

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

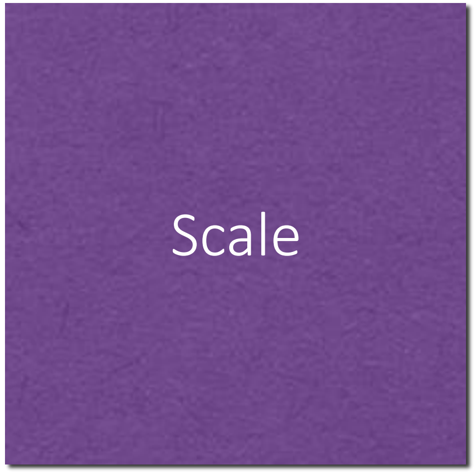
What's the effect of using different scale factors?



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

2D planar transformations

y



$$x' = ax$$

$$y' = by$$

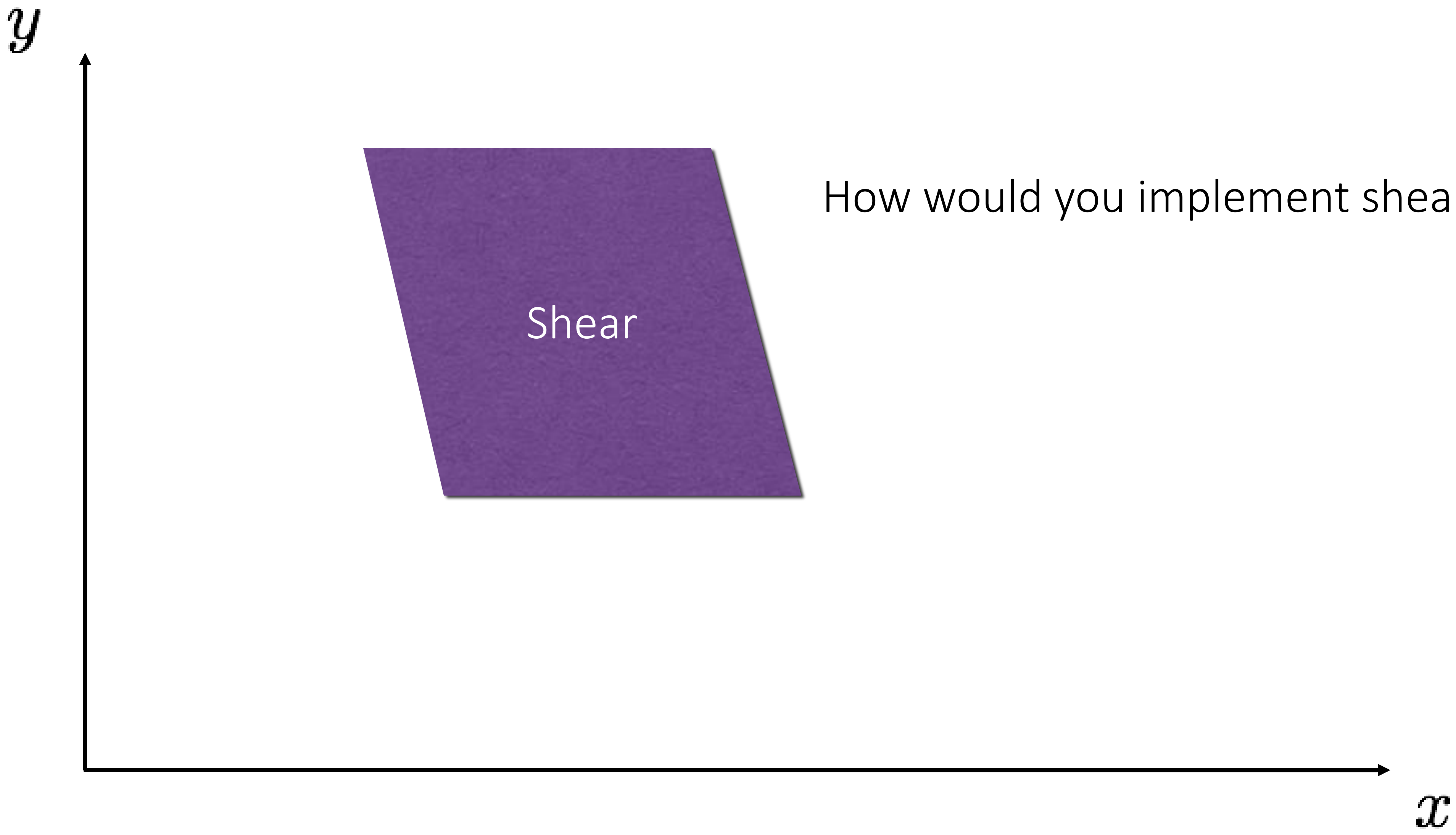
matrix representation of scaling:

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

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

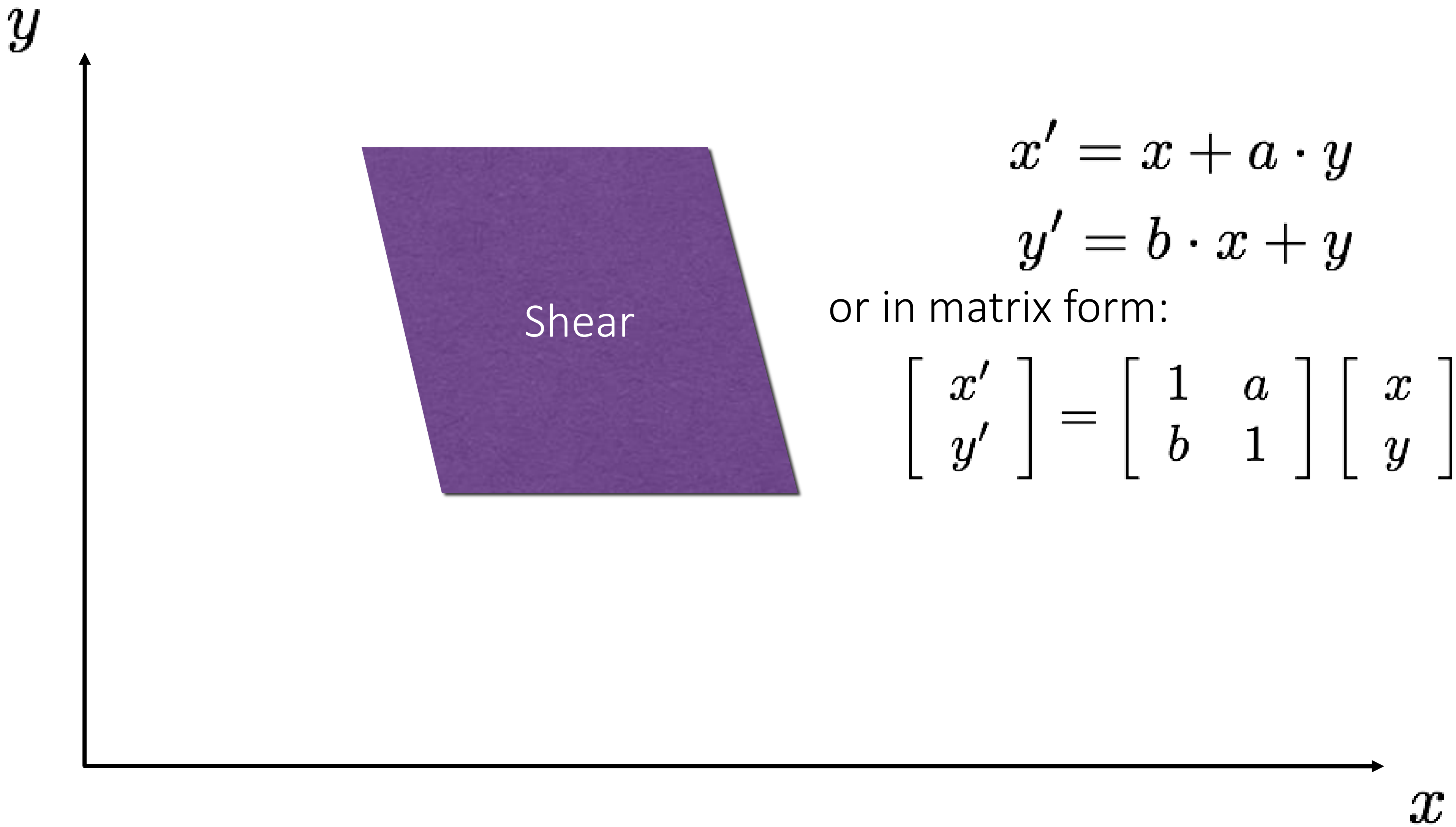
x

2D planar transformations



How would you implement shearing?

2D planar transformations



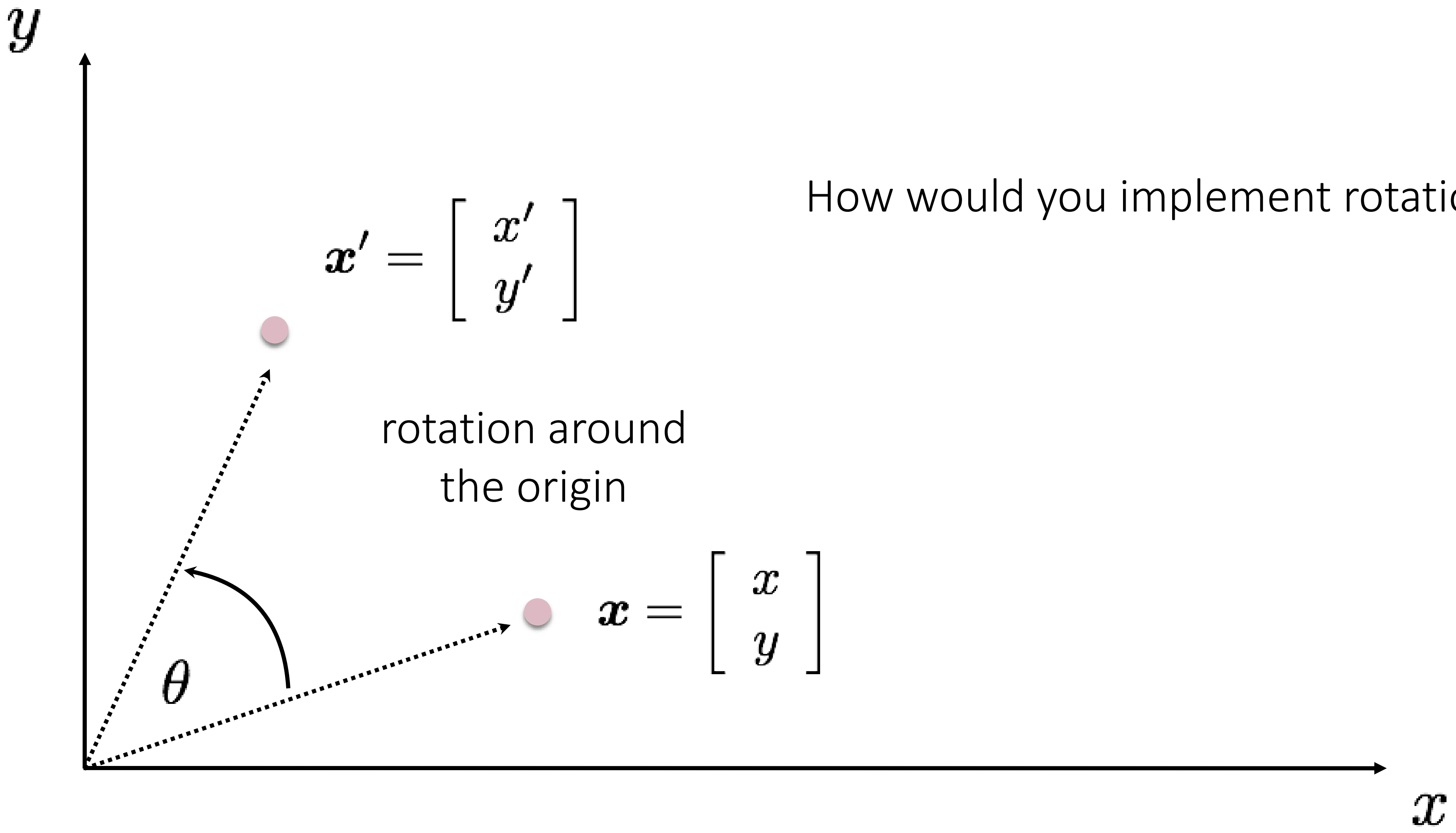
$$x' = x + a \cdot y$$

$$y' = b \cdot x + y$$

or in matrix form:

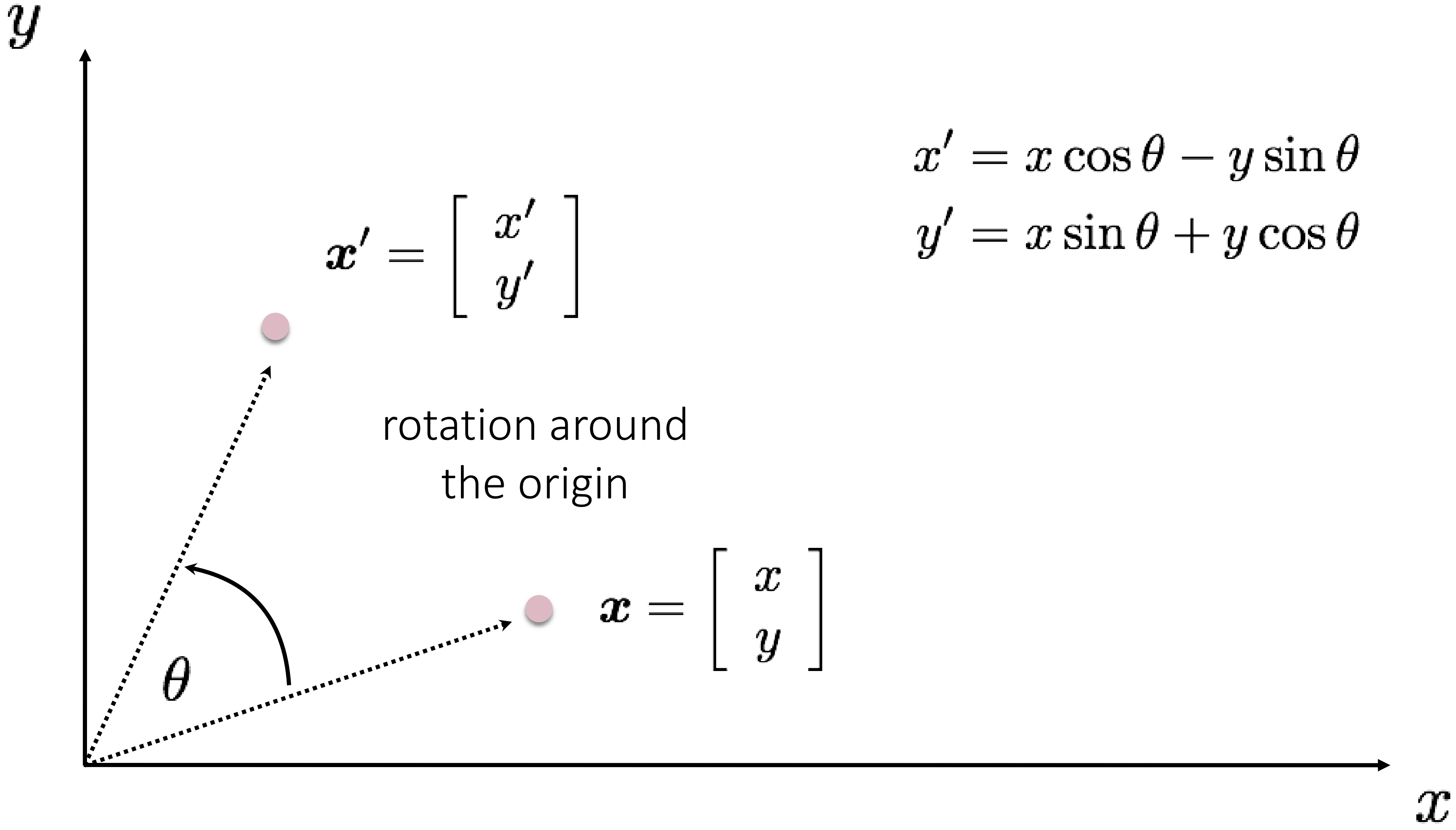
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & a \\ b & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D planar transformations



