图像拼接

周晓巍

计算摄影学第七讲

Some slides adapted from Noah Snavely, Lingqi Yan

Panorama (全景图)

360度VR

360度VR

全景相机

GoPro odyssey

图像拼接

VR眼镜

Huawei

How to combine two images?

How to combine two images?

What is image warping?
How to compute it?

Part I
Image Warping
图像变形

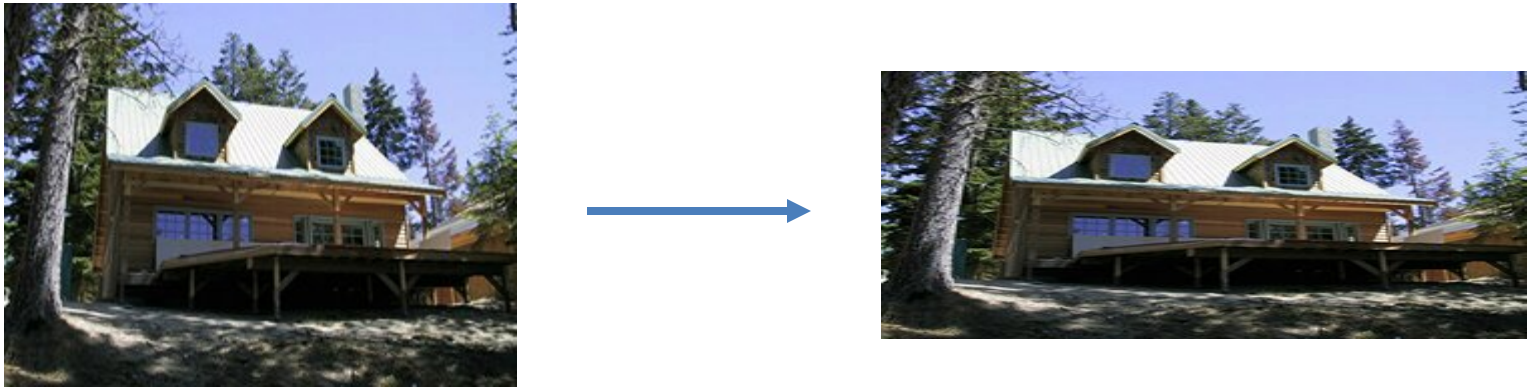
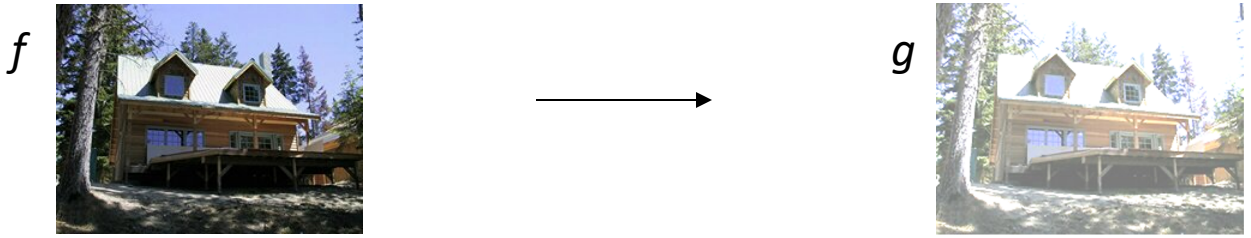
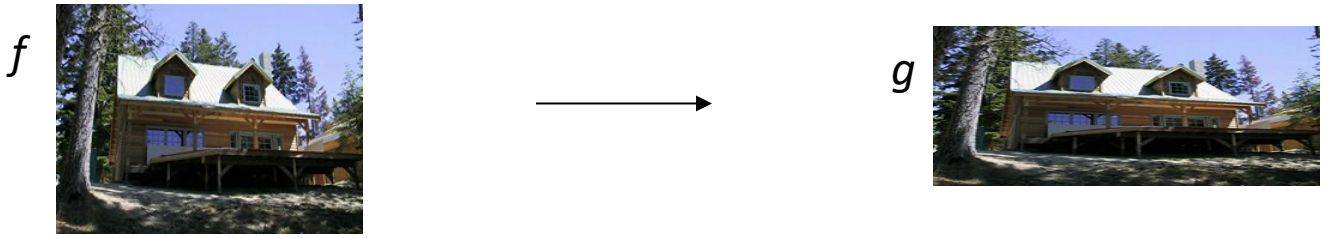


Image Warping

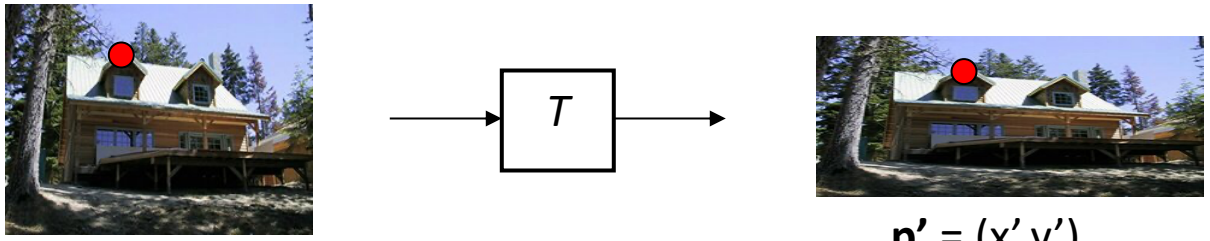
- **image filtering**: change *intensity* of image



- image warping: change *shape* of image



Parametric (global) warping



p = (x,y)

p' = (x',y')

