Computer Vision and Image Processing

Gil Ben-Artzi

5

Agenda

- Topic 1
  - Image Enhancement: histogram, quantization
- Topic 2
  - Filtering: smoothing, median filtering, sharpening
  - Low level detection: Template matching, Edges, Line, Circles
- Topic 3
  - Image Pyramids and Blending,
- Topic 4
  - Optical Flow
- Topic 5
  - Geometry: 2D Transformations, Image Warping

Image Warping

Image Warping: 2D->2D

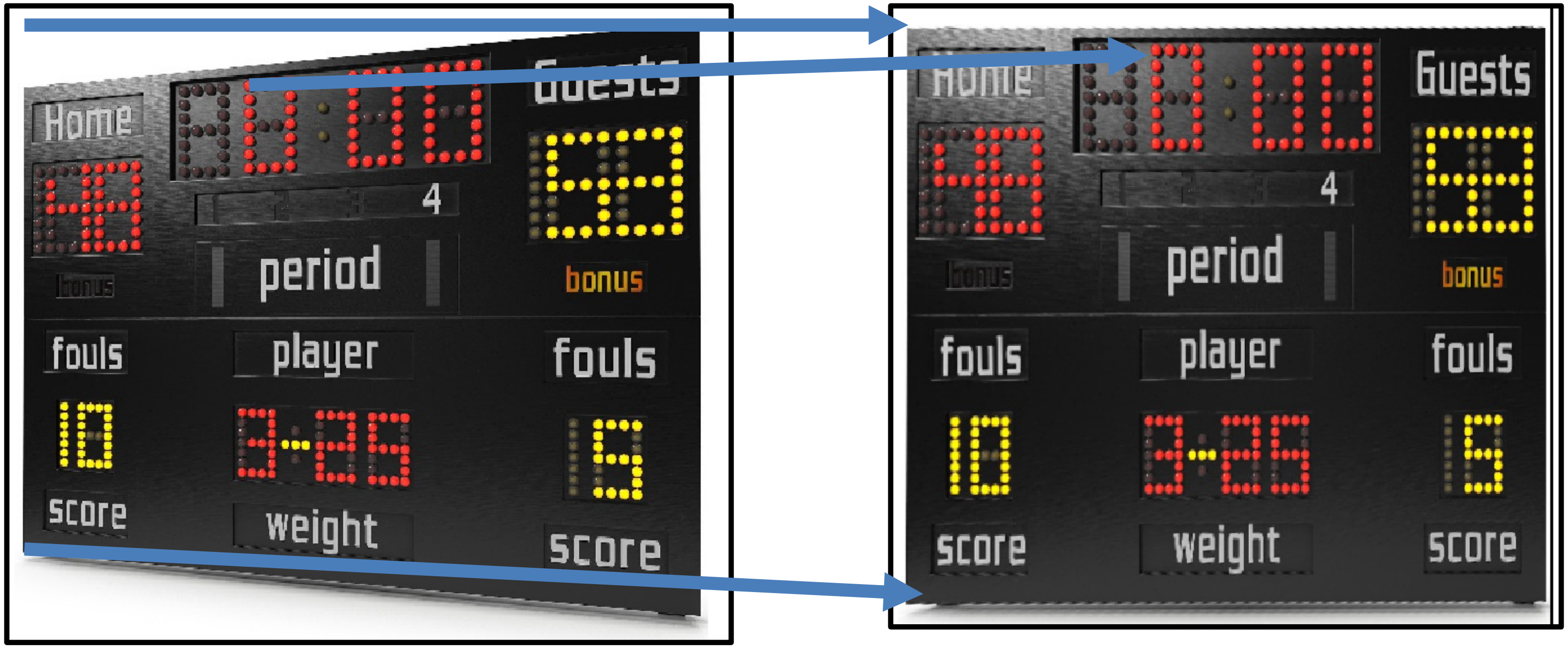


Image Warping

image filtering: change the ***intensities*** of the image

$$g(x,y) = T(\ f(x,y)\ )$$

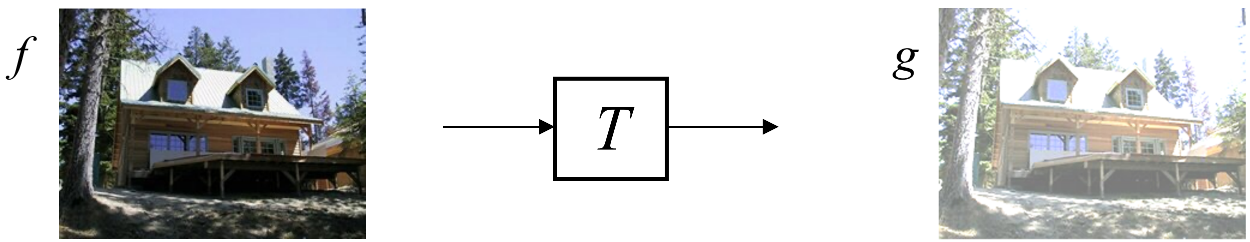


image warping: change the ***coordinates*** of the image

$$g(x,y) = f\,(T(x,y)\ )$$

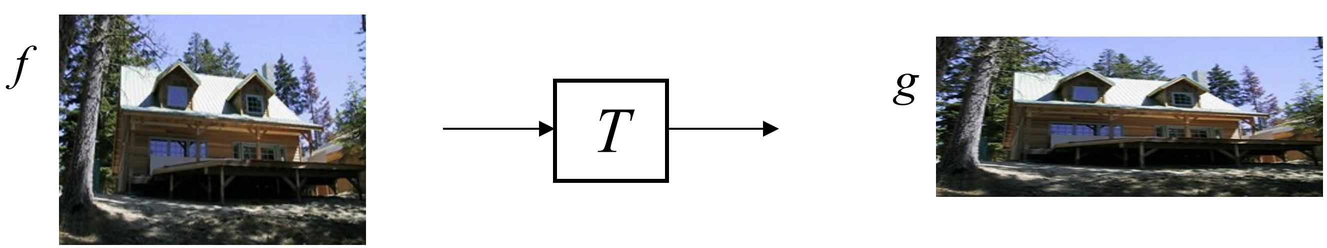
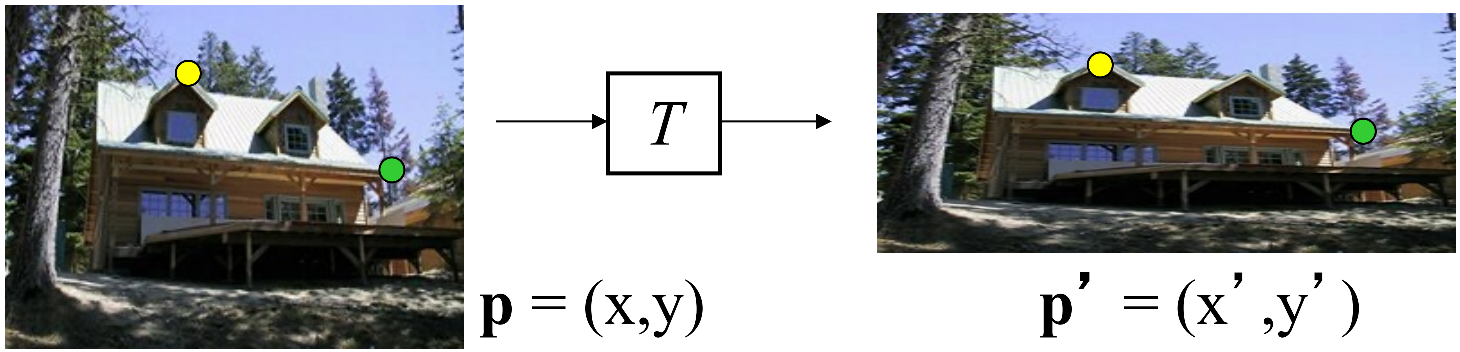