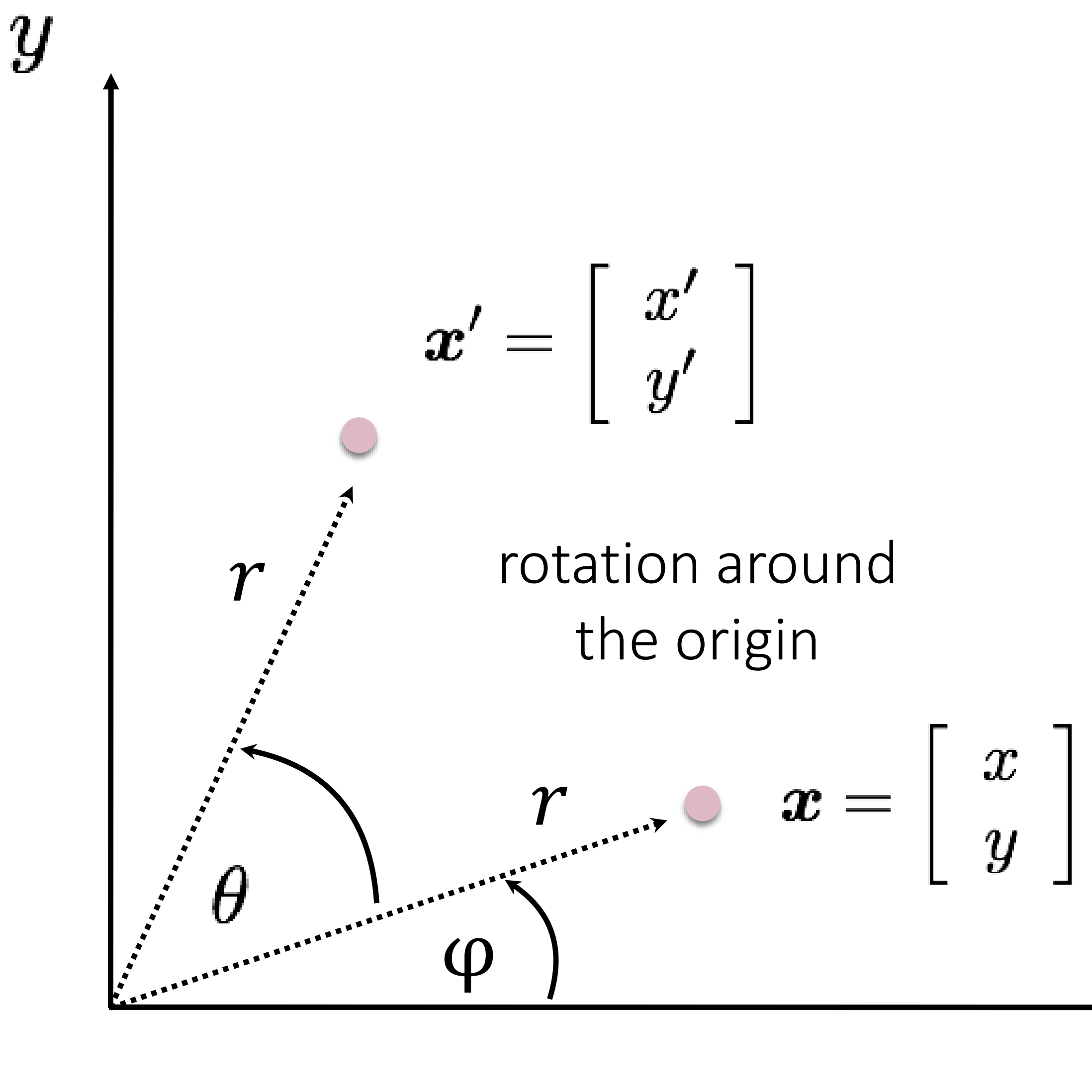
How would you implement rotation?

2D planar transformations



$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

2D planar transformations



Polar coordinates…

x = r cos (ϕ)

y = r sin (ϕ)

x’ = r cos (ϕ + θ)

y’ = r sin (ϕ + θ)

Trigonometric Identity…

x’ = r cos(ϕ) cos(θ) – r sin(ϕ) sin(θ)

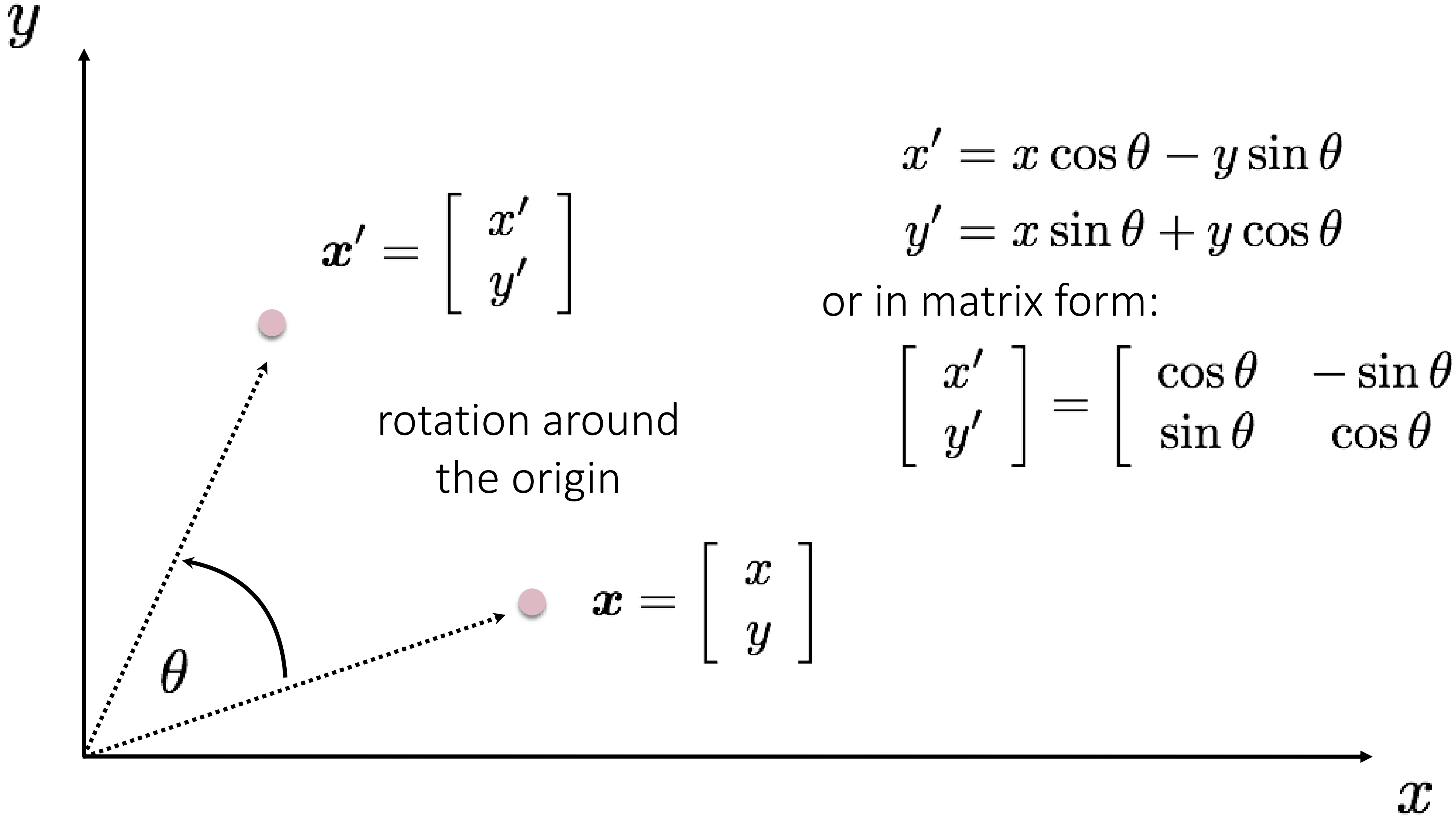
y’ = r sin(ϕ) cos(θ) + r cos(ϕ) sin(θ)

Substitute…

x’ = x cos(θ) - y sin(θ)

y’ = x sin(θ) + y cos(θ)

2D planar transformations



$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

or in matrix form:

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

2D planar and linear transformations

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

$$\downarrow$$

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

parameters $\boldsymbol{p}$

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

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

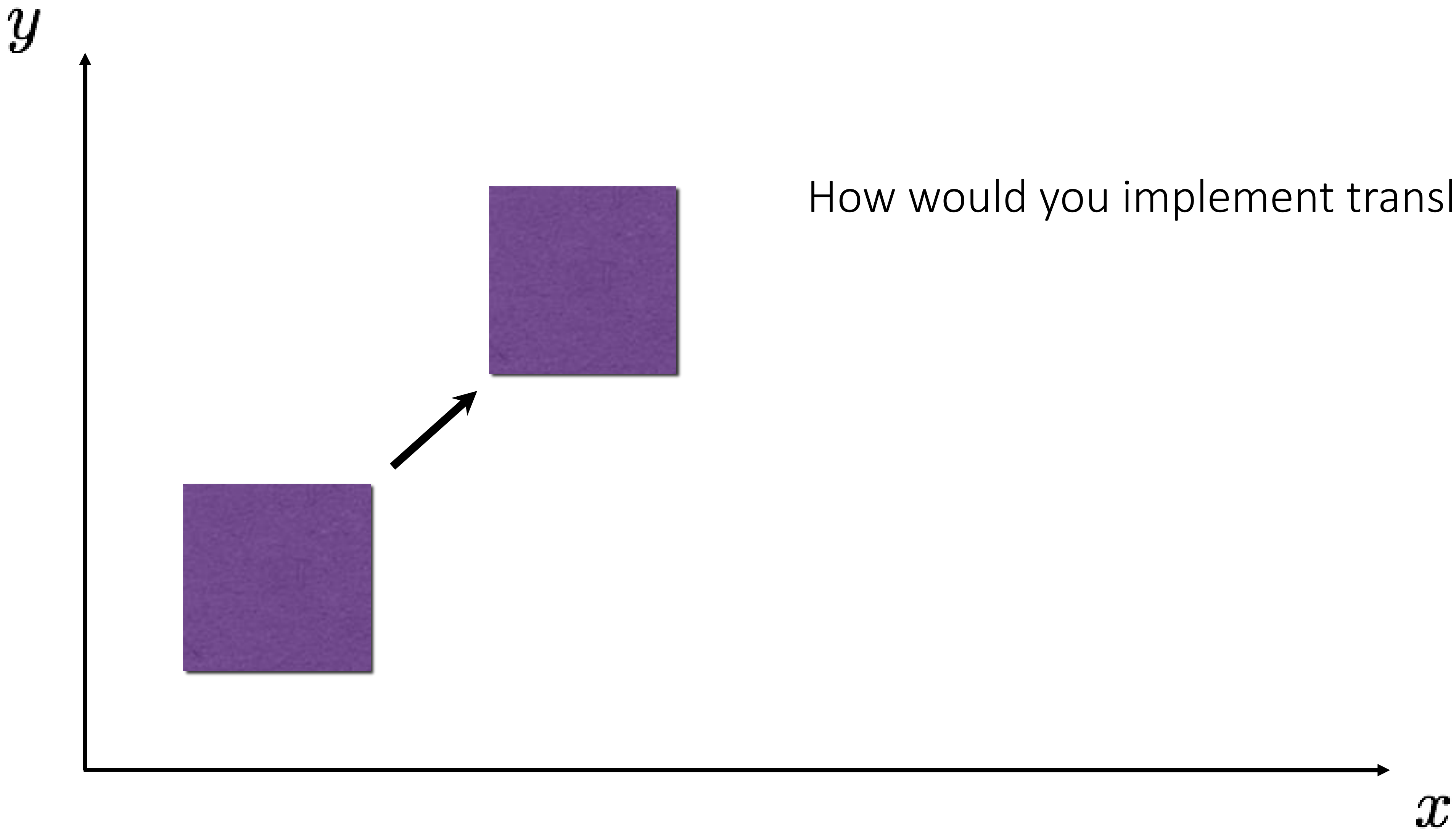
Shear

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

Identity

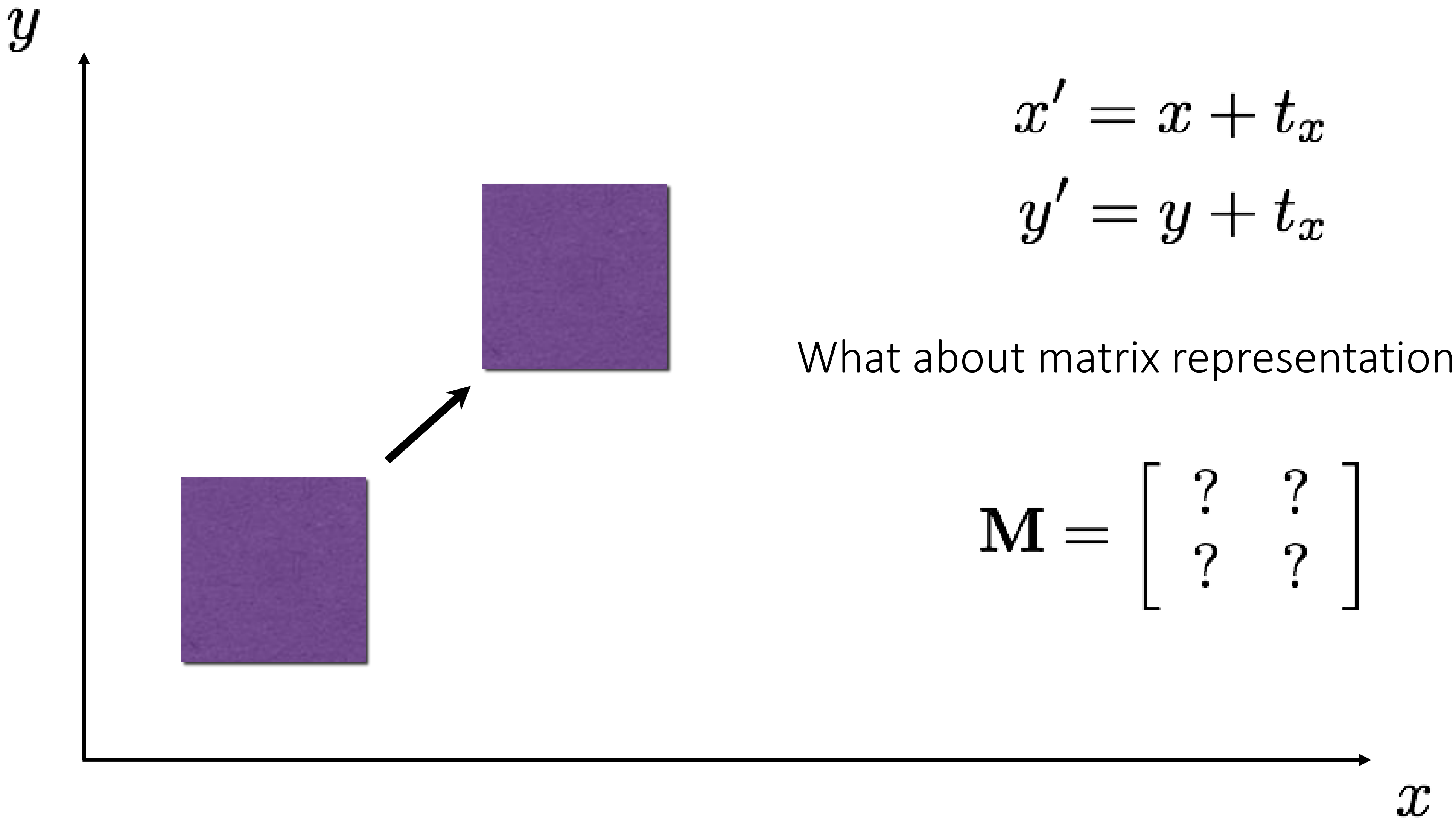
$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

2D translation



How would you implement translation?