- Transformation T is a coordinate transformation:

$$p' = T(p)$$

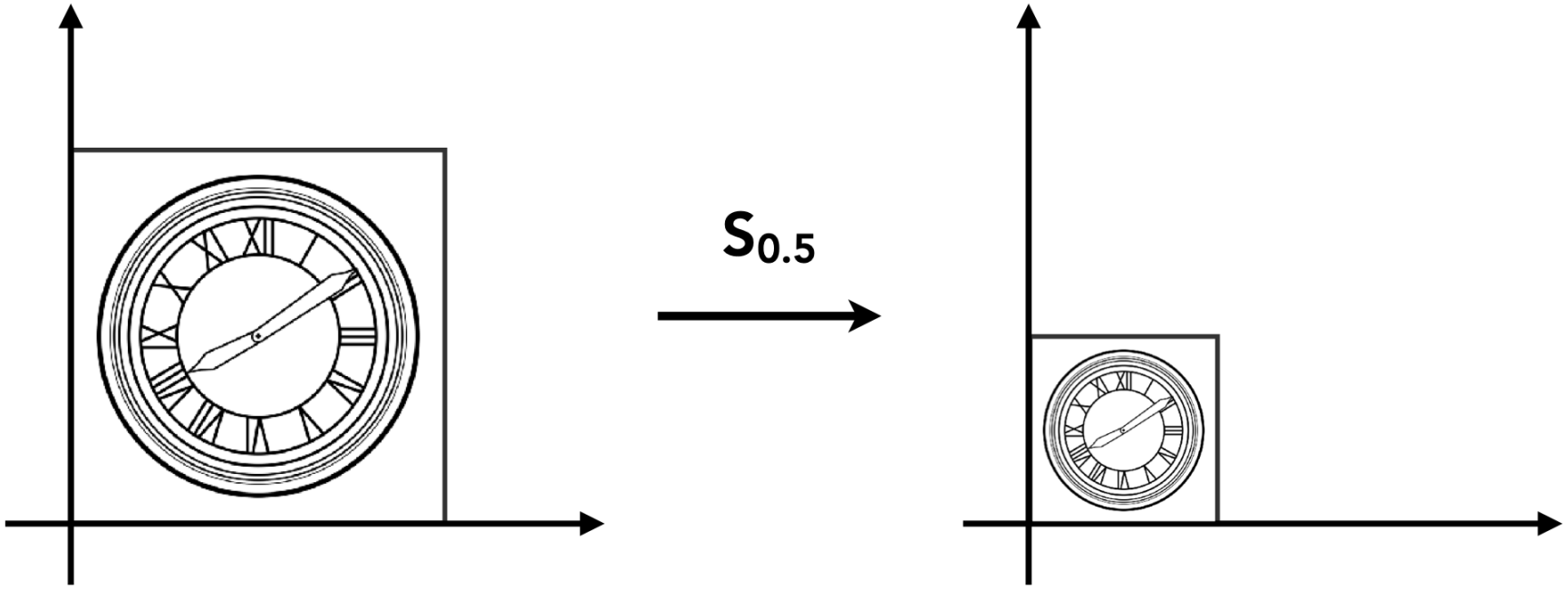
- Examples:

translation

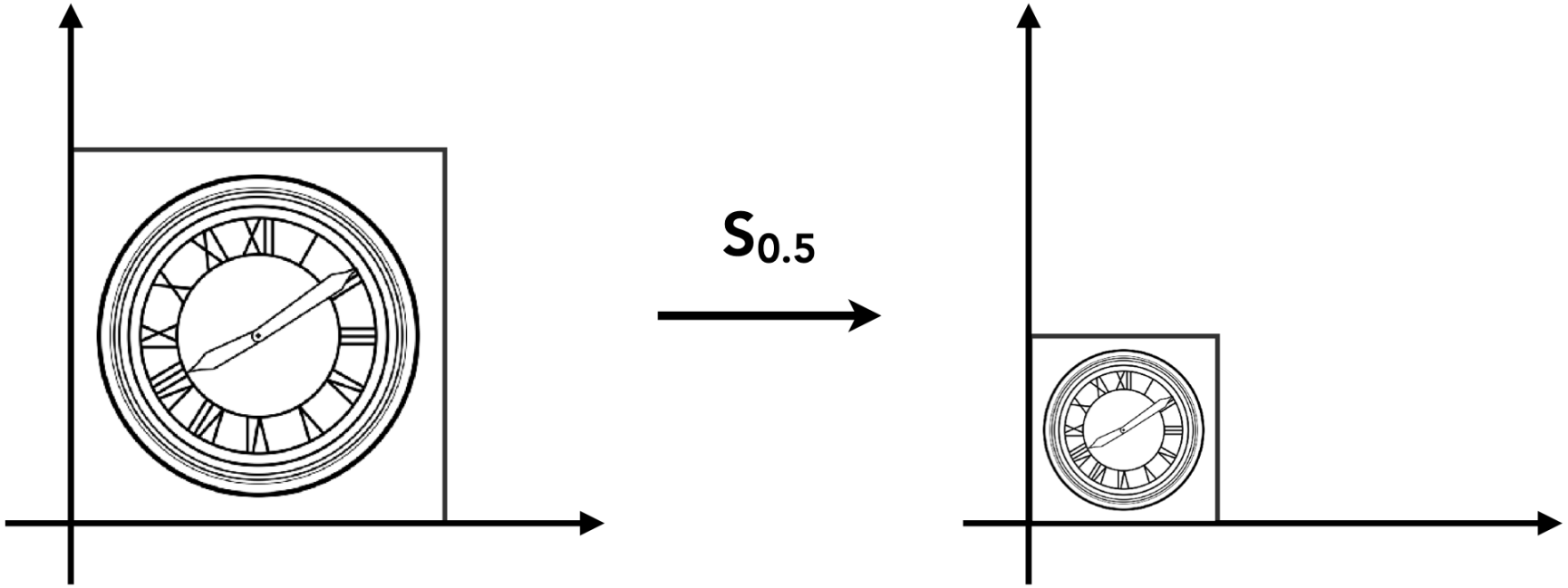
rotation

aspect

Scale



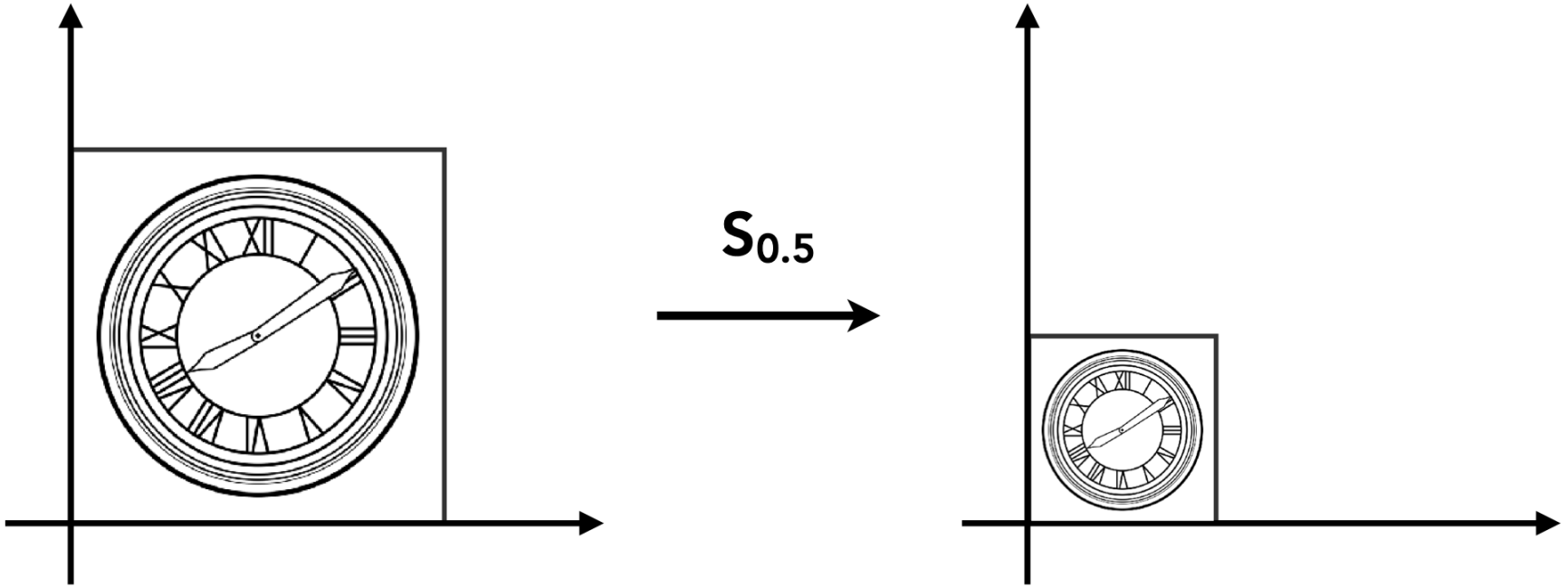
Scale Transform



$$x' = sx$$

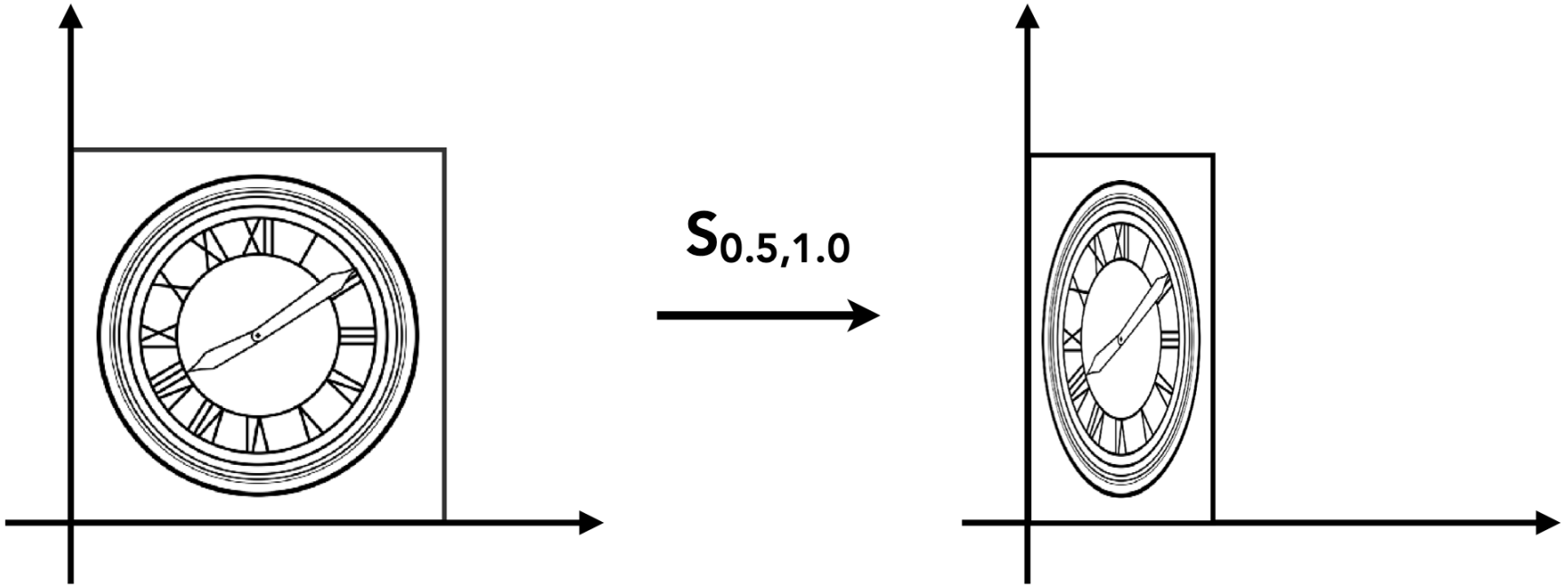
$$y' = sy$$