Image Warping



$\mathbf{p} = (x,y)$ $\quad$ $\mathbf{p}' = (x',y')$

Transformation T is a **coordinate-changing** machine:

$$p' = T(p)$$

- Two types of transformations
  - Local
  - Global
- Local transformation is different for each position and is usually described as a *vector field (Flow)*

Global Parametric Warping

$\longrightarrow \boxed{T} \longrightarrow$

$\mathbf{p} = (x,y)$ $\mathbf{p}' = (x',y')$

Global transformation *T*:

- Has the same form for any point *p*
- Determined by just a few numbers (parameters)

We will represent *T* by a matrix: p' = **M**p

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

Global Parametric Warping

Examples of parametric warps:

translation

scaling

rotation

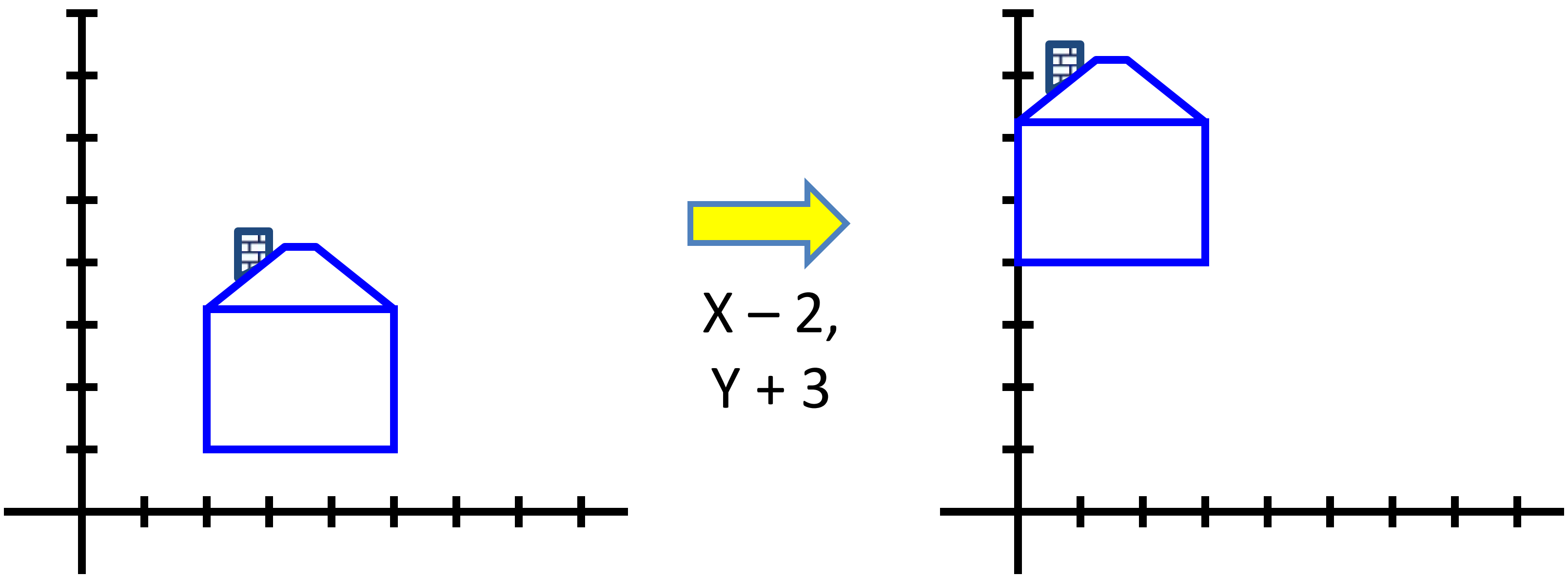
mirror

shear

perspective

Translation

Translation of a coordinate means adding a scalar to each of its components:

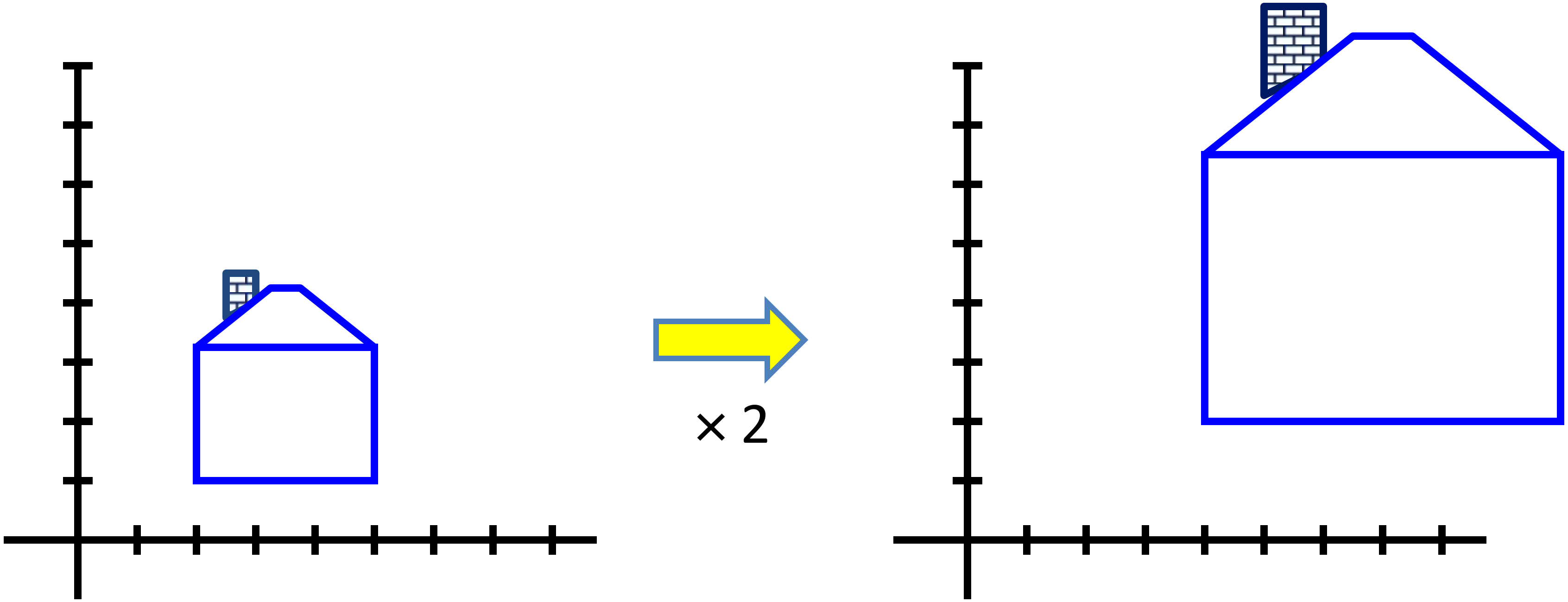


Q: How to represent in matrix form?

Scaling

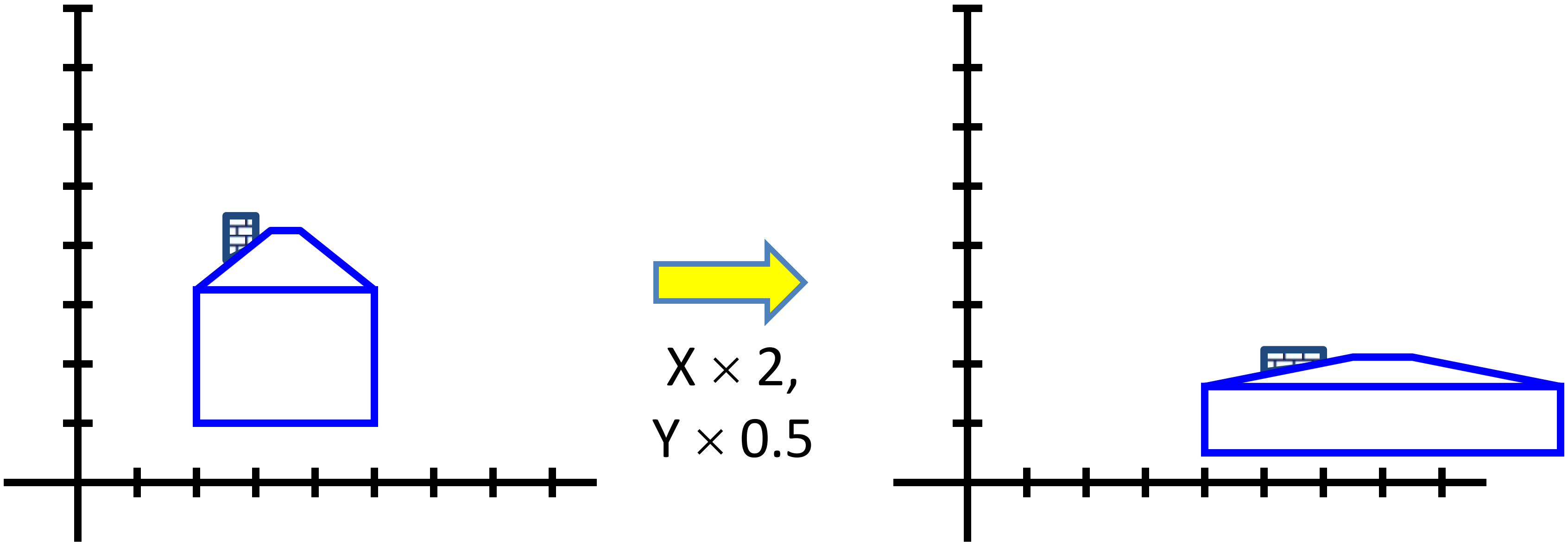
Scaling a coordinate means multiplying each of its components by a scalar

Uniform scaling means this scalar is the same for all components (dimensions):



Scaling

Non-uniform scaling: different scalars per component (dimension):



Scaling

The scaling operation:

$$x' = ax$$

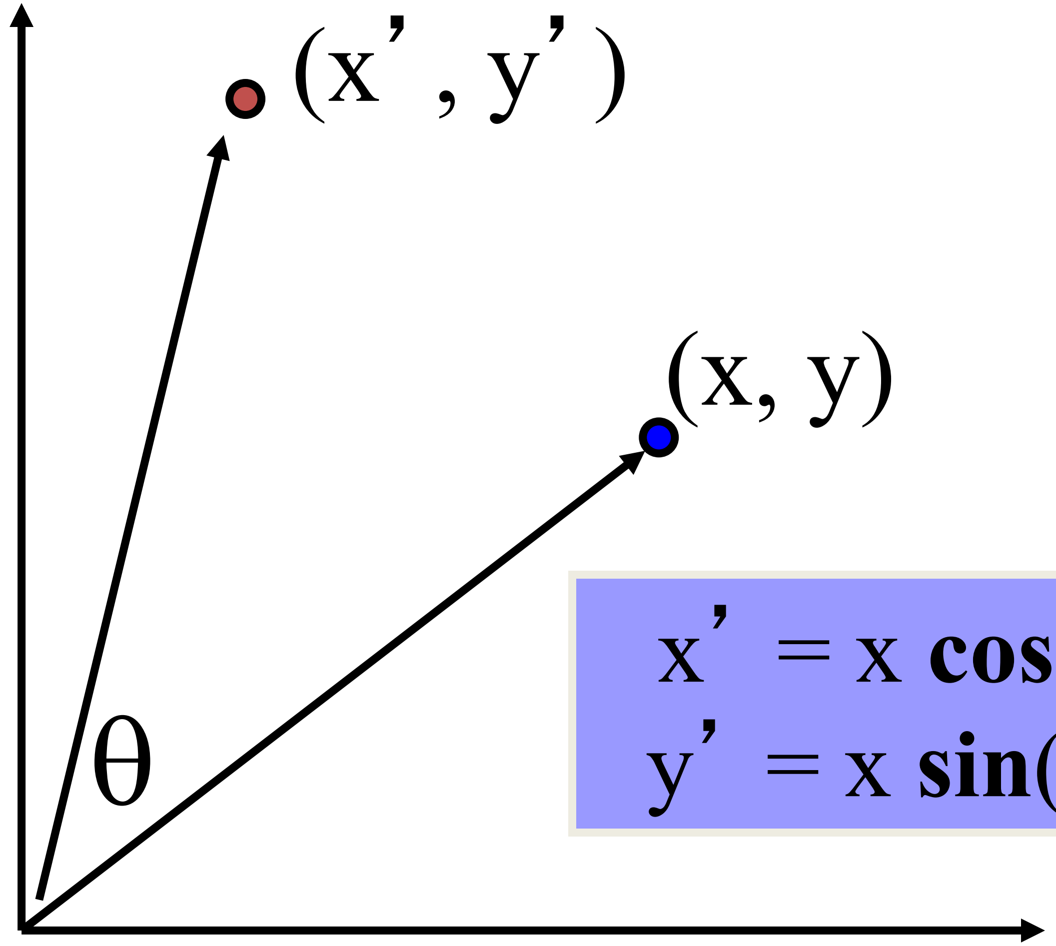
$$y' = by$$

In matrix form:

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

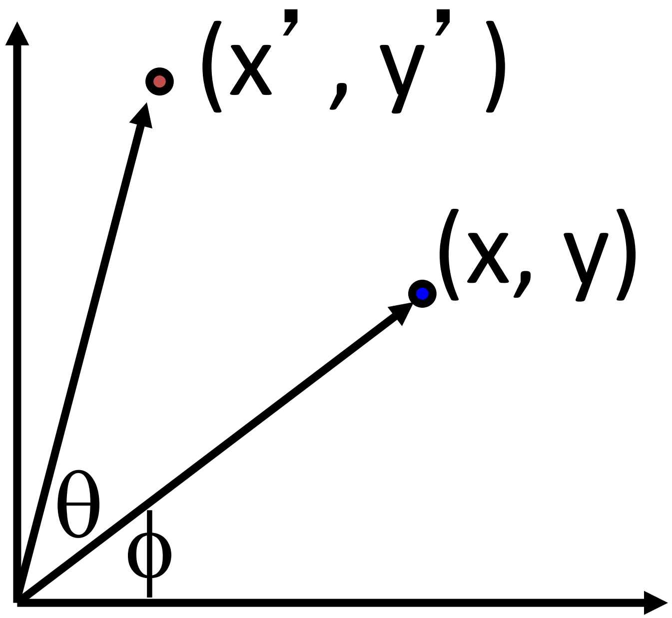
(What is the inverse of S?)

2-D Rotation



$$x' = x\,\mathbf{cos}(\theta) - y\,\mathbf{sin}(\theta)$$
$$y' = x\,\mathbf{sin}(\theta) + y\,\mathbf{cos}(\theta)$$

2-D Rotation



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

Trig Identity...

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

Substitute...

$x' = x\, \mathbf{cos}(\theta) - y\, \mathbf{sin}(\theta)$
$y' = x\, \mathbf{sin}(\theta) + y\, \mathbf{cos}(\theta)$

2-D Rotation

This is easy to capture in matrix form:

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

Even though sin(θ) and cos(θ) are nonlinear functions of θ,

- ***x' is a linear combination of x and y***
- ***y' is a linear combination of x and y***

What is the inverse transformation?

- Rotation by –θ
- For rotation matrices, det(R) = 1 so $\mathbf{R}^{-1} = \mathbf{R}^{T}$

2x2 Matrices

Which types of transformations can be represented by 2x2 matrices?

2D Identity:

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

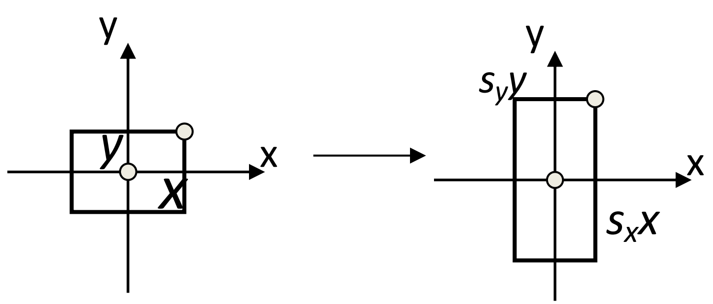
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D Scale around (0,0):

$$x' = s_x \cdot x$$

$$y' = s_y \cdot y$$

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



2x2 Matrices

Which types of transformations can be represented by 2x2 matrices?

2D Rotate around (0,0):

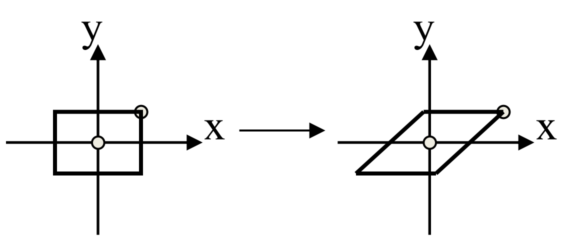
$$x' = \cos\Theta \cdot x - \sin\Theta \cdot y$$
$$y' = \sin\Theta \cdot x + \cos\Theta \cdot y$$

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

2D Shear:

$$x' = x + sh_x \cdot y$$
$$y' = sh_y \cdot x + y$$

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



Shear along x axis

2x2 Matrices

Which types of transformations can be represented by 2x2 matrices?

2D Mirror over the Y axis:

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

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

2D Mirror over (0,0):

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

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

2x2 Matrices

Which types of transformations can be represented by 2x2 matrices?

2D Translation?

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

NO!

2D translation is not a linear transformation from R^2 to R^2!

(e.g. it is not homogenous)

All 2D Linear Transformations

Linear transformations are combinations of scale, rotation and shear.

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

Properties of linear transformations:

- Origin maps to origin
- Lines map to lines
- Parallel lines remain parallel
- Ratios are preserved along lines
- Closed under composition:
- Easy to find inverse when nonsingular

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

Homogeneous Coordinates

Q: How can we represent translation as a 3x3 matrix?