2D translation



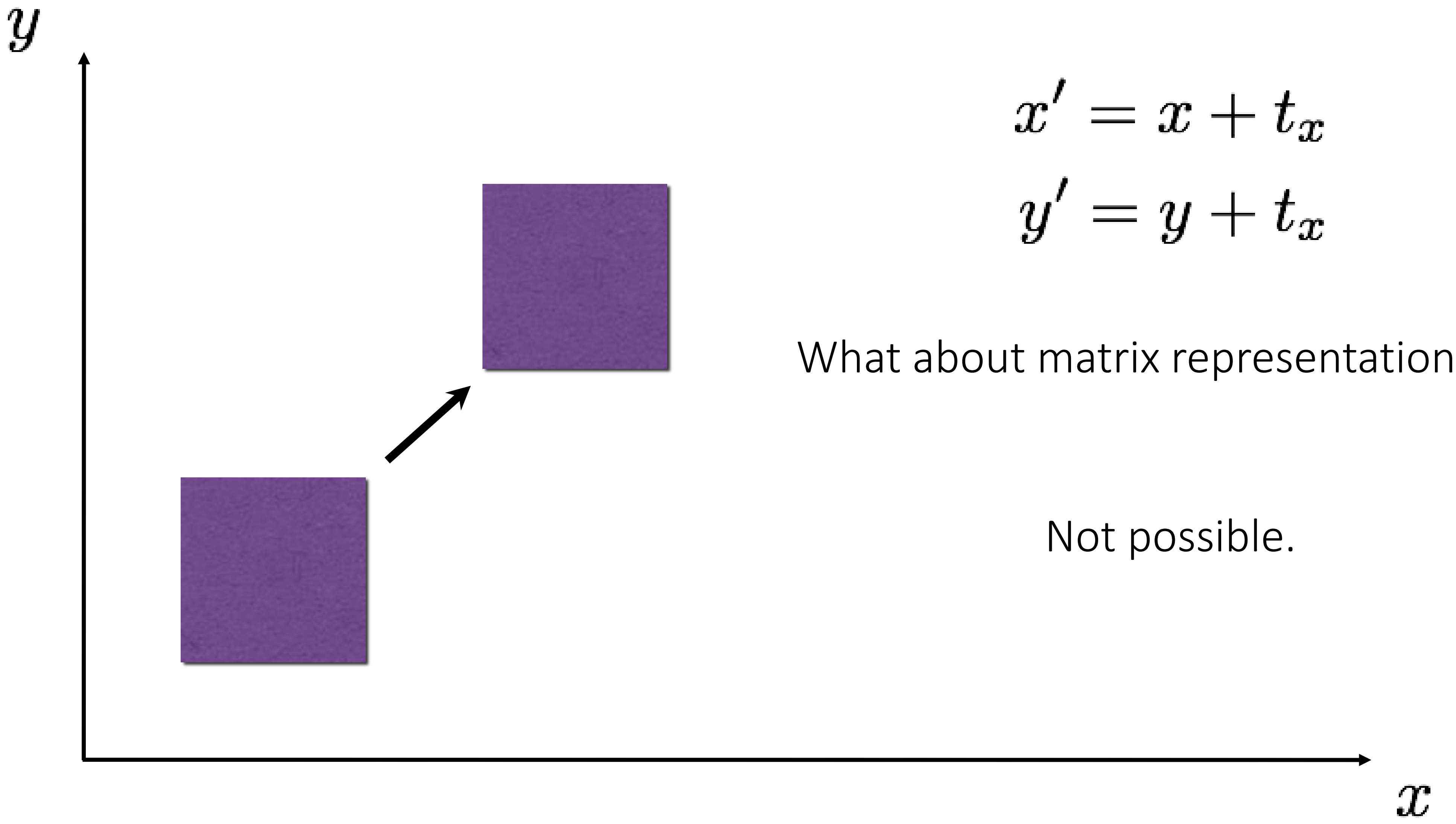
$$x' = x + t_x$$

$$y' = y + t_x$$

What about matrix representation?

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

2D translation



$$x' = x + t_x$$
$$y' = y + t_x$$

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

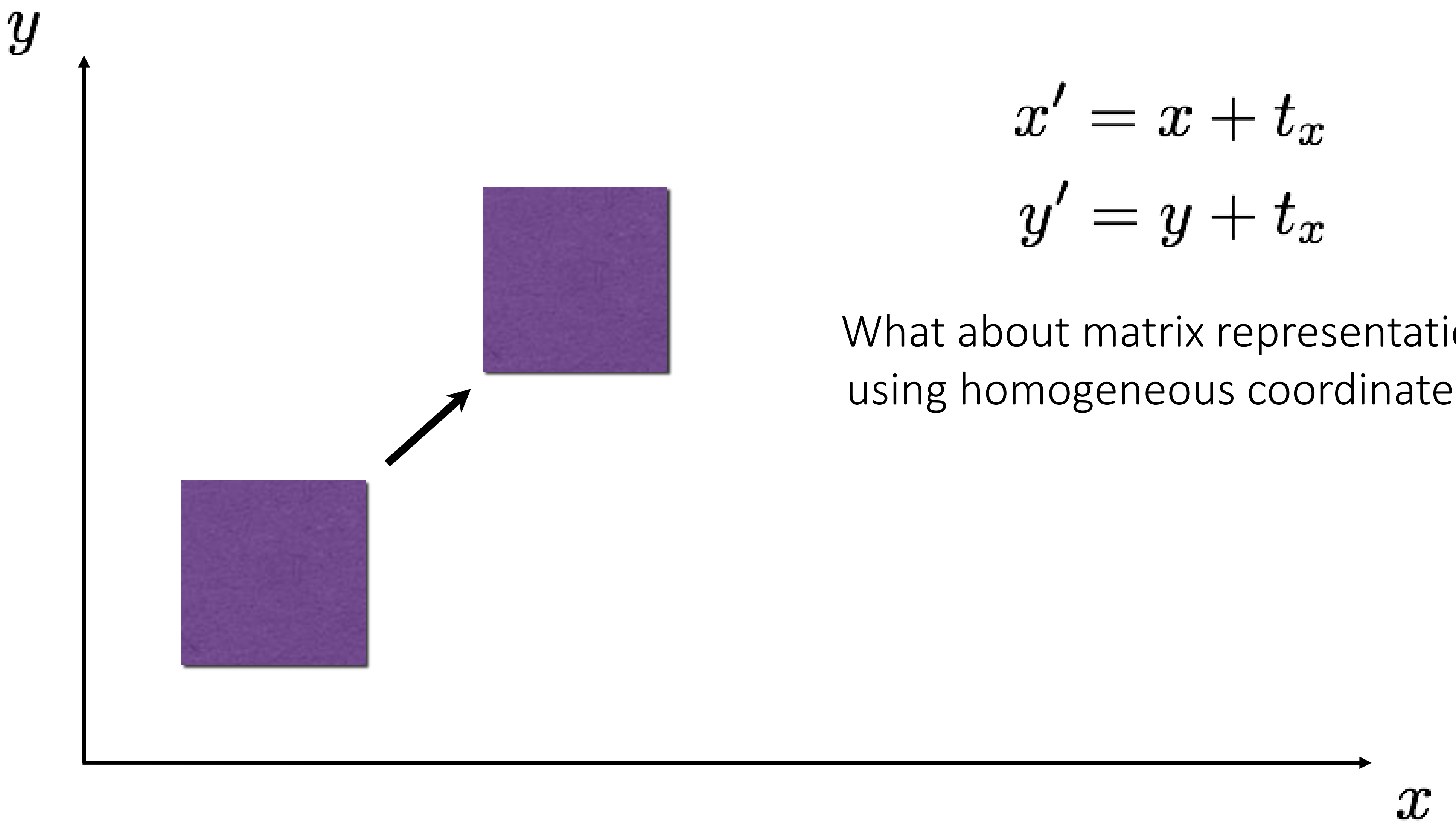
Homogeneous coordinates

heterogeneous coordinates | homogeneous coordinates

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

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

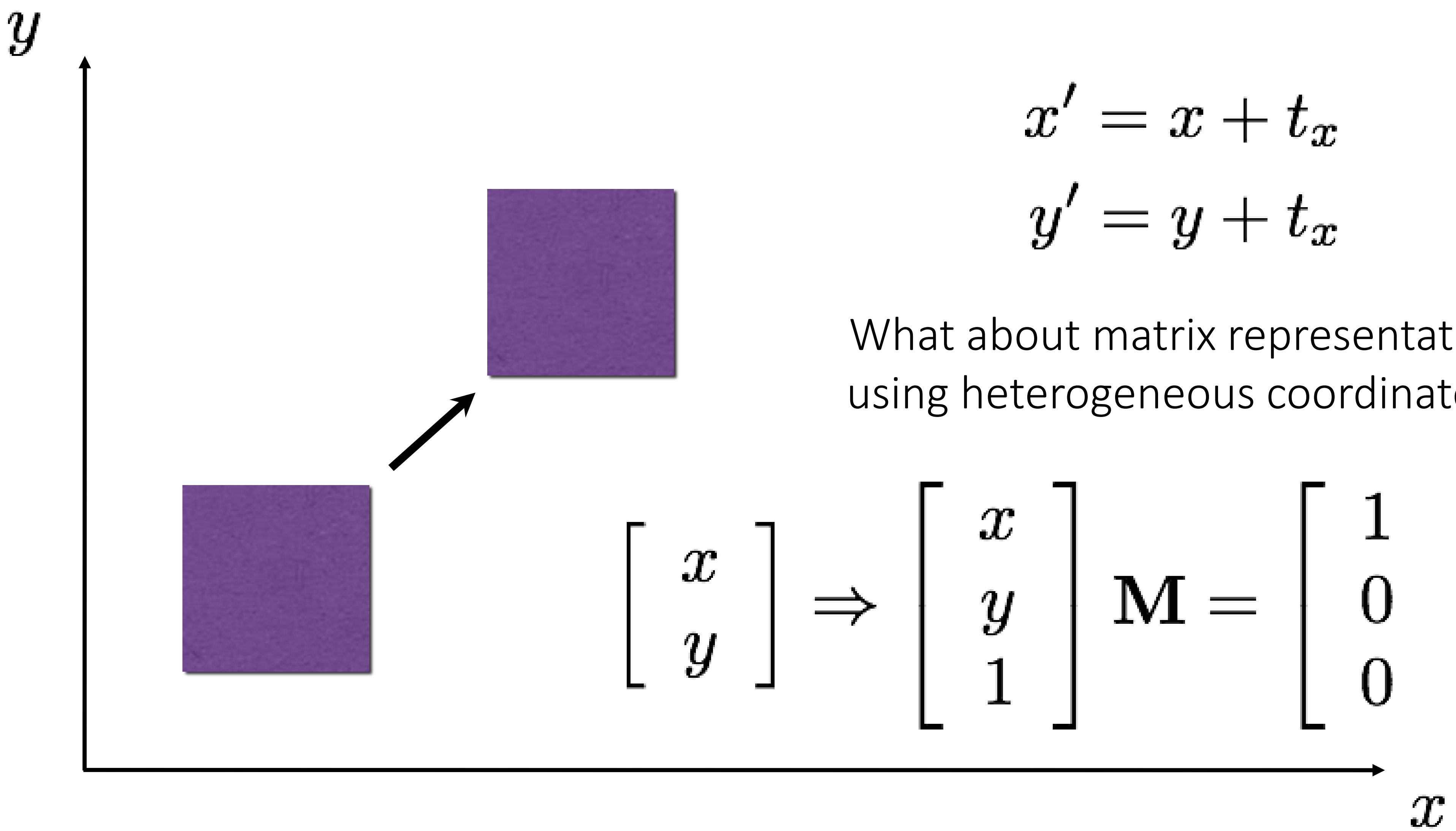
2D translation



$$x' = x + t_x$$
$$y' = y + t_x$$

What about matrix representation using homogeneous coordinates?

2D translation



$$x' = x + t_x$$

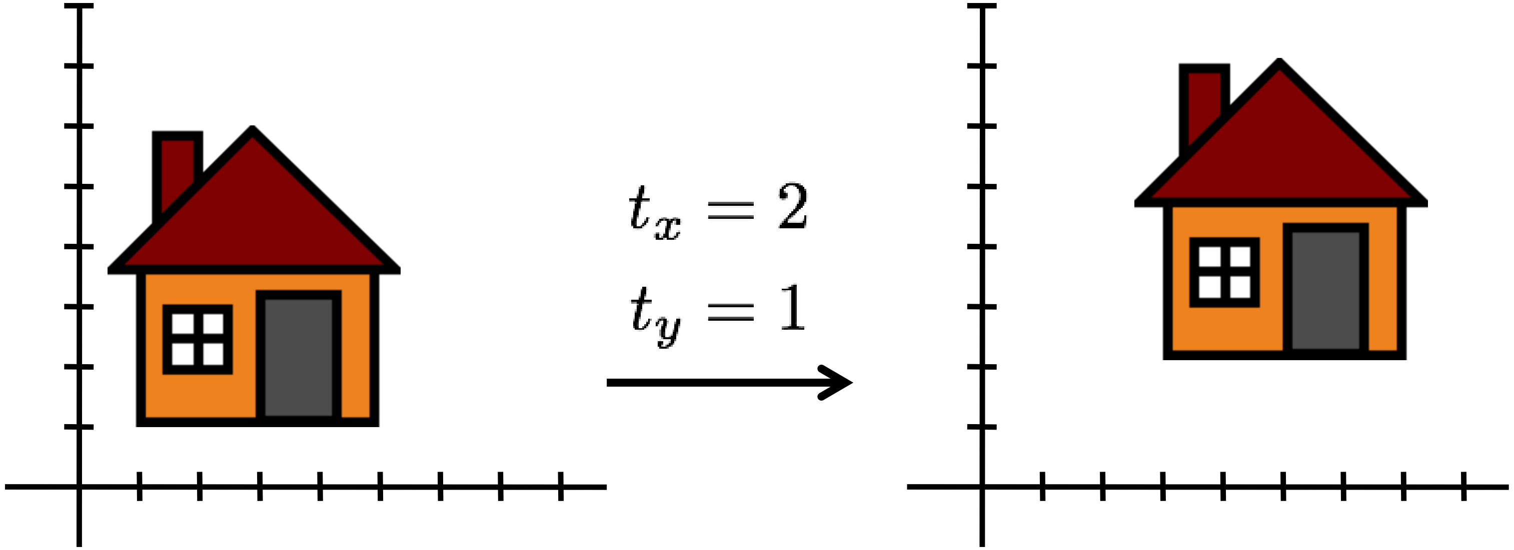
$$y' = y + t_x$$

What about matrix representation
using heterogeneous coordinates?