Scale Matrix



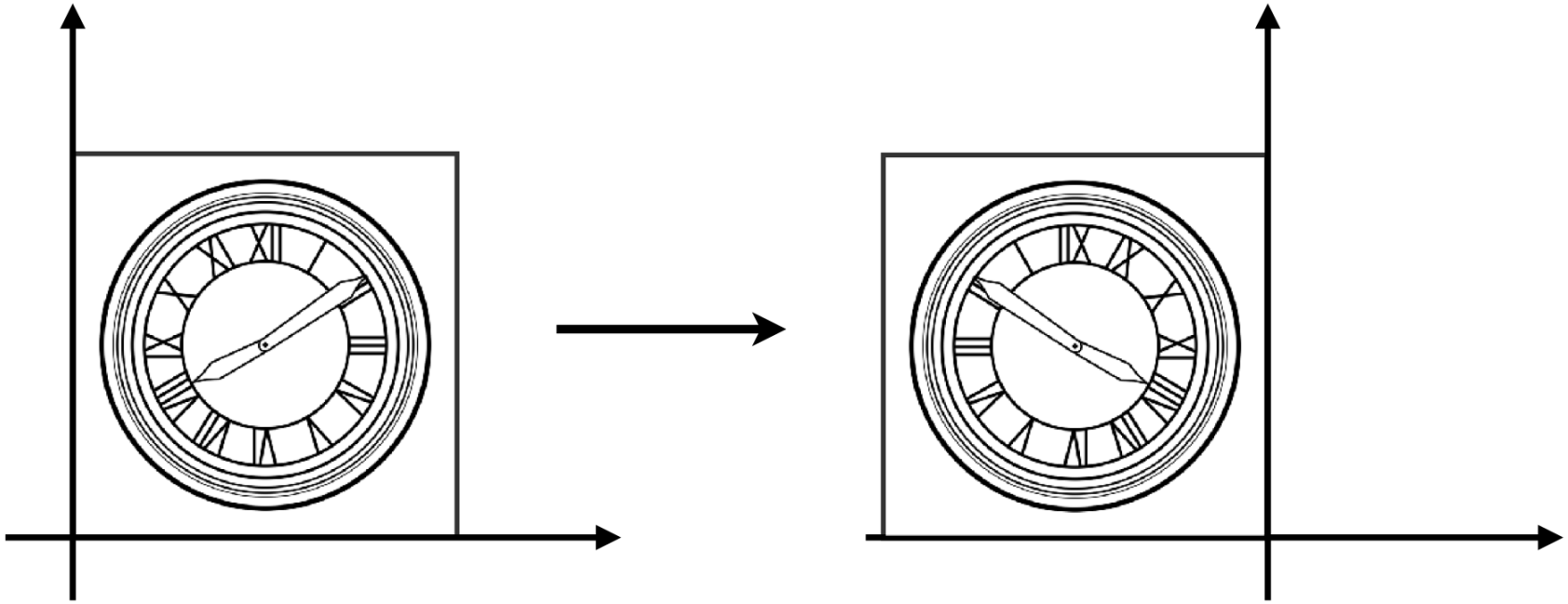
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s & 0 \\ 0 & s \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Scale (Non-Uniform)