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

A: Add a 3rd coordinate to every 2D point

Homogeneous Coordinates

represent coordinates in 2 dimensions with a 3-vector

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

Scalar multiplication does not change the point:

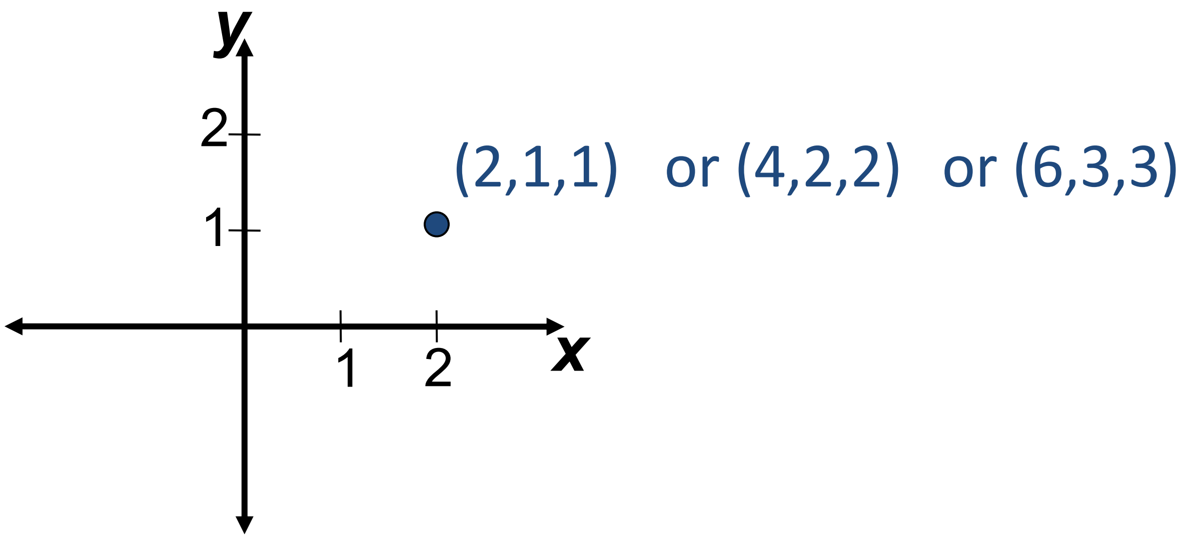
$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \cong \begin{pmatrix} 2x \\ 2y \\ 2 \end{pmatrix} \cong \begin{pmatrix} 3.5x \\ 3.5y \\ 3.5 \end{pmatrix} \cong \begin{pmatrix} \lambda x \\ \lambda y \\ \lambda \end{pmatrix}$$

are all equivalent to $\begin{pmatrix} x \\ y \end{pmatrix}$

Homogeneous Coordinates

Add a 3rd coordinate to every 2D point

- (x, y, w) represents a point at location (x/w, y/w)
- (x, y, 0) represents a point at infinity
- (0, 0, 0) is not allowed



Convent coordinate system to
represent many useful transformations

Homogeneous Coordinates

Q: How can we represent translation as a 3x3 matrix?

$$\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{t}_x$$

$$\boldsymbol{y}' = \boldsymbol{y} + \boldsymbol{t}_y$$

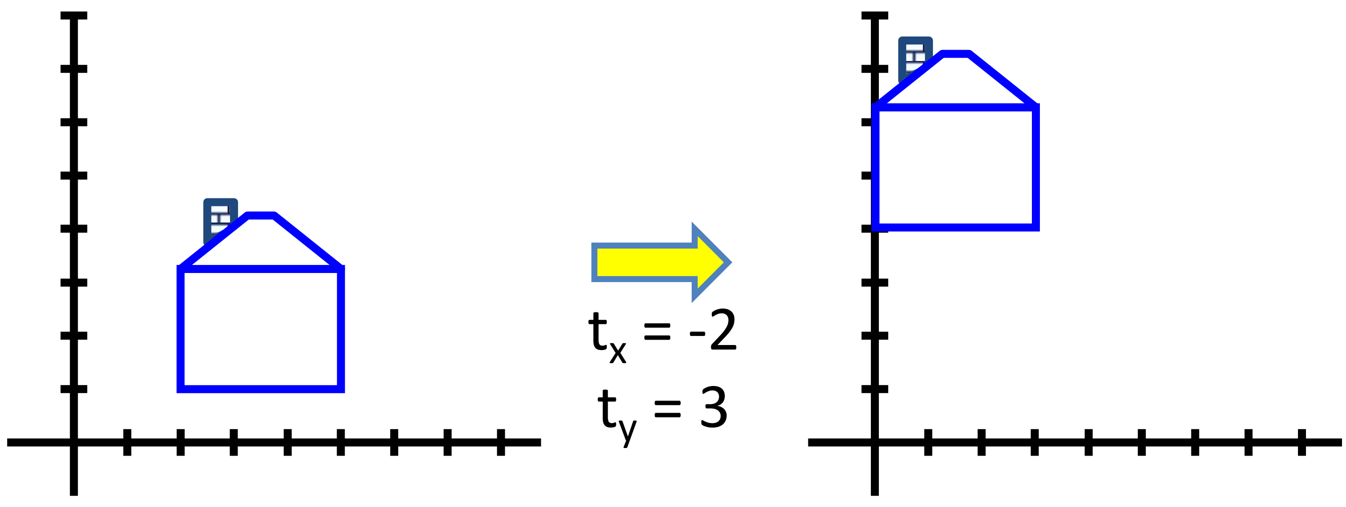
A: Using the rightmost column

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

Translation: Example

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

Homogeneous Coordinates



Basic 2D Transformations

Basic 2D transformations as 3x3 matrices

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

Translate

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

Scale

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

Rotate

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

Shear

Matrix Composition

Transformations can be combined by matrix multiplication:

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

$\mathbf{p}'$ = $T(t_x,t_y)$ $R(Q)$ $S(s_x,s_y)$ $\mathbf{p}$

Advantages:

- General purpose representation
- Hardware implementation

Be aware: the order is important!