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

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 → homogeneous

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

- homogeneous → 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

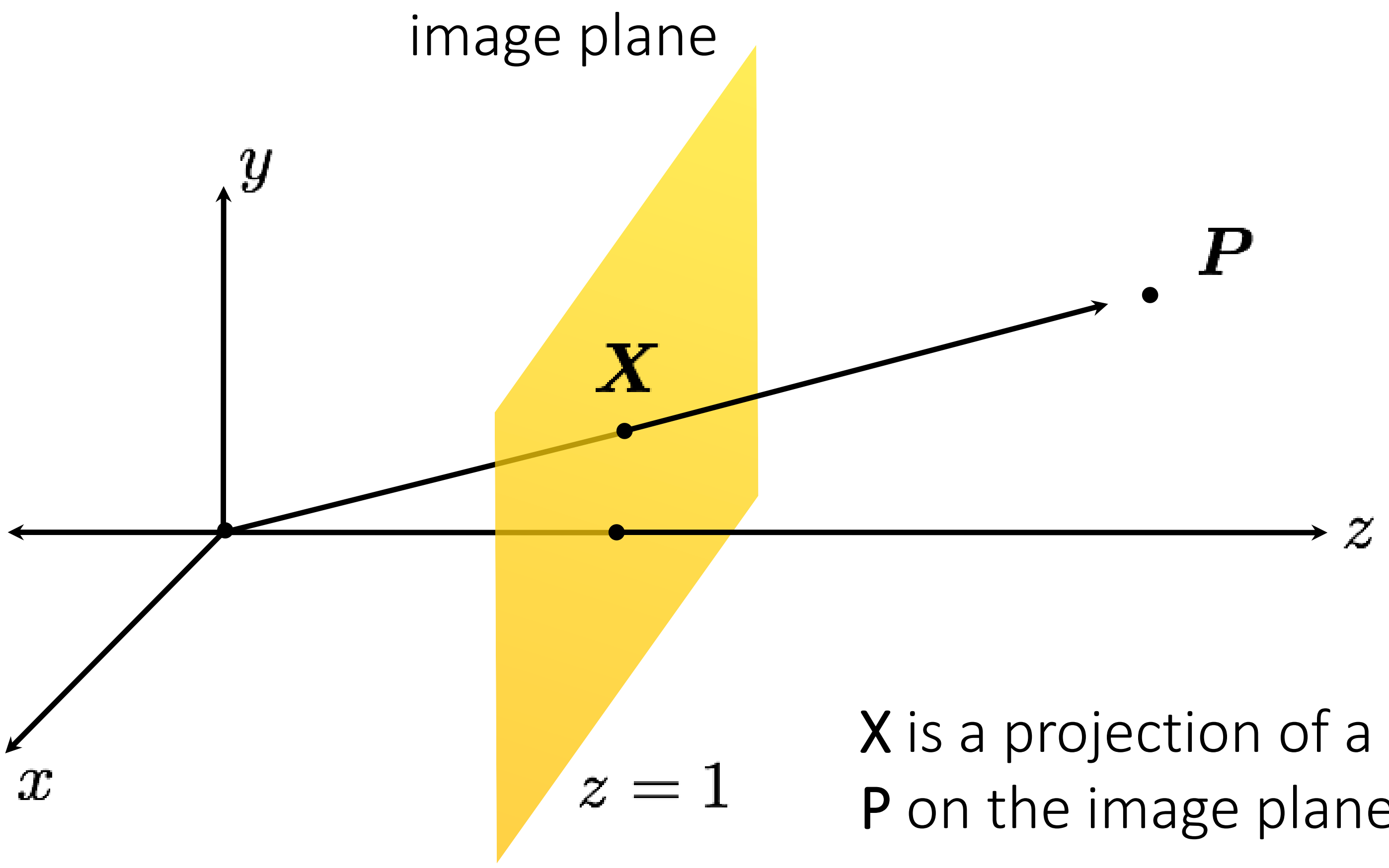
$$\begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

Projective geometry

image point in pixel coordinates $\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$\Updownarrow$

image point in homogeneous coordinates $\boldsymbol{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$



X is a projection of a point P on the image plane

What does scaling 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

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

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

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

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

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

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

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

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:

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

p' = ? ? ? **p**

Matrix composition

Transformations can be combined by matrix multiplication:

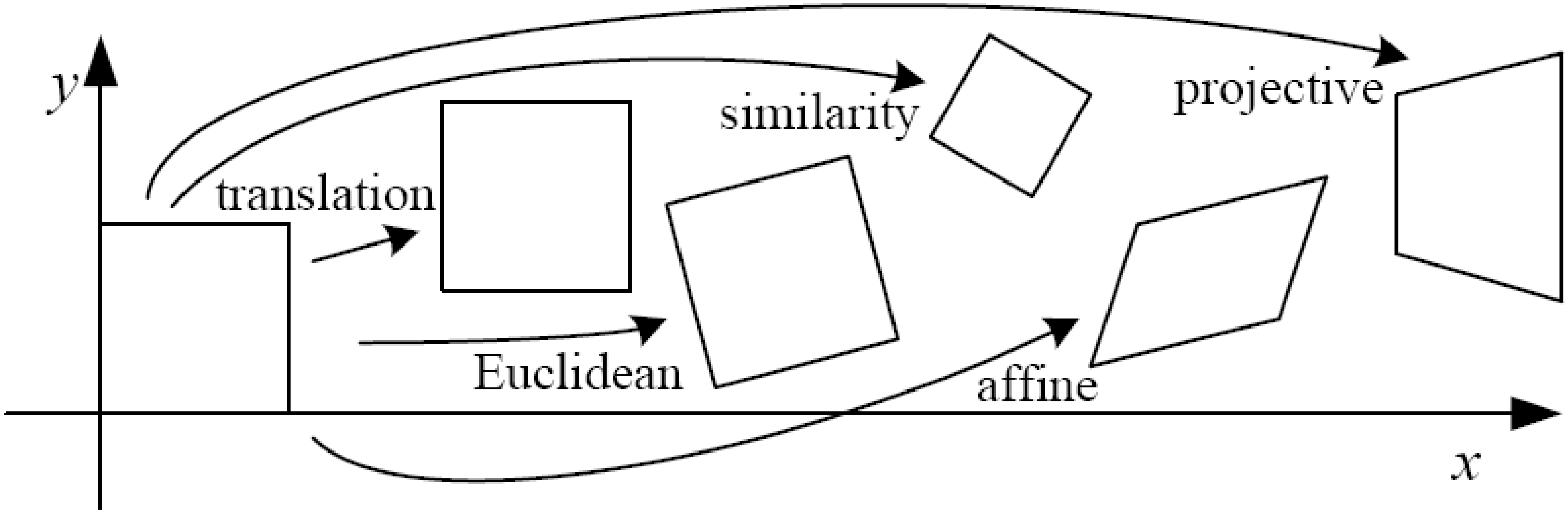
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right)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

p' = translation(t_x,t_y) rotation(θ) scale(s,s) **p**

Does the multiplication order matter?

Classification of 2D transformations

Classification of 2D transformations



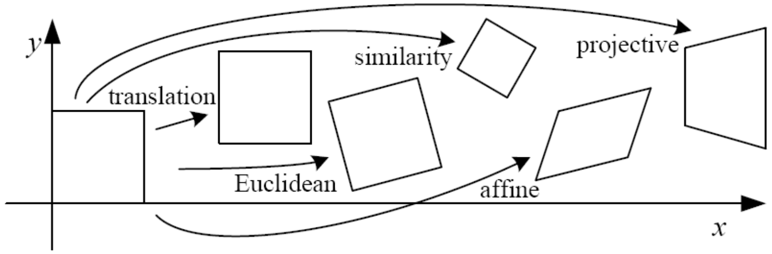
Classification of 2D transformations

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

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?

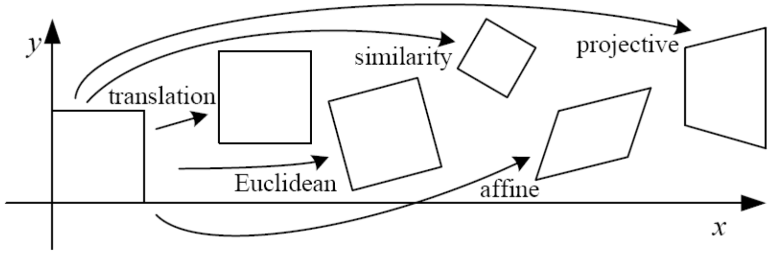


Classification of 2D transformations

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?

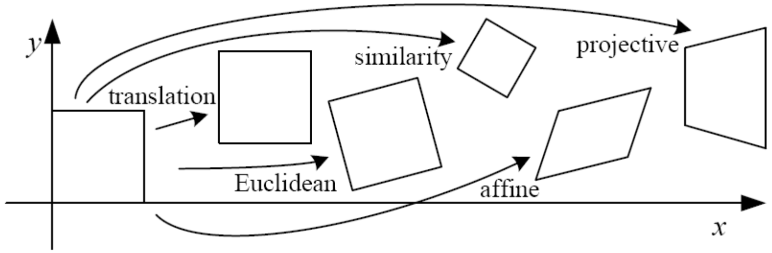


Classification of 2D transformations

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?

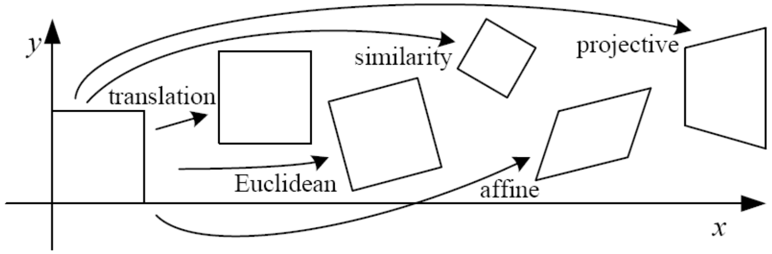


Classification of 2D transformations

which other matrix values will change if this increases?

Euclidean (rigid): rotation + translation

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

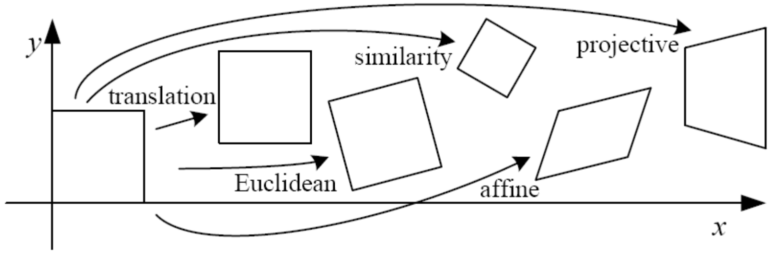


Classification of 2D transformations

what will happen to the image if this increases?

Euclidean (rigid): rotation + translation

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

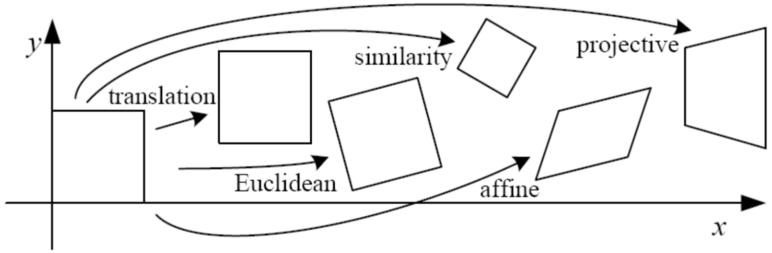


Classification of 2D transformations

what will happen to the image if this increases?

Euclidean (rigid): rotation + translation

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

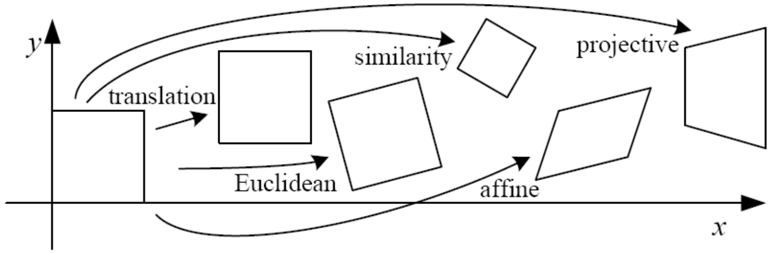


Classification of 2D transformations

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?



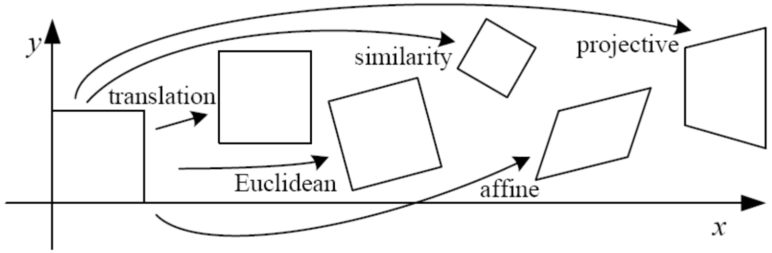
Classification of 2D transformations

multiply these four by scale **s**

Similarity:
uniform scaling + rotation
+ translation

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

How many degrees of freedom?

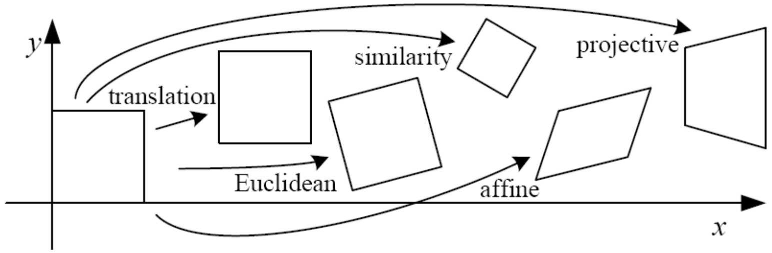


Classification of 2D transformations

what will happen to the image if this increases?