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

Reflection Matrix

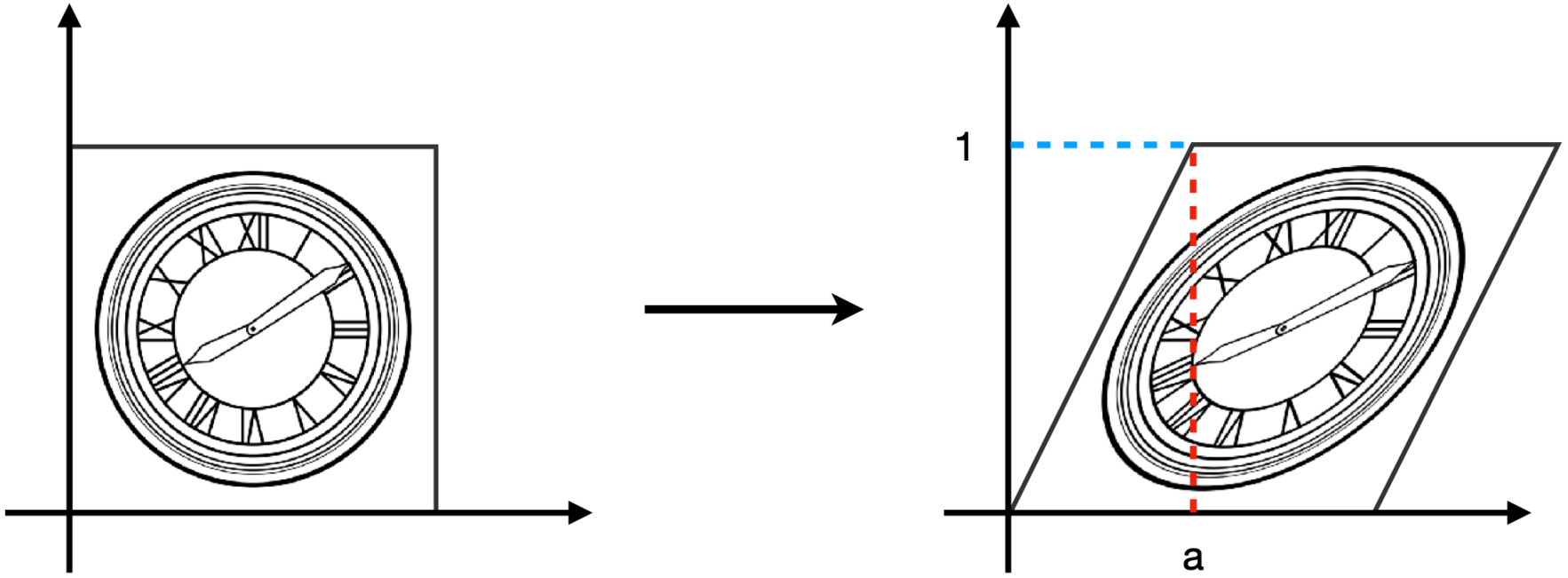


Horizontal reflection:

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

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

Shear Matrix



Hints:

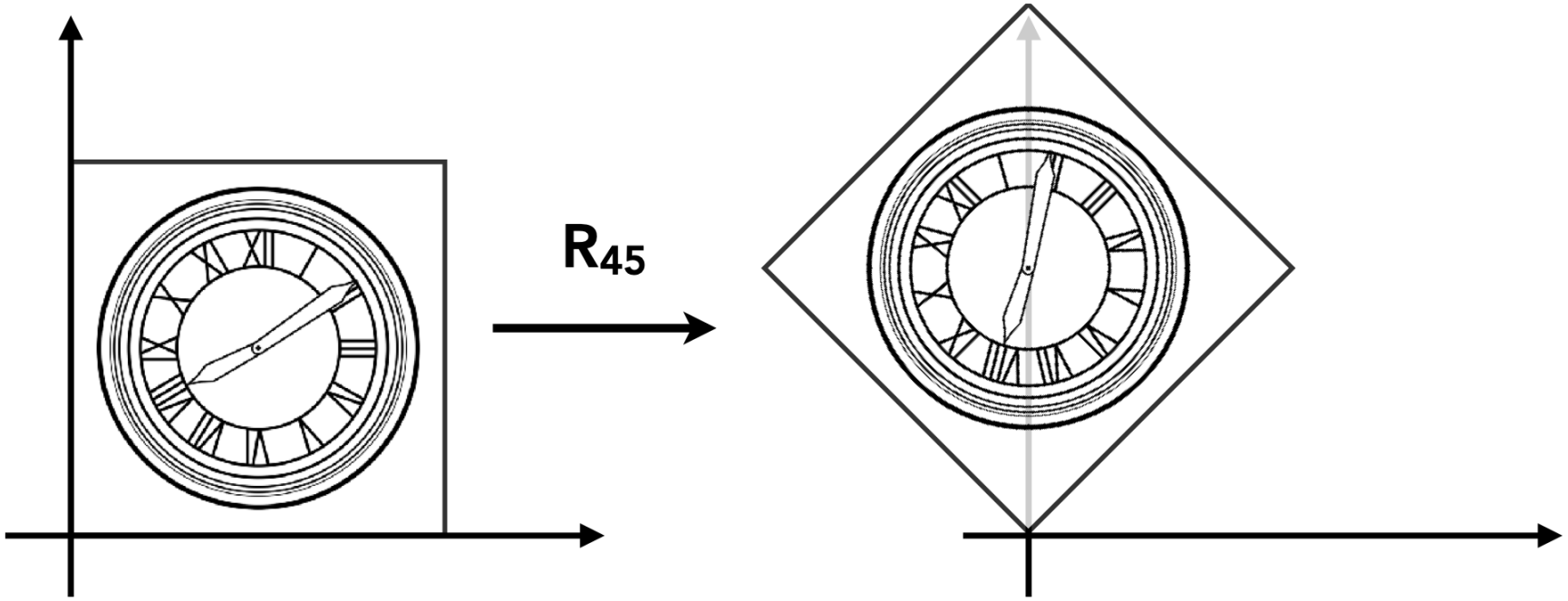
Horizontal shift is 0 at y=0

Horizontal shift is a at y=1

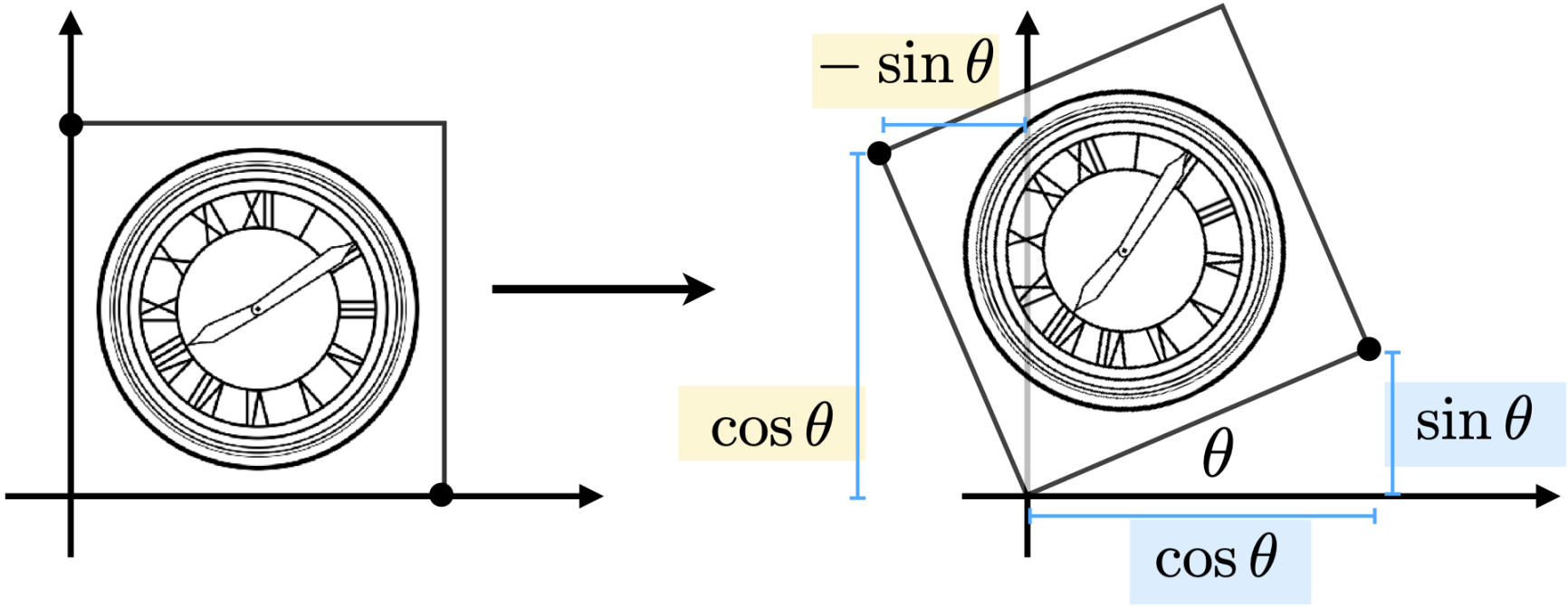
Vertical shift is always 0

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

Rotate (about the origin (0, 0), CCW by default)



Rotation Matrix



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

Linear Transforms = Matrices

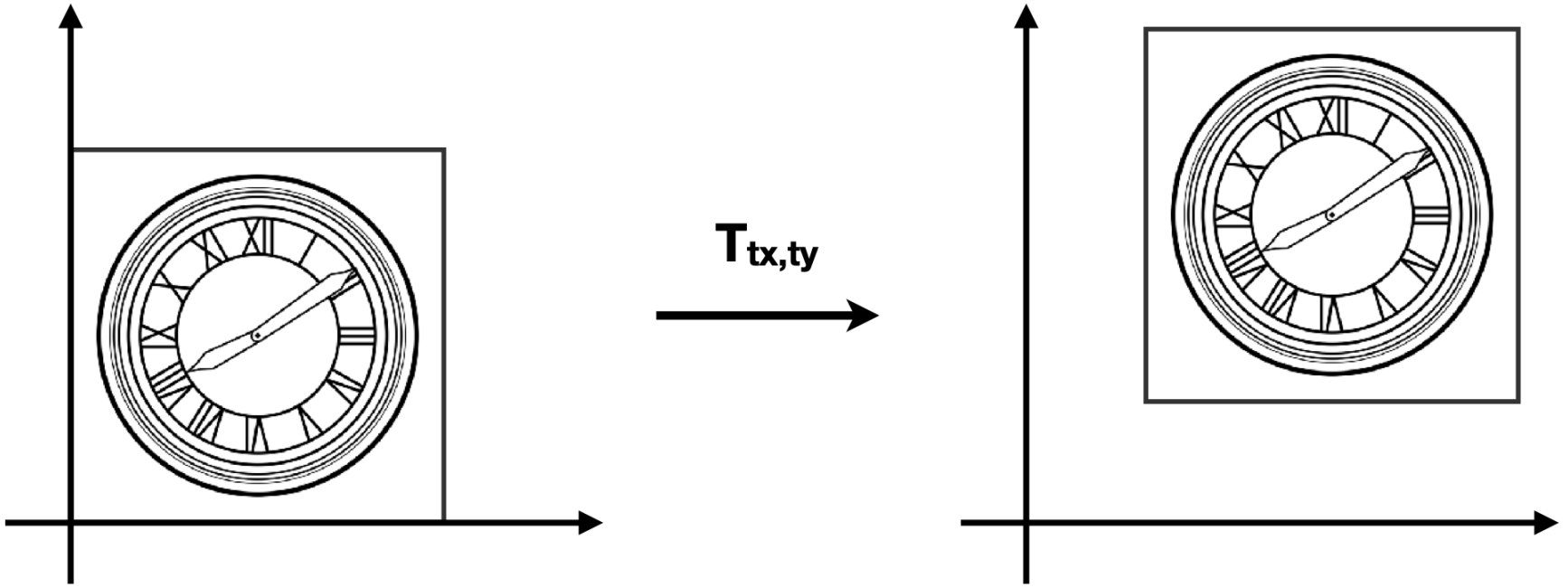
(of the same dimension)