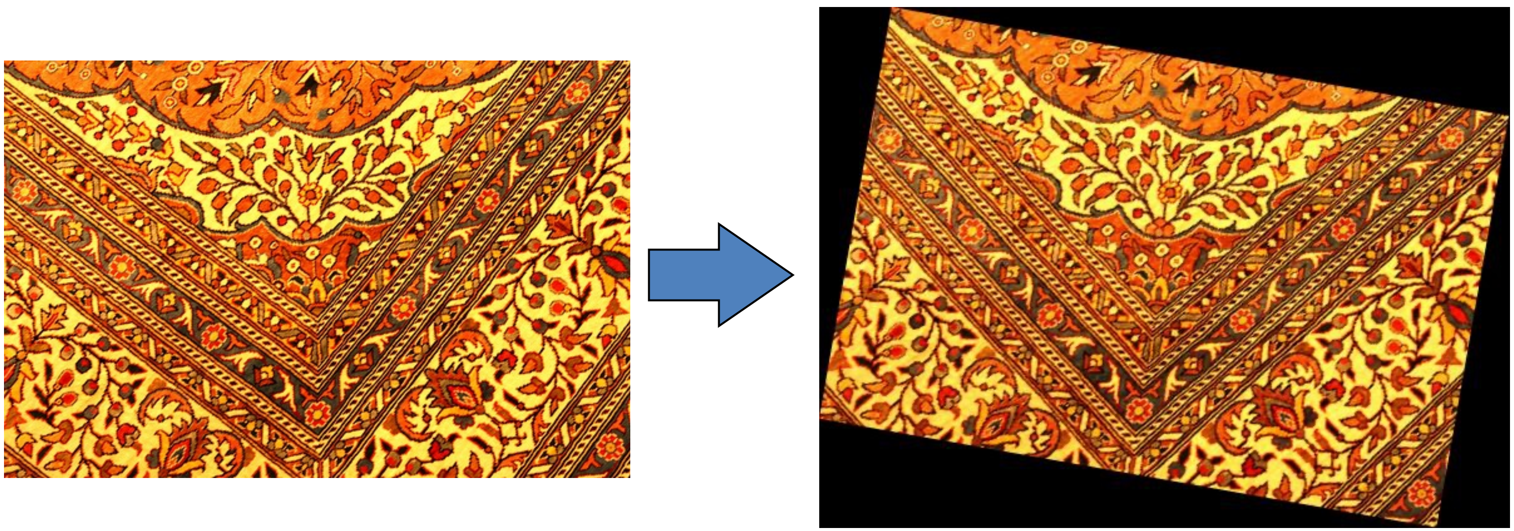
- Matrix multiplication is not commutative

$\mathbf{p}' = T\ R\ S\ \mathbf{p}$

Rigid Transformations

Combination of translation and rotation

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

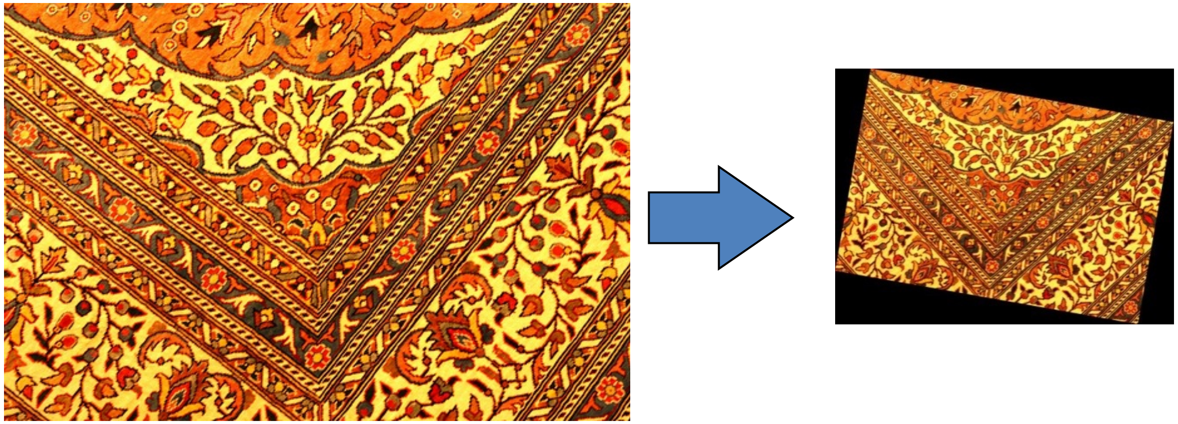


- # Parameters: 3 (t_x, t_y, θ)
- Preserves all information except of orientation (lengths, angles etc.)

Similarity Transformations

Combination of translation, rotation and uniform scaling

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} s \cdot \cos(\theta) & -s \cdot \sin(\theta) & t_x \\ s \cdot \sin(\theta) & s \cdot \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$



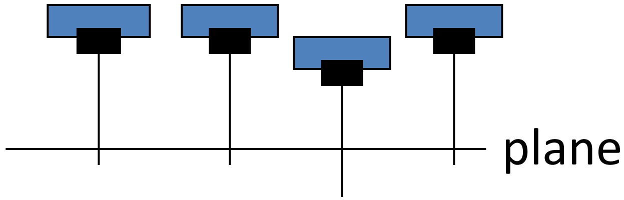
- \# Parameters: 4 (s, t_x, t_y, θ)
- Preserves proportions and angles, but not lengths and orientation

Similarity Transformations

When do we meet them?

- When the camera is scanning a plane in parallel

Camera moving



Change of Coordinates

What is the meaning of transforming one image to the other?

- We change the coordinate system of one to the other
- Translation as an example