Similarity:
uniform scaling + rotation
+ translation

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

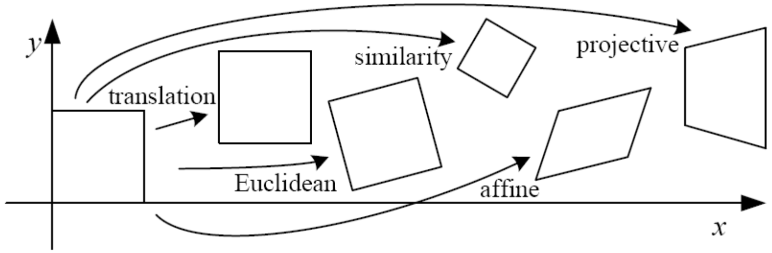


Classification of 2D transformations

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

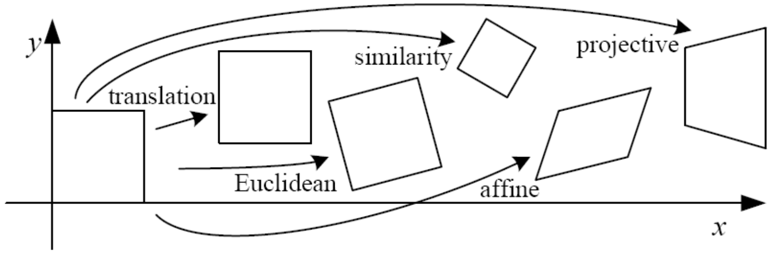
Affine transform:
uniform scaling + shearing
+ rotation + translation

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

Are there any values that are related?

similarity shear

$$\begin{bmatrix} sr_1 & sr_2 \\ sr_3 & sr_4 \end{bmatrix} \begin{bmatrix} 1 & h_1 \\ h_2 & 1 \end{bmatrix} = \begin{bmatrix} sr_1 + h_2 sr_2 & sr_2 + h_1 sr_1 \\ sr_3 + h_2 sr_4 & sr_4 + h_1 sr_3 \end{bmatrix}$$



Classification of 2D transformations

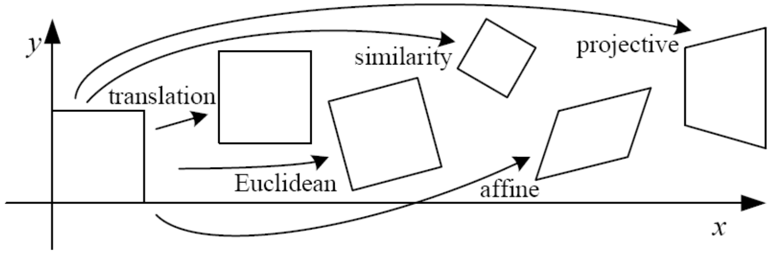
Affine transform:
uniform scaling + shearing
+ rotation + translation

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

How many degrees of freedom?

similarity shear

$$\begin{bmatrix} sr_1 & sr_2 \\ sr_3 & sr_4 \end{bmatrix} \begin{bmatrix} 1 & h_1 \\ h_2 & 1 \end{bmatrix} = \begin{bmatrix} sr_1 + h_2 sr_2 & sr_2 + h_1 sr_1 \\ sr_3 + h_2 sr_4 & sr_4 + h_1 sr_3 \end{bmatrix}$$



Affine transformations

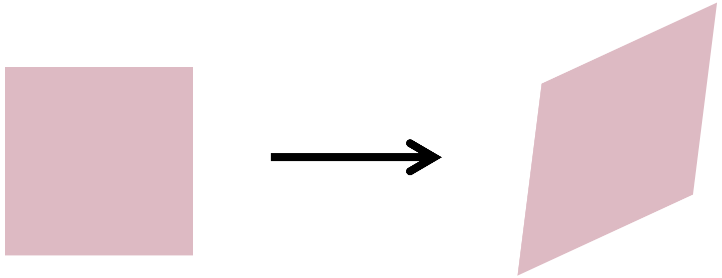
Affine transformations are combinations of

- 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

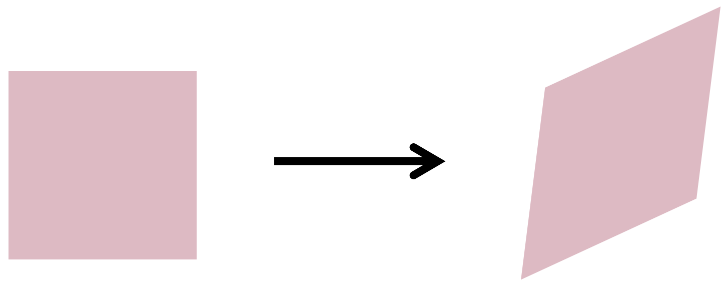
Affine transformations are combinations of

- 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



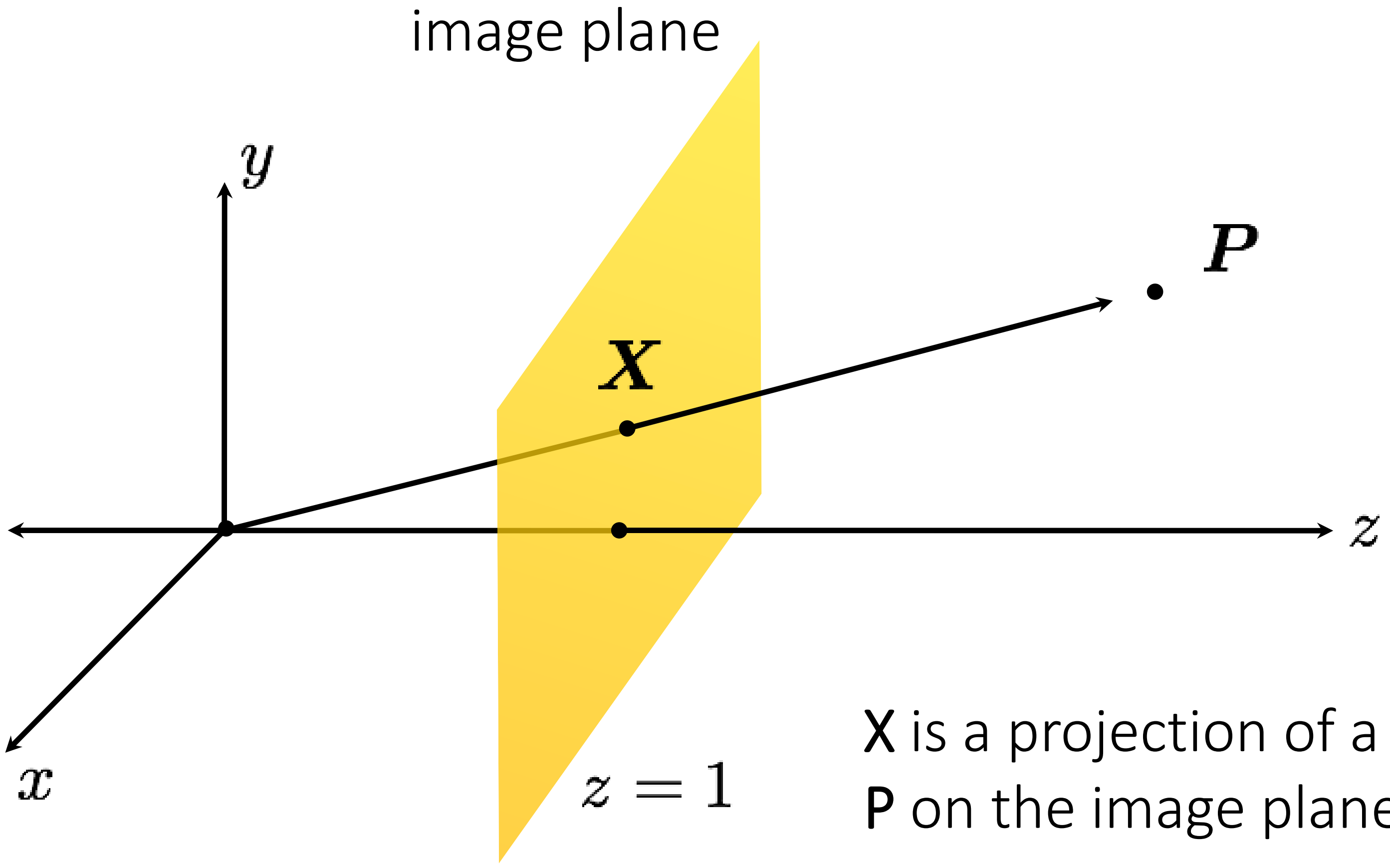
Nope! But what does that mean?

How to interpret affine transformations here?

image point in pixel coordinates $\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$\Updownarrow$$

image point in heterogeneous coordinates $\boldsymbol{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$



X is a projection of a point P on the image plane

Projective transformations

Projective transformations are combinations of

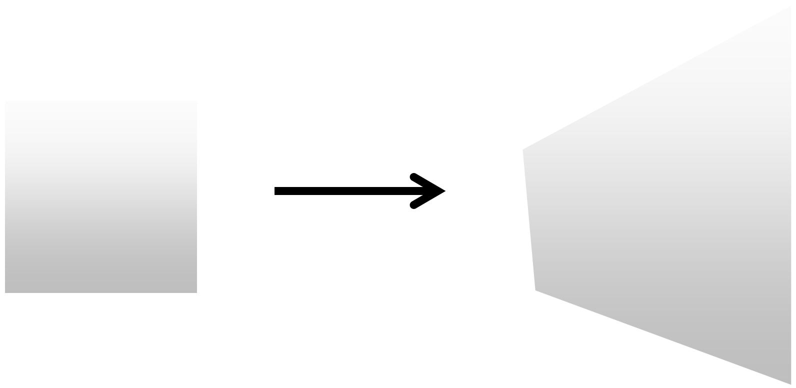
- affine transformations; and
- projective wraps

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

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

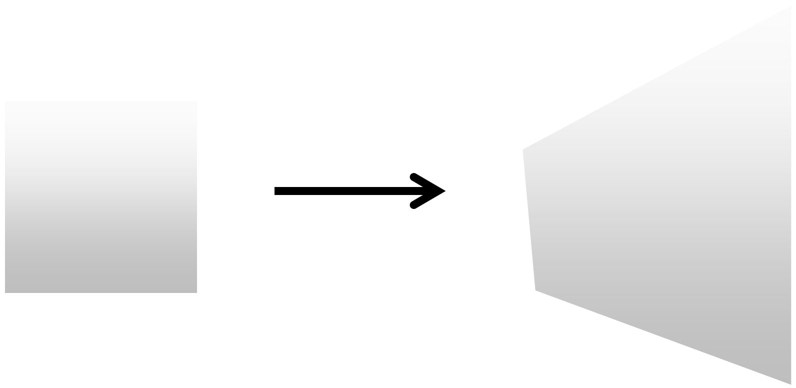
- affine transformations; and
- projective wraps

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

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

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

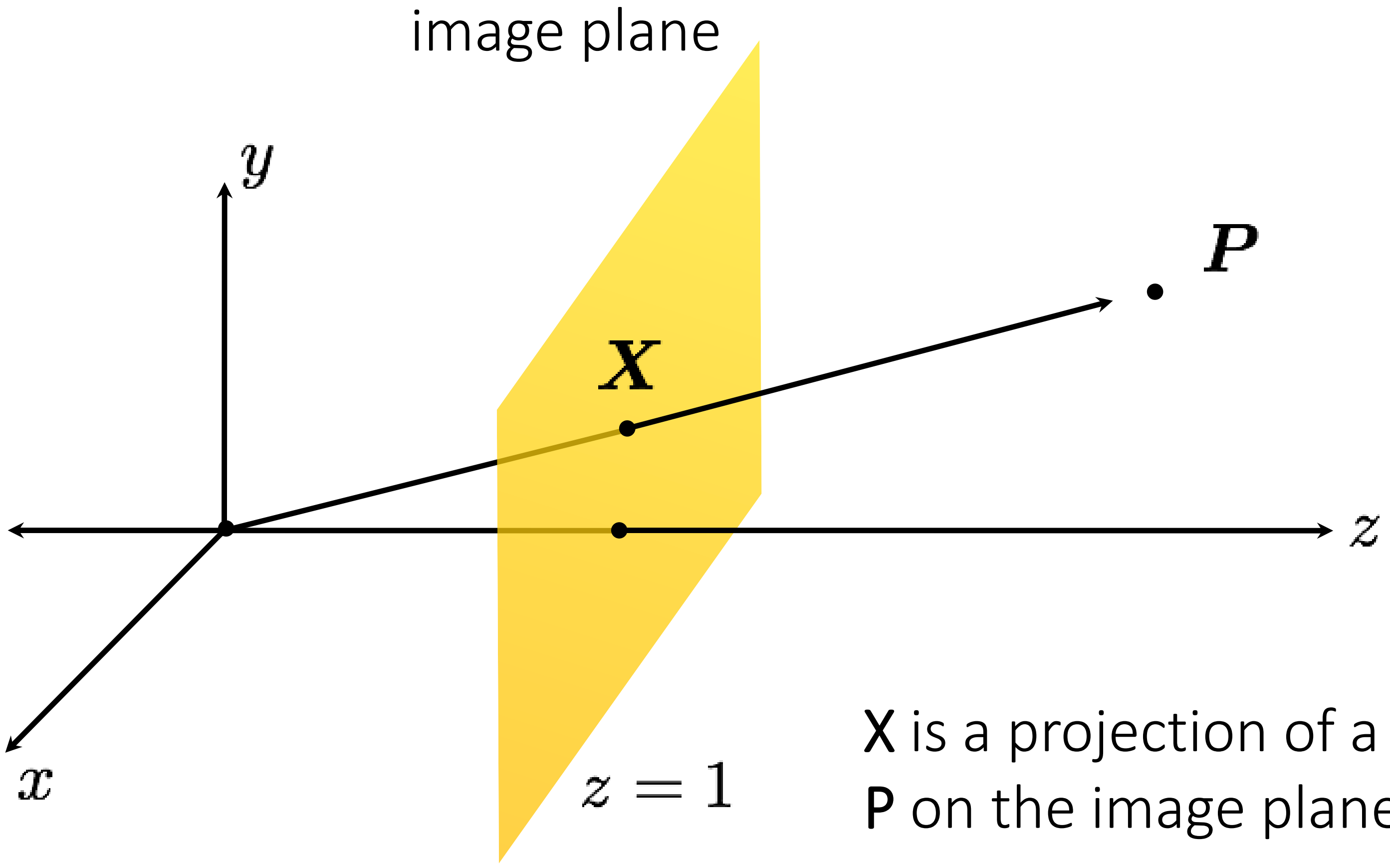


How to interpret projective transformations here?

image point in pixel coordinates $\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$\Updownarrow$$

image point in heterogeneous coordinates $\boldsymbol{X} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

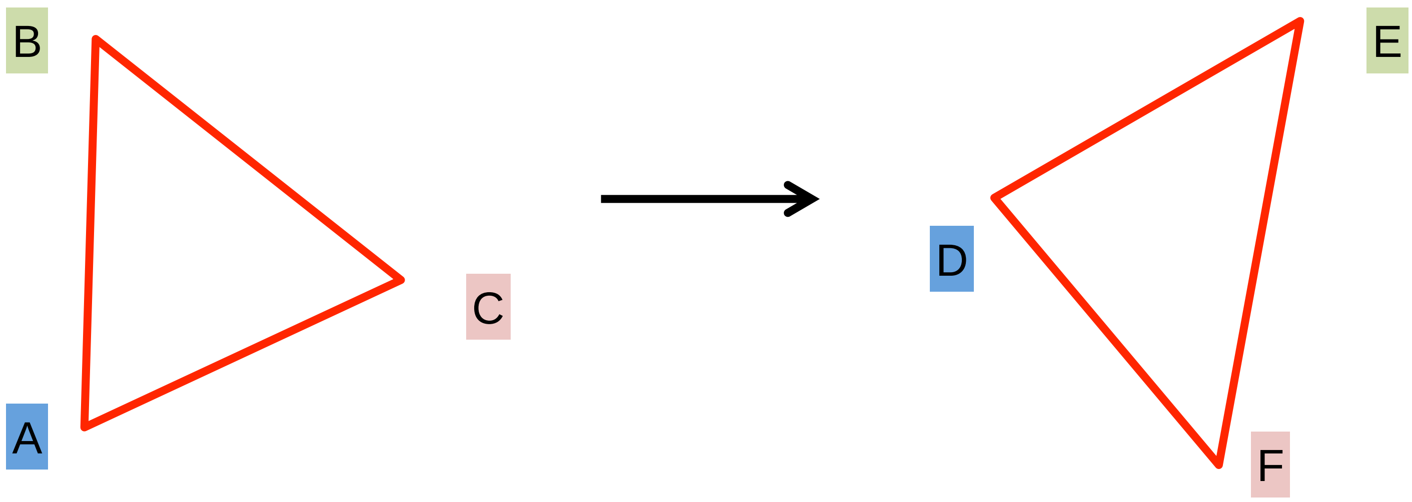


X is a projection of a point P on the image plane

Determining unknown 2D transformations

Determining unknown transformations

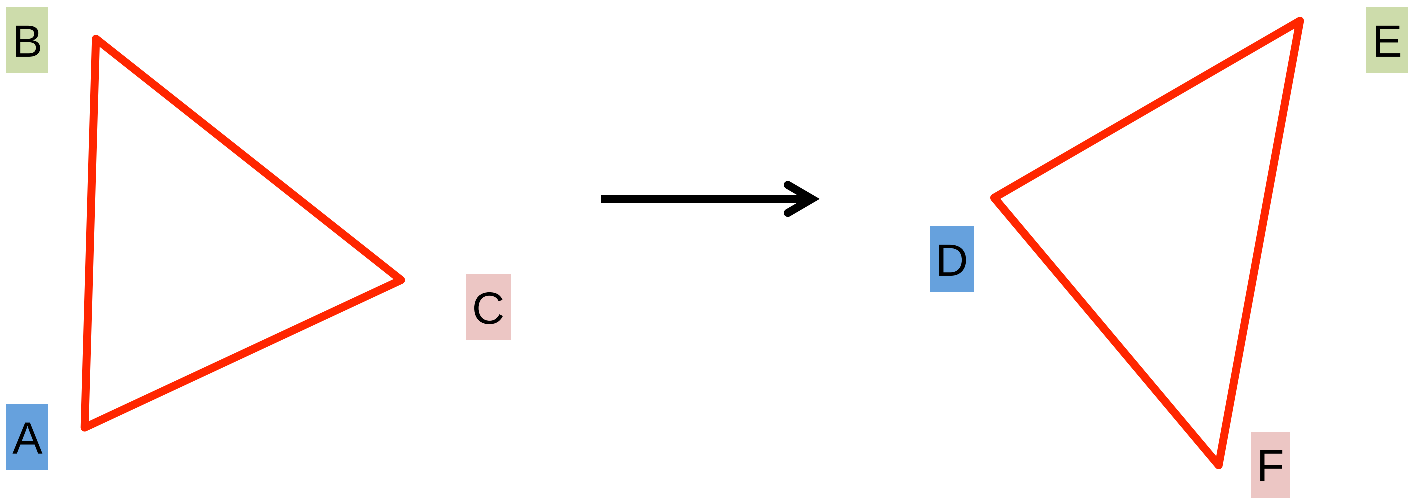
Suppose we have two triangles: ABC and DEF.