$$x' = a\,x + b\,y$$
$$y' = c\,x + d\,y$$

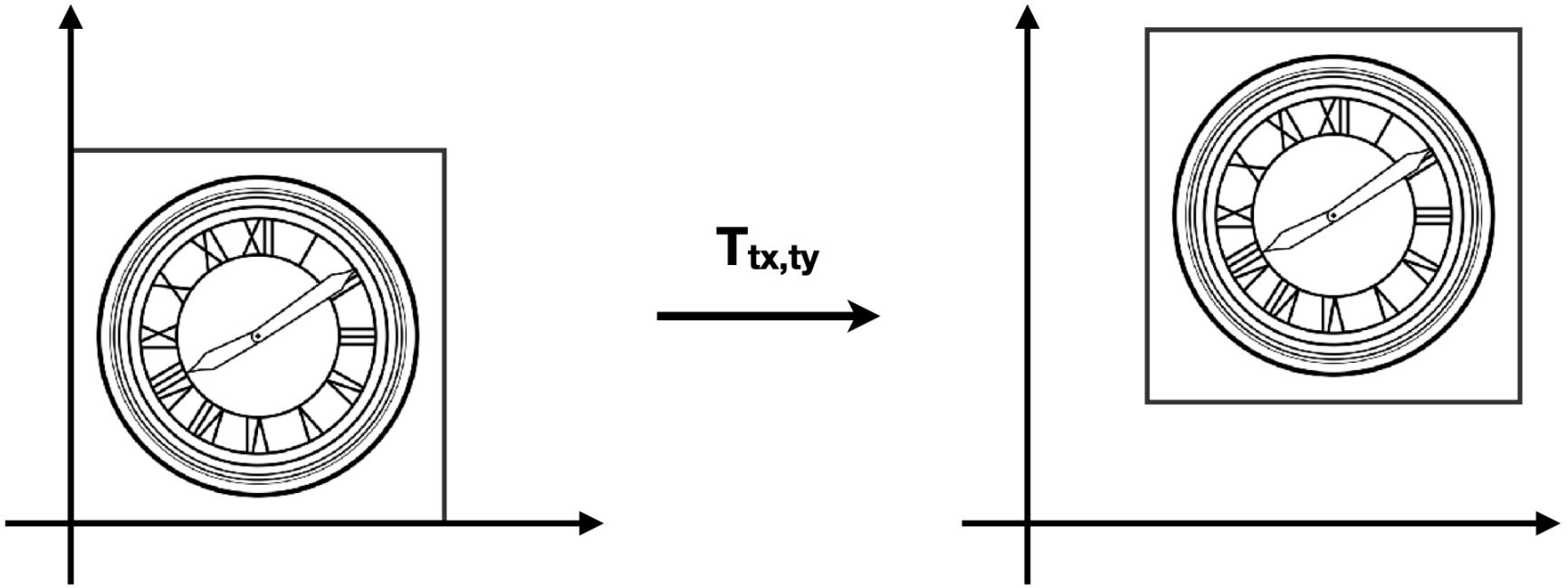
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

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

Translation



Translation??



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

Why Homogeneous Coordinates

- Translation cannot be represented in matrix form

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

(So, translation is NOT linear transform!)

- But we don't want translation to be a special case

- Is there a unified way to represent all transformations? (and what's the cost?)

Solution: Homogenous Coordinates

Add a third coordinate (*w-coordinate*)

- 2D point = (x, y, 1)^T
- 2D vector = (x, y, 0)^T

Matrix representation of translations

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

Homogenous Coordinates

Valid operation if w-coordinate of result is 1 or 0

- **vector + vector = vector**
- **point – point = vector**
- **point + vector = point**
- **point + point = ??**

In homogeneous coordinates,

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} \text{ is the 2D point } \begin{pmatrix} x/w \\ y/w \\ 1 \end{pmatrix}, \; w \neq 0$$

2D Transformations

Scale

$$\mathbf{S}(s_x, s_y) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Rotation

$$\mathbf{R}(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Translation

$$\mathbf{T}(t_x, t_y) = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

Affine Transformations

Affine map = linear map + translation

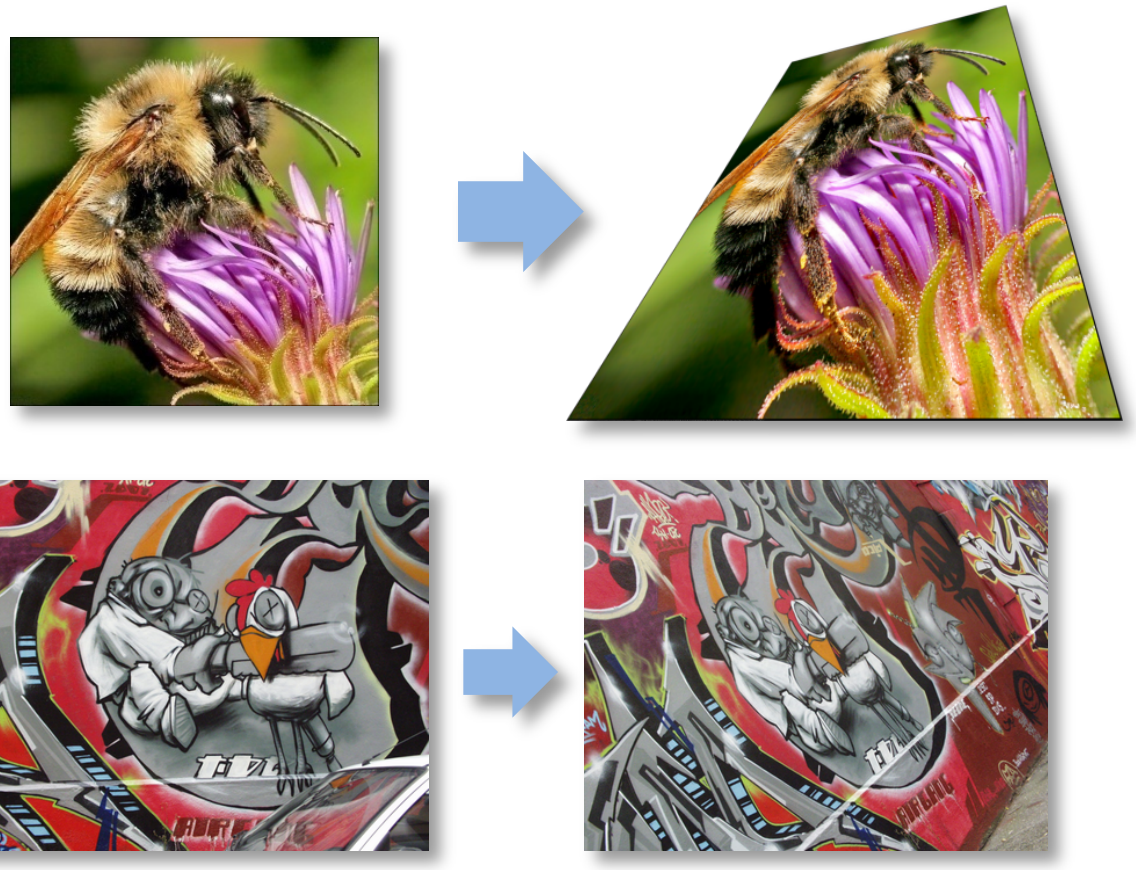
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Using homogenous coordinates:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