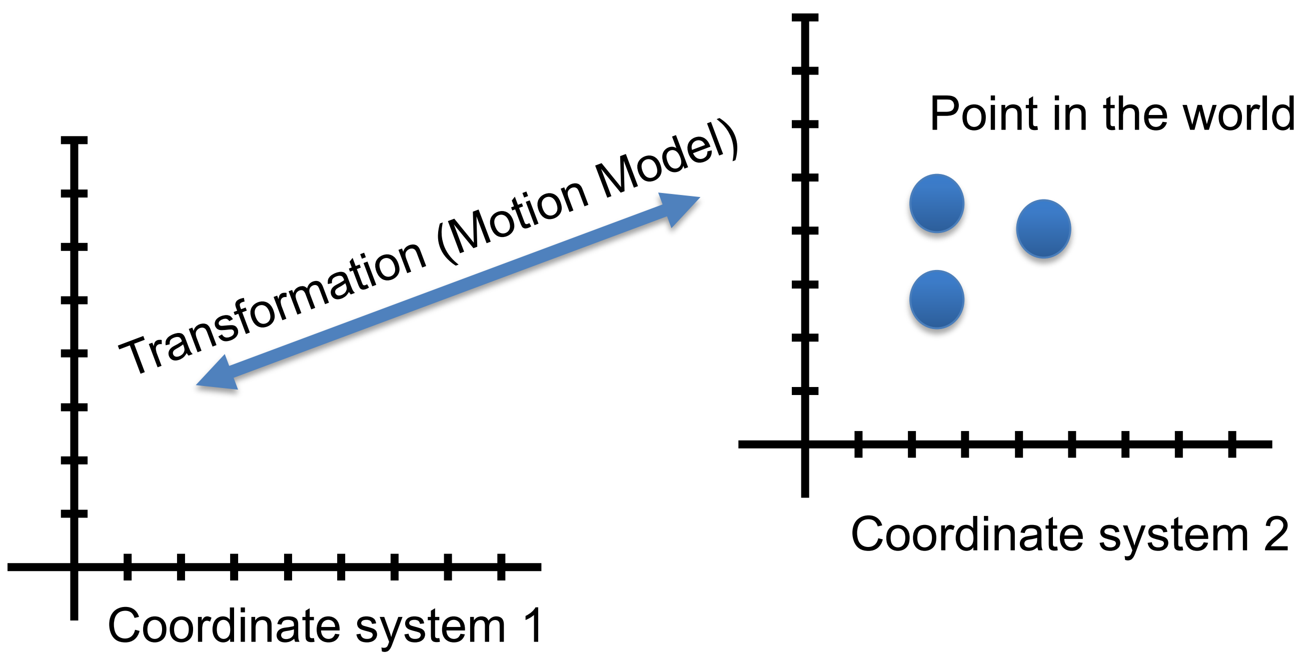
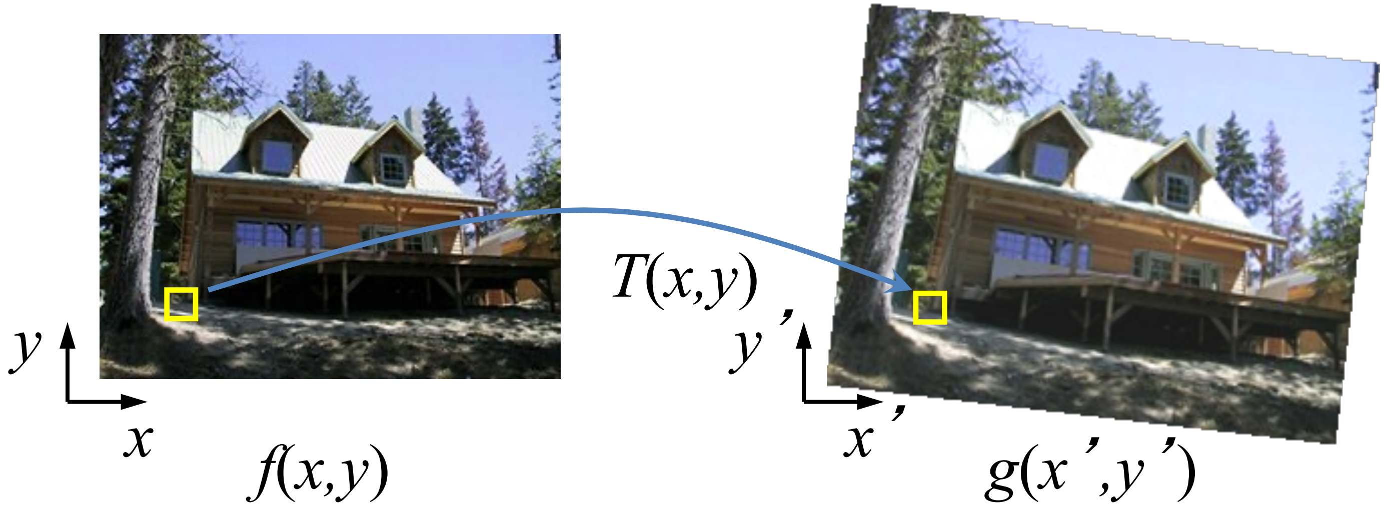


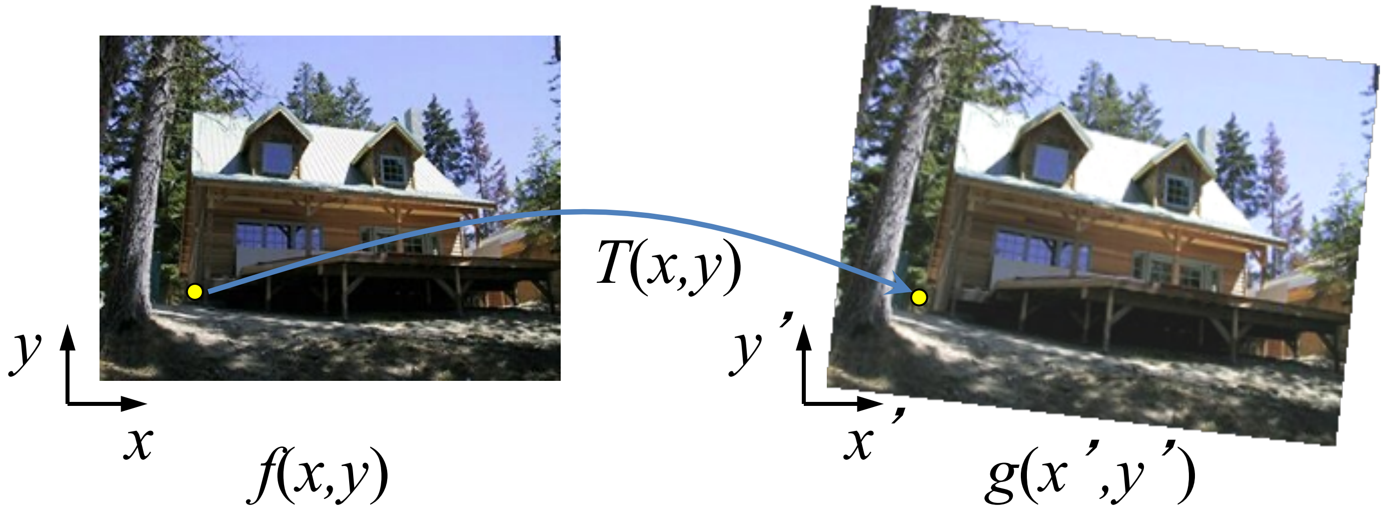
Image warping



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

Important Note: T can be either a function (like we saw before) or any other mapping, i.e. flow field

Forward warping



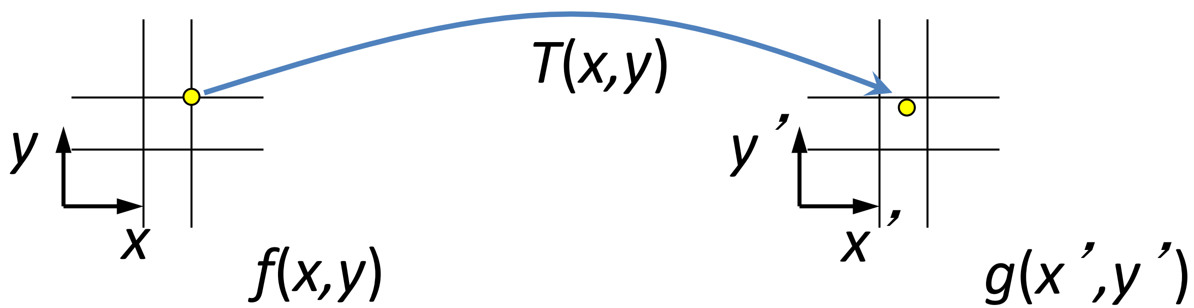
Send each pixel (*x,y*) to its corresponding location

(x',y') = *T*(*x,y*) in the second image

Q: what if pixel lands "between" two pixels?