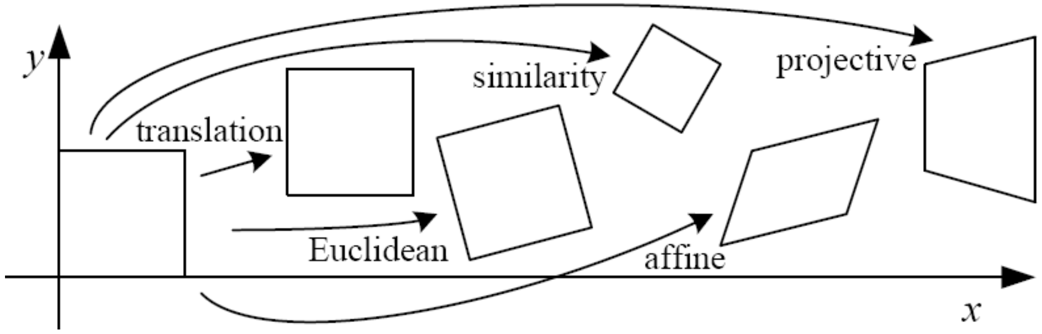


Determining unknown transformations

Suppose we have two triangles: ABC and DEF.

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

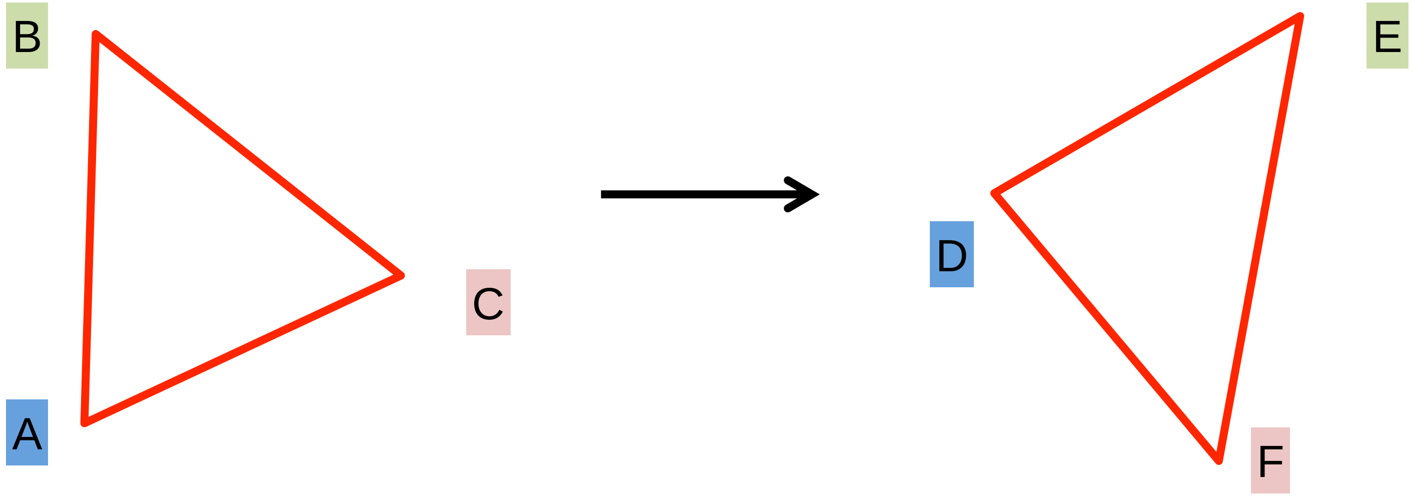




Determining unknown transformations

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

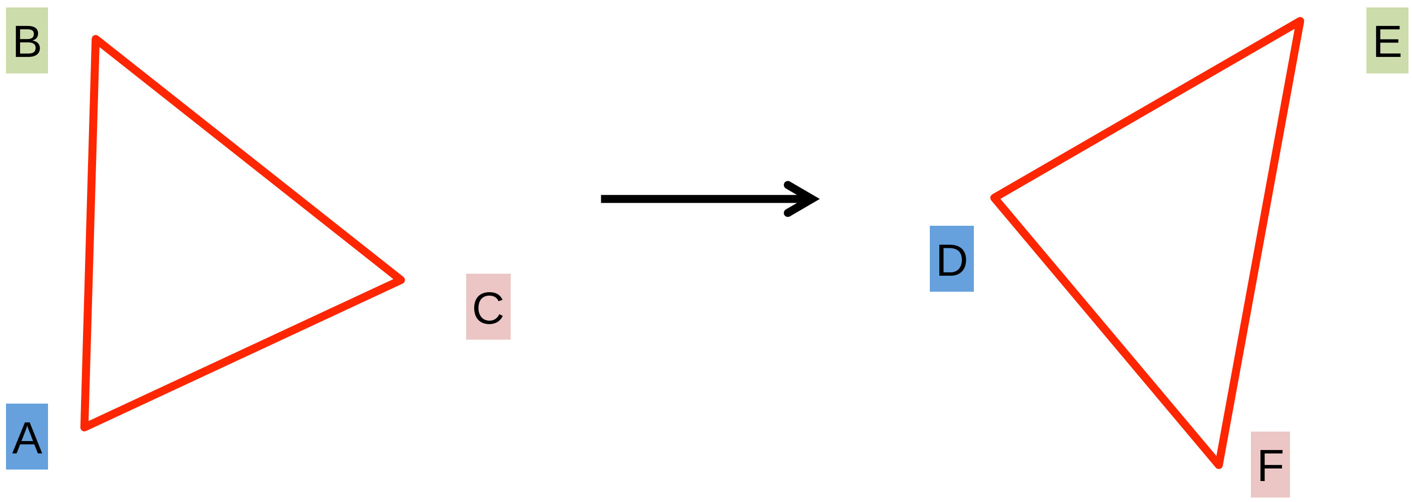
$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

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

$$\boldsymbol{x}' = \mathbf{M}\boldsymbol{x}$$

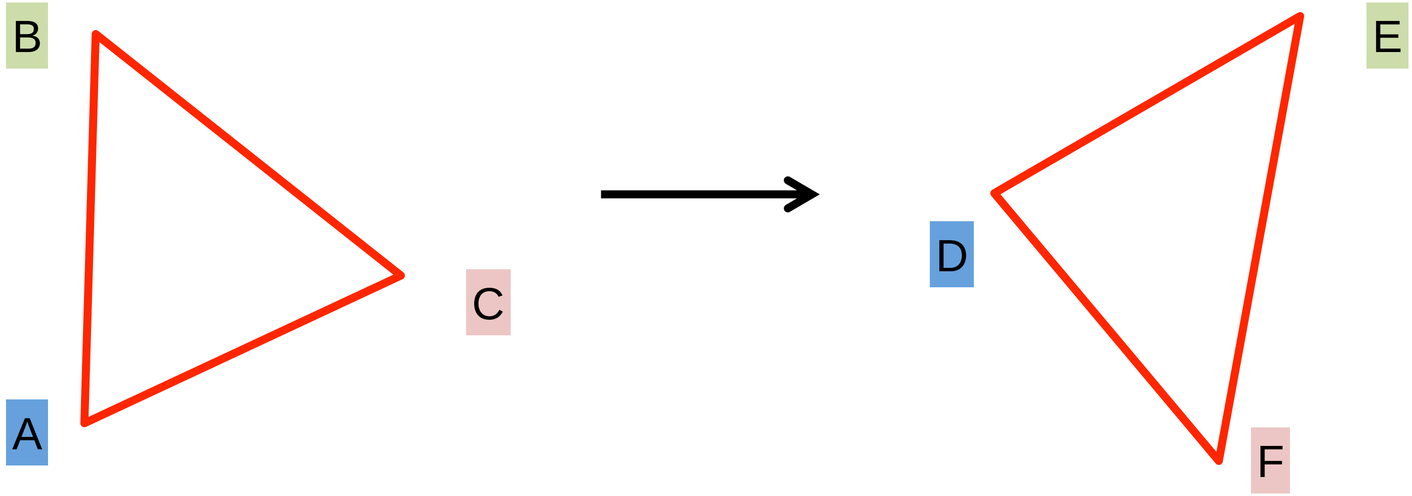
point correspondences

- 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?

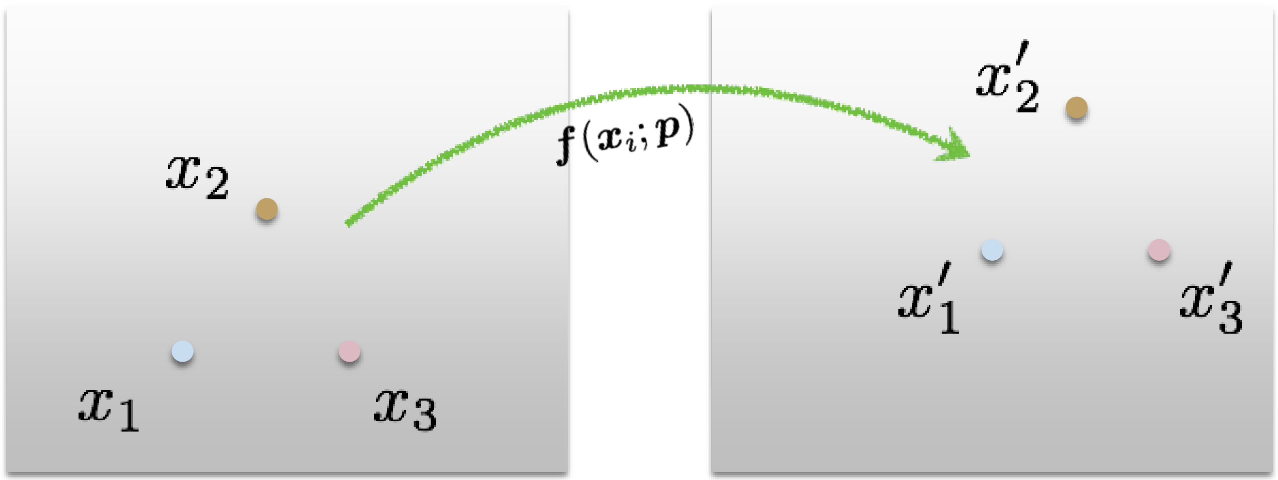


unknowns

$$\boldsymbol{x}' = \mathbf{M}\boldsymbol{x}$$

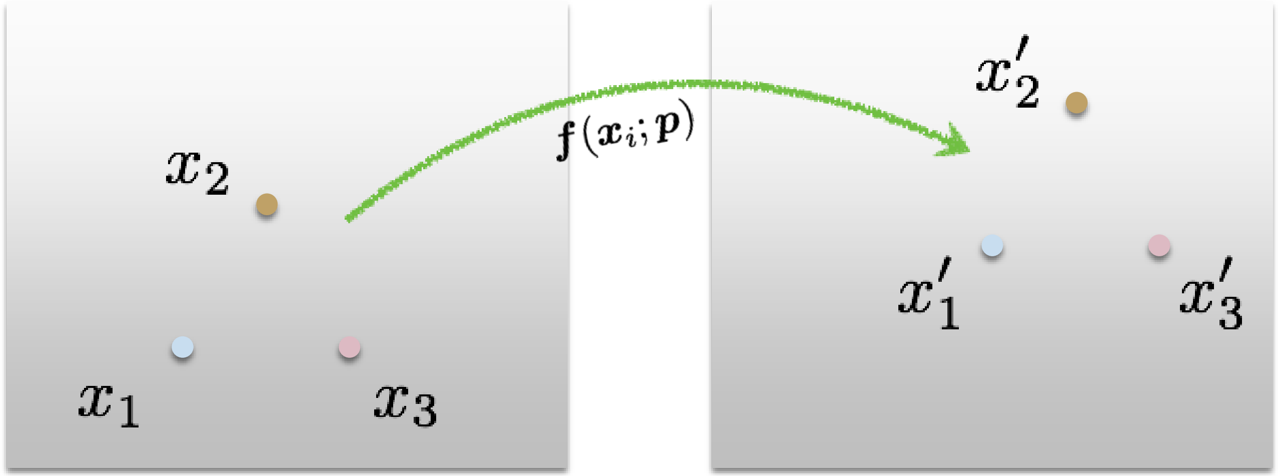
point correspondences

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



Least Squares Error

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



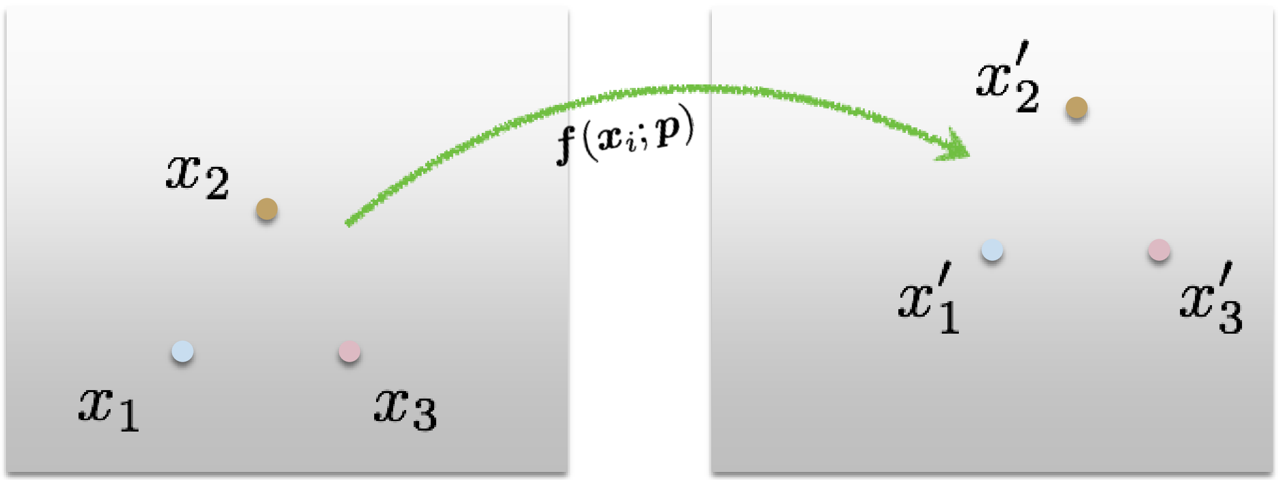
Least Squares Error

What is this?

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

What is this?

What is this?



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

Least Squares Error

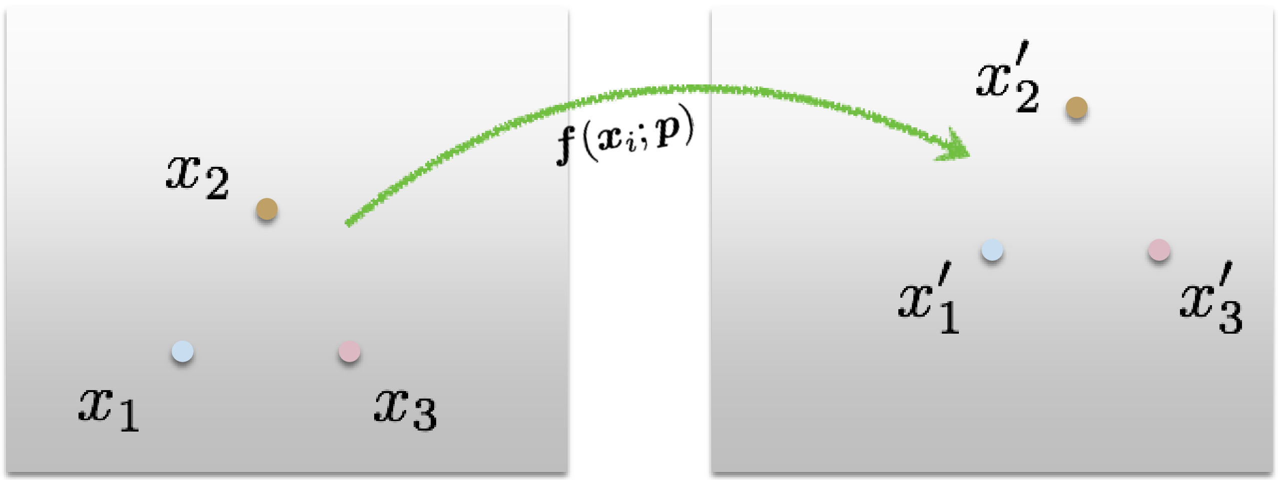
Euclidean (L2) norm

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

squared!

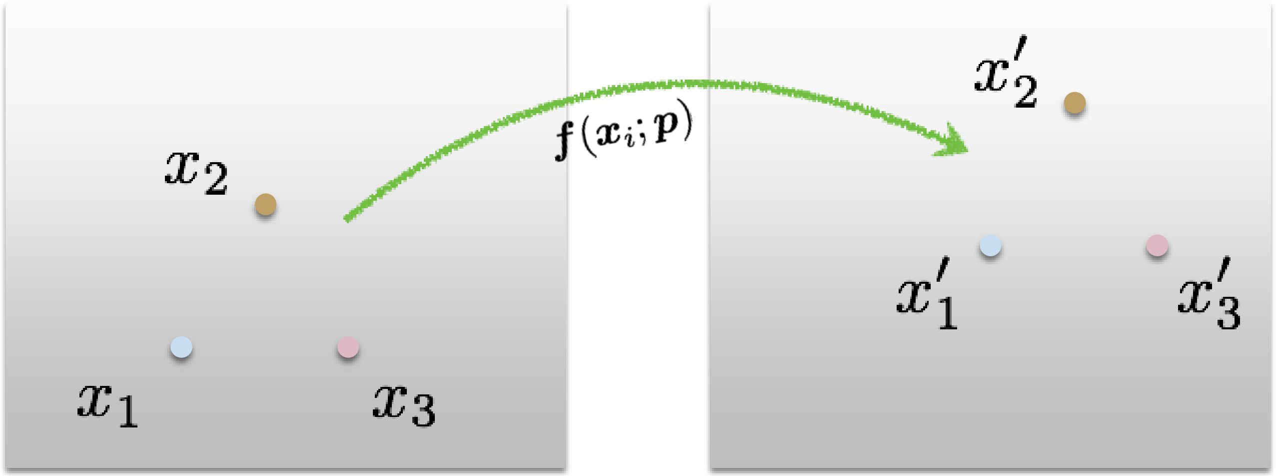
predicted location

measured location



Least Squares Error

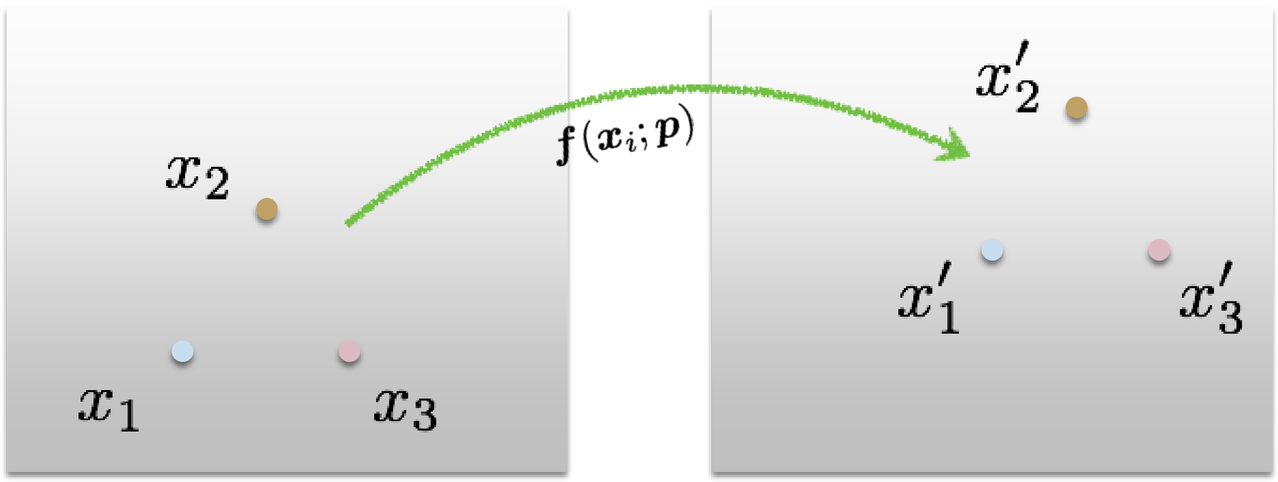
$$E_{\mathrm{LS}} = \sum_i \underbrace{\| f(x_i; p) - x'_i \|}_{\text{Residual (projection error)}}{}^2$$



Least Squares Error

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

What is the free variable?
What do we want to optimize?



Find parameters that minimize squared error

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

General form of linear least squares

(**Warning:** change of notation. x is a vector of parameters!)

$$E_{\mathrm{LLS}} = \sum_i |\boldsymbol{a}_i \boldsymbol{x} - \boldsymbol{b}_i|^2$$

$$= \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|^2 \qquad \text{(matrix form)}$$

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:

Stack equations from point correspondences:

Notation in system form:

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

General form of linear least squares

(**Warning:** change of notation. x is a vector of parameters!)

$$E_{\mathrm{LLS}} = \sum_i |\boldsymbol{a}_i \boldsymbol{x} - \boldsymbol{b}_i|^2$$

$$= \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|^2 \quad \text{(matrix form)}$$

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:

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

Expand the error:

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

Minimize the error:

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

Solve for x $\boldsymbol{x} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\boldsymbol{b} \longleftarrow$ Note: You almost never want to compute the inverse of a matrix.

In Matlab:

```
x = A \ b
```

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

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

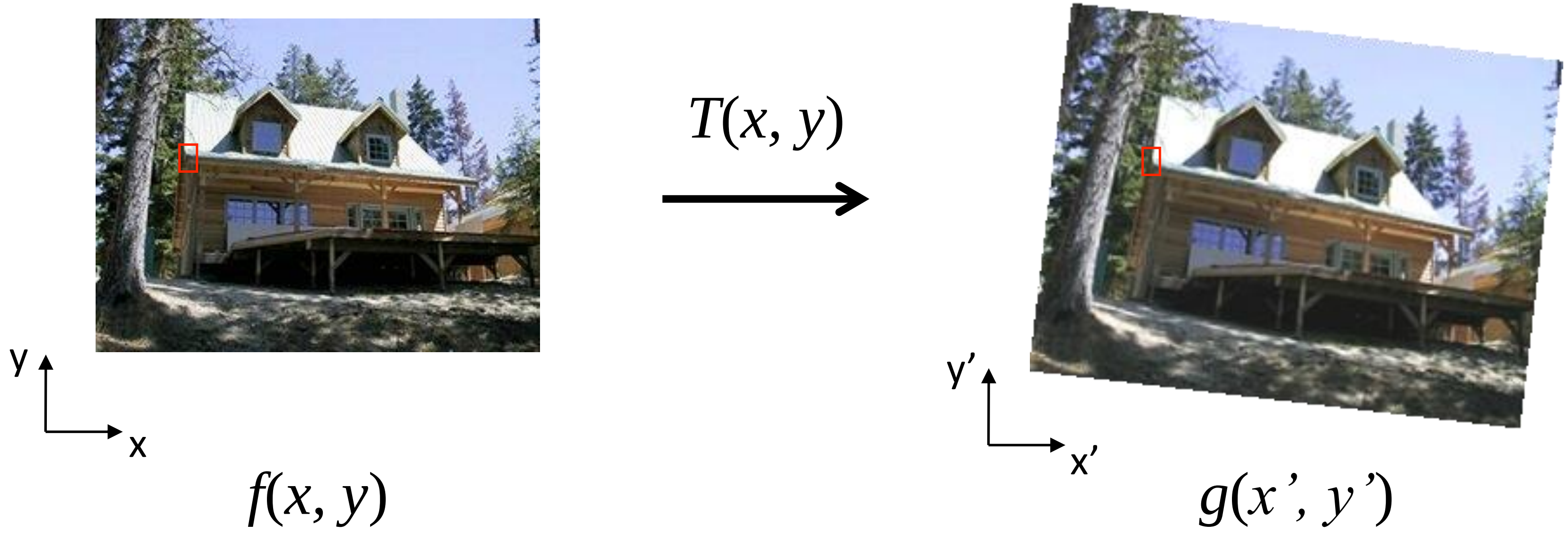
Also doesn’t deal well with outliers

Determining unknown image warps

Determining unknown image warps

Suppose we have two images.

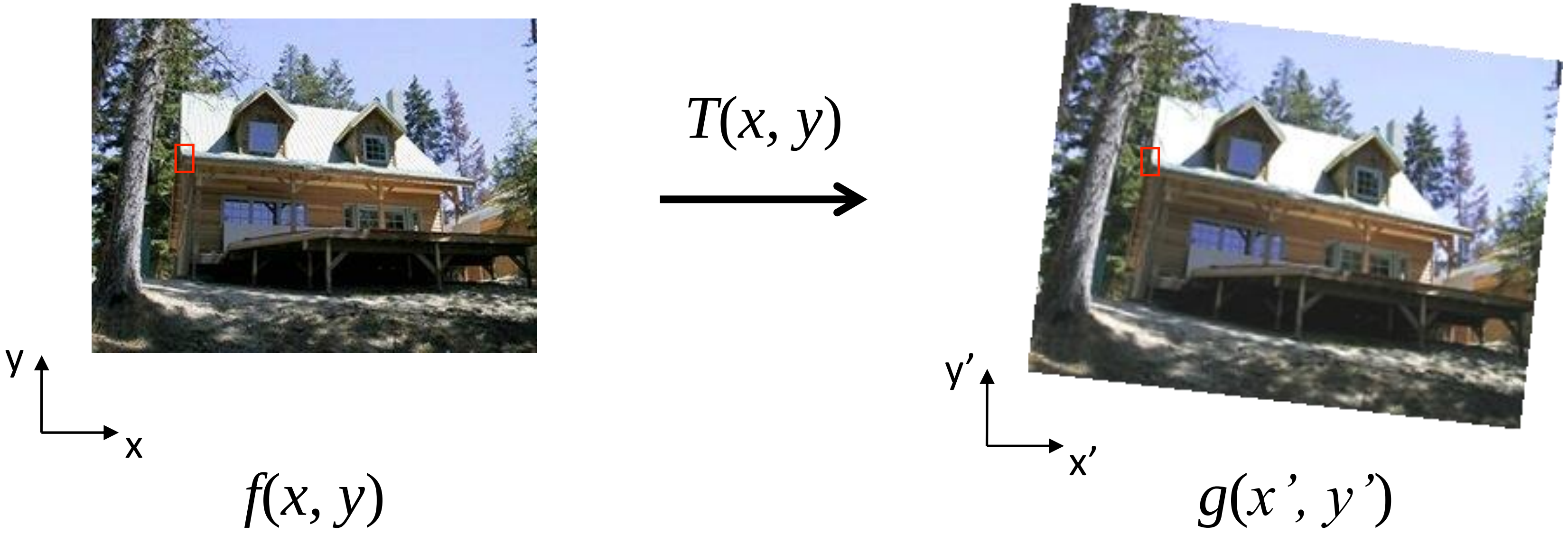
- How do we compute the transform that takes one to the other?



Forward warping

Suppose we have two images.

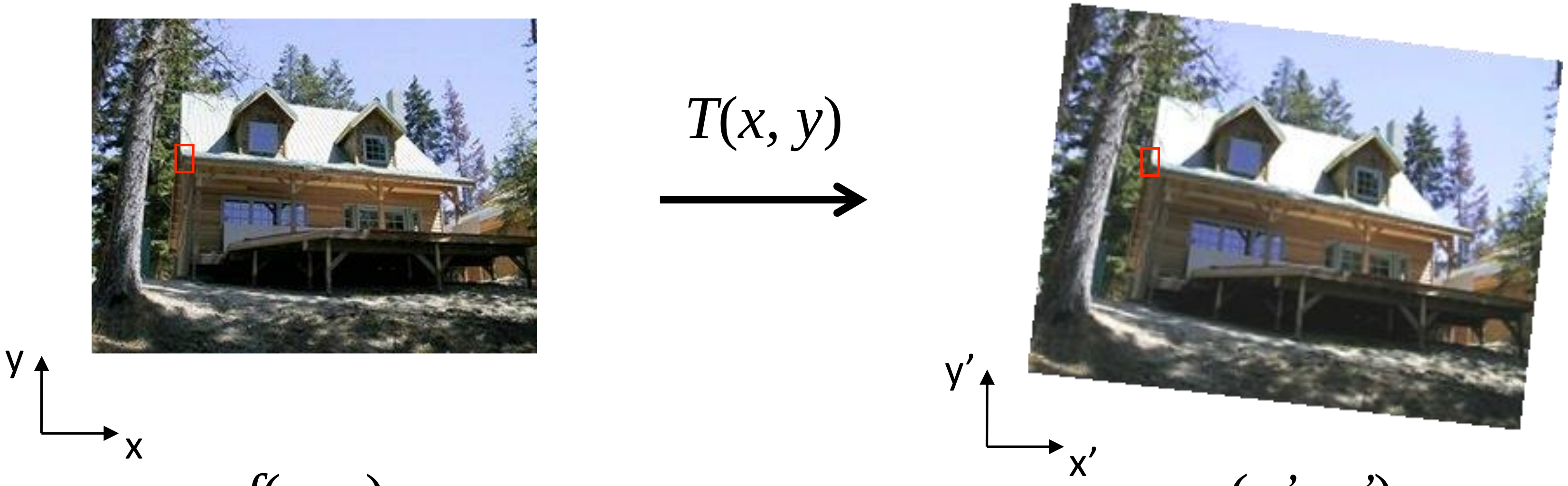
- How do we compute the transform that takes one to the other?



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?