Q: what about "holes" between pixels?

Forward warping



Send each pixel $f(x,y)$ to its corresponding location

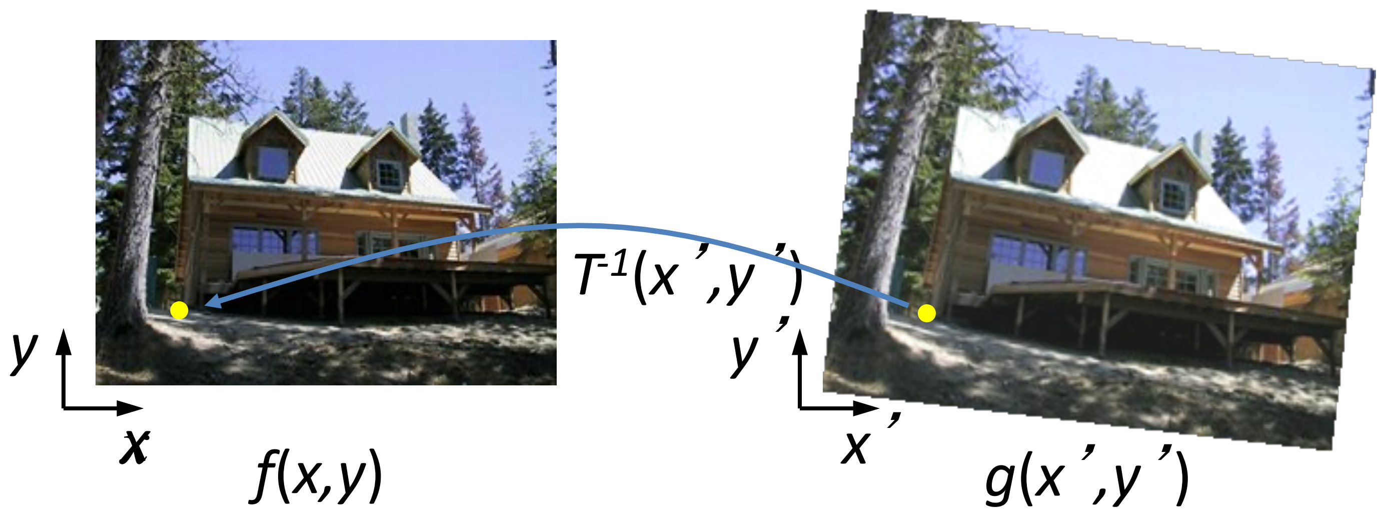
$(x',y') = T(x,y)$ in the second image

Q: what if pixel lands "between" two pixels?

A: distribute its color among the neighboring pixels of (x',y')

– Known as "splatting"

Backward (inverse) warping

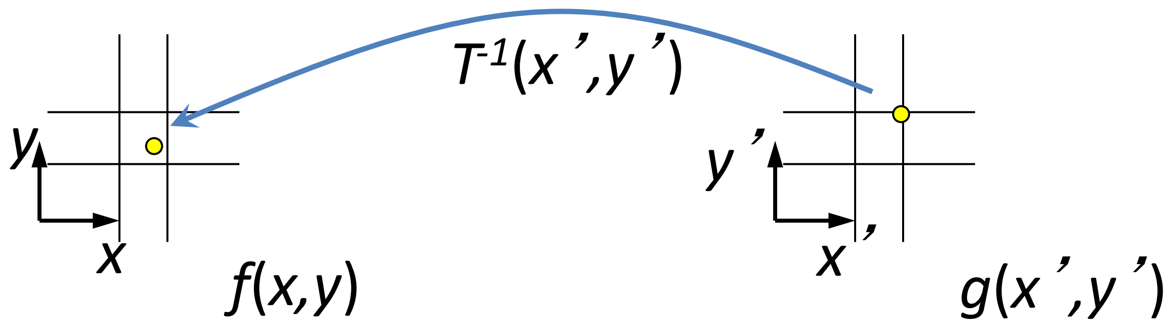


Get each pixel (x',y') from its corresponding location

$(x,y) = T^{-1}(x',y')$ in the first image

Q: what if pixel comes from "between" two pixels?

Inverse warping



Get each pixel $g(x',y')$ from its corresponding location

$(x,y) = T^{-1}(x',y')$ in the first image

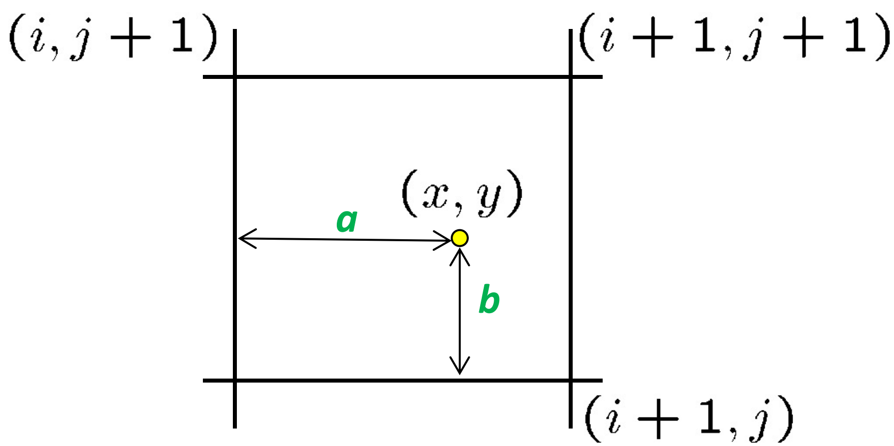
Q: what if pixel comes from "between" two pixels?

A: ***Interpolate*** the color value from its neighbors

- nearest neighbor, bilinear, bicubic, Gaussian

Bilinear interpolation

Sampling at *f(x,y)*:



$$
\begin{aligned}
f(x,y) = \quad & (1-a)(1-b) \quad && f[i,j] \\
& +a(1-b) && f[i+1,j] \\
& +ab && f[i+1,j+1] \\
& +(1-a)b && f[i,j+1]
\end{aligned}
$$

Forward vs. inverse warping

Q: which is better?

A: usually inverse – eliminates holes

- however, it requires an invertible warp function, which isn't always available.

OpenCV:

https://docs.opencv.org/3.4/d1/da0/tutorial_remap.html

https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html