$f(x, y)$ $g(x', y')$

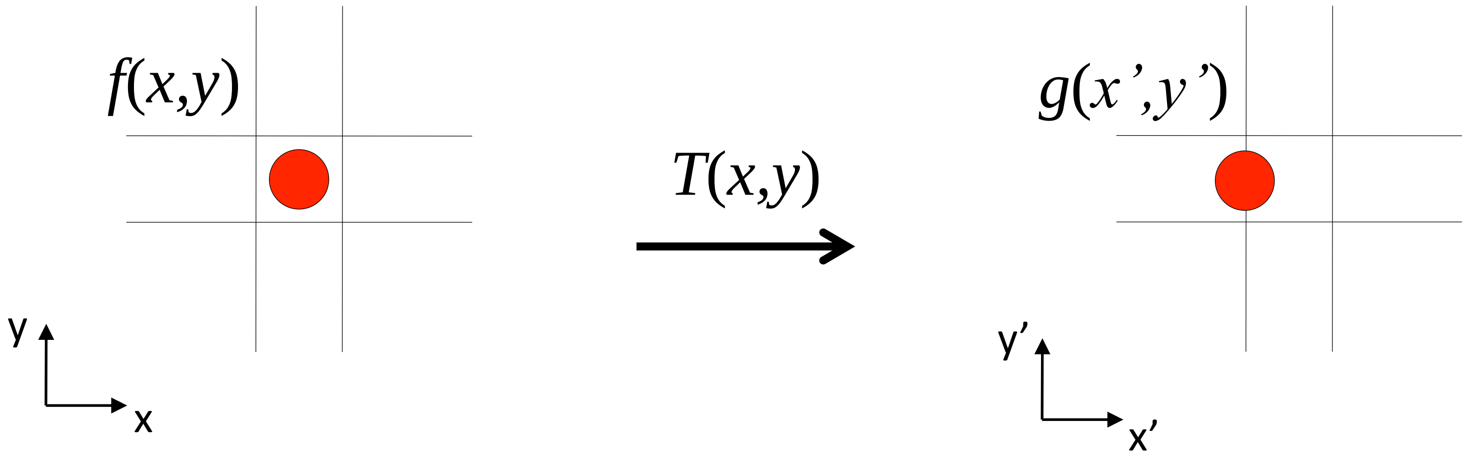
what is the problem with this?

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?

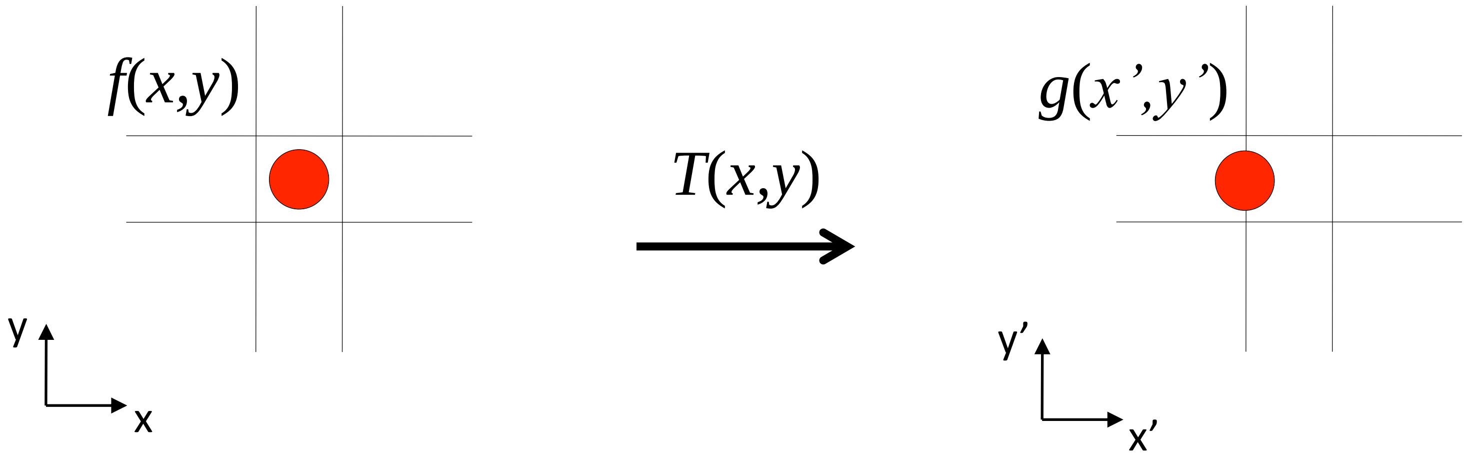


Forward warping

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”)



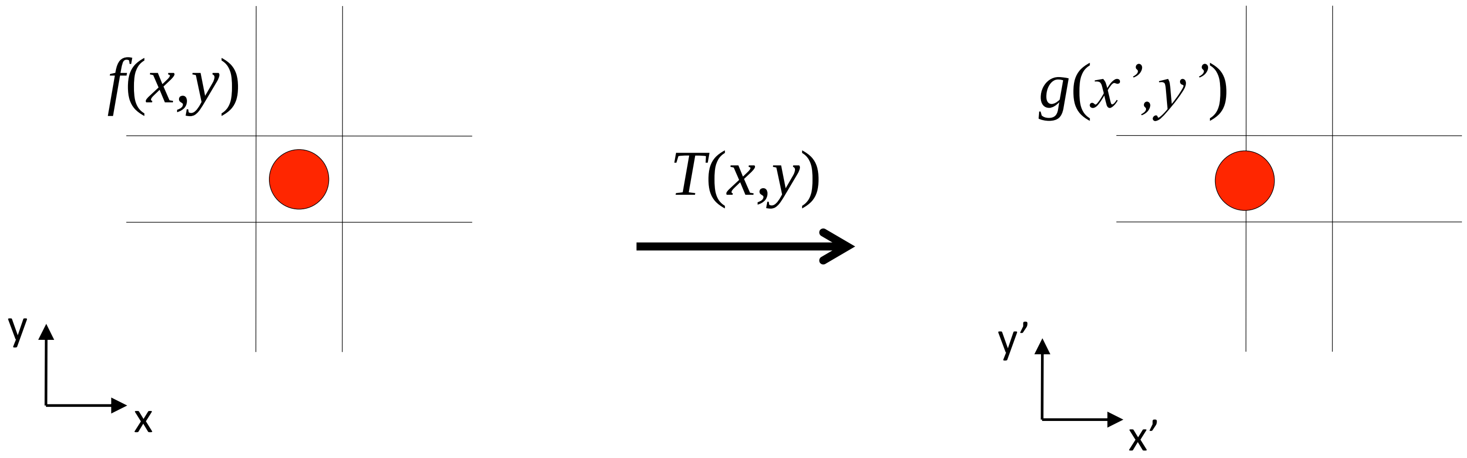
- What if a pixel (x’,y’) receives intensity from more than one pixels (x,y)?

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?

$T^{-1}(x, y)$ ←

y, x: $f(x, y)$

y', x': $g(x', y')$

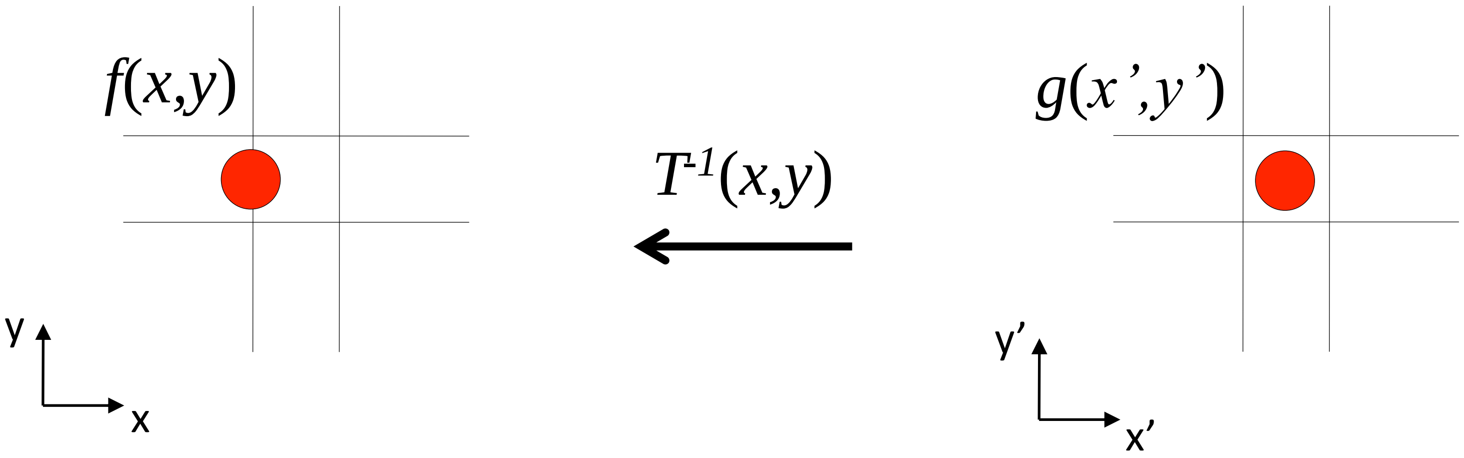
what is the problem with this?

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 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?

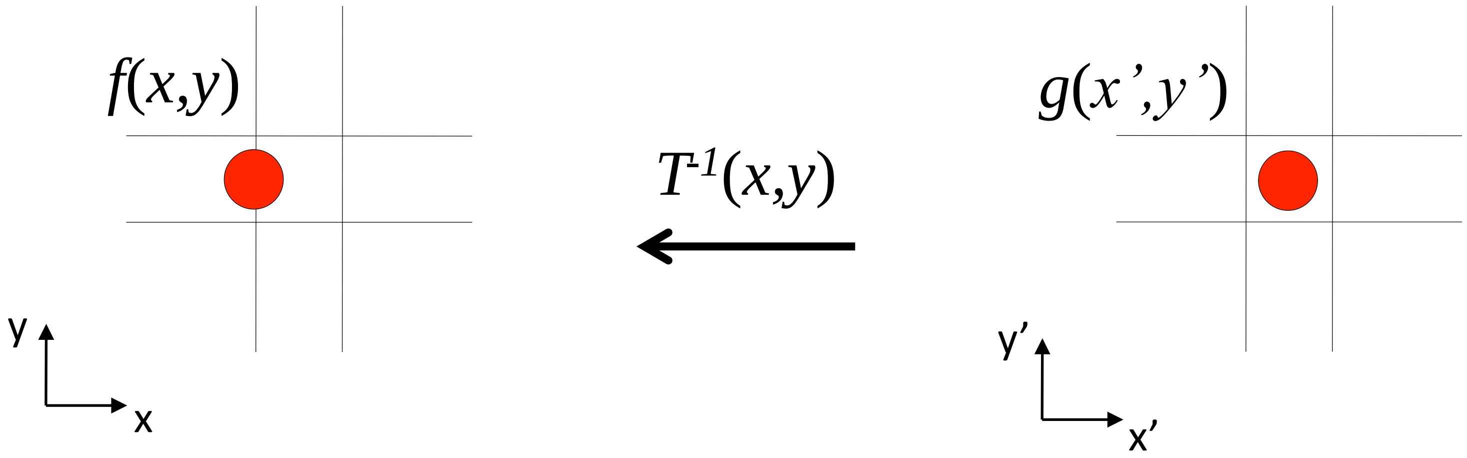


Inverse warping

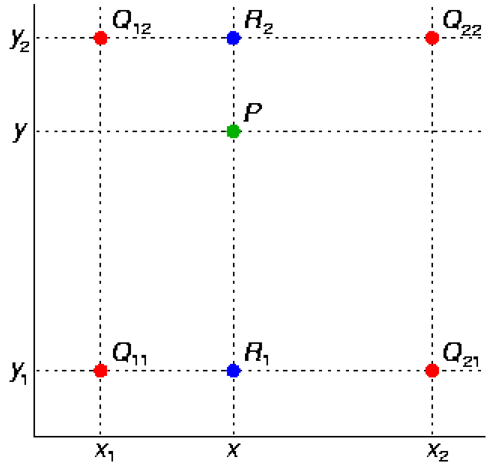
Pixel may come from between two points

- How do we determine its intensity?

✓ Use interpolation

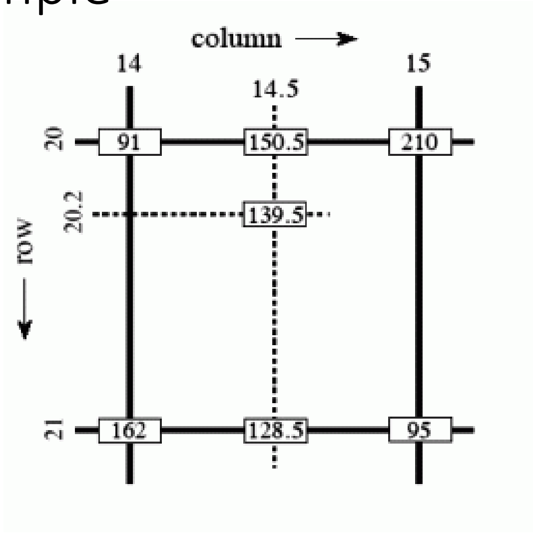


Bilinear interpolation



1. `Interpolate to find R2`
2. `Interpolate to find R1`
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?

y

x

$f(x, y)$

$$T(x, y)$$

$$\longleftrightarrow$$

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

y'

x'

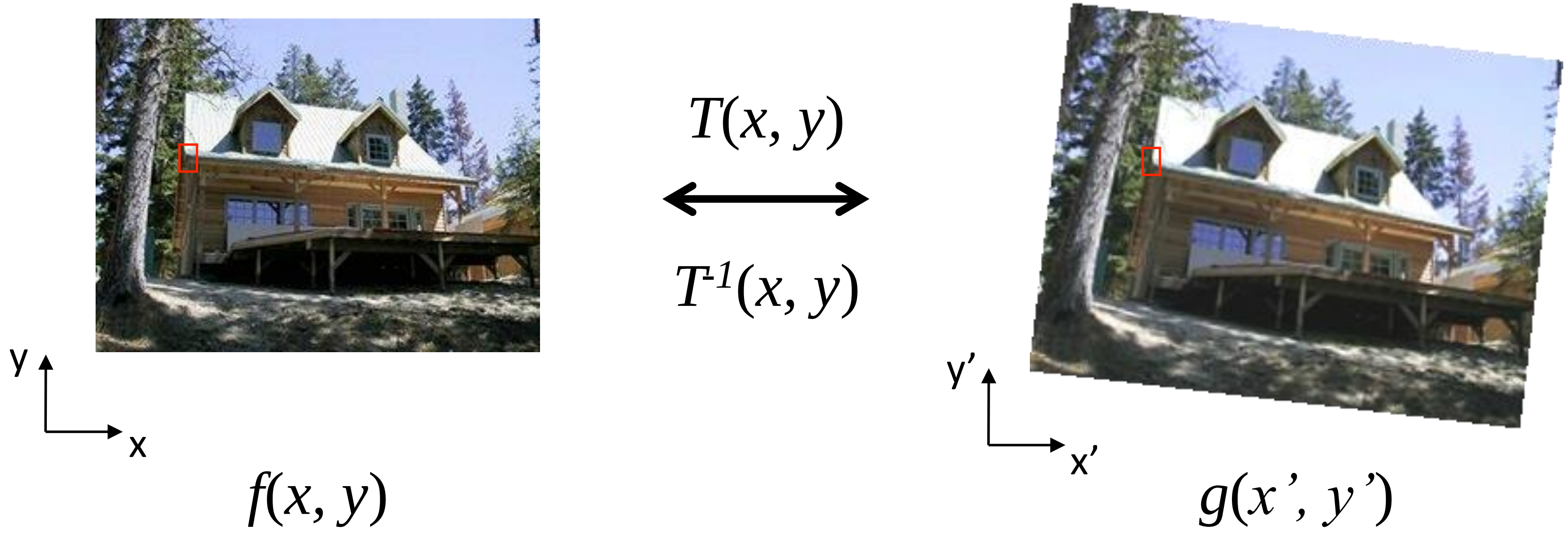
$g(x', y')$

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).