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

Projective Transformation (Homography)



In what case is the transformation projective?

Projective Transformation (Homography)

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

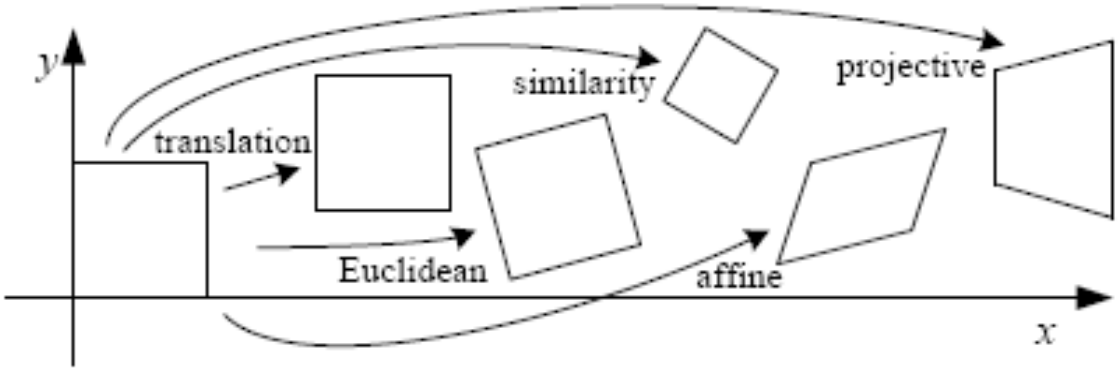
Maybe nonzero

$$x_i' = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

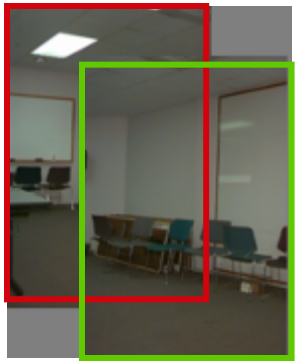
$$y_i' = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

- We usually constrain the length of the vector [h_{00} h_{01} … h_{22}] to be 1, which means the degree of freedom is 8

Summary of 2D transformations

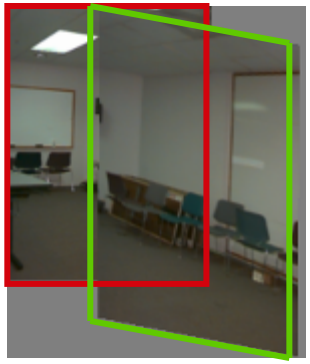


Translation



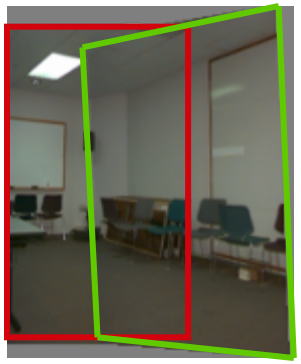
2 unknowns

Affine



6 unknowns

Projective

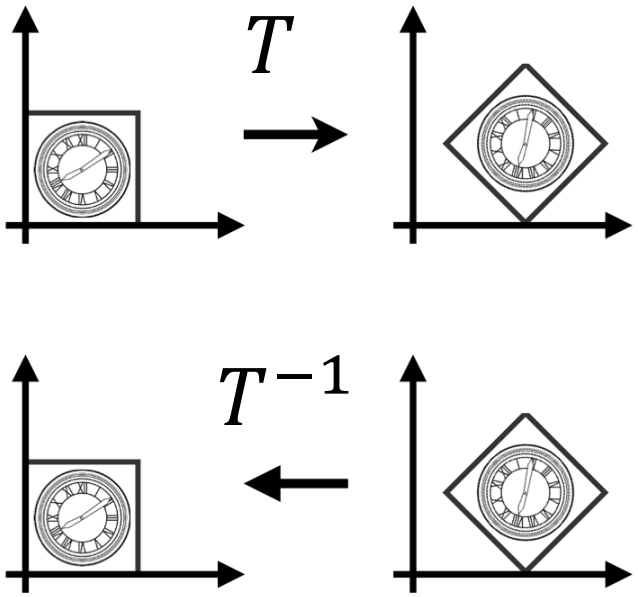


8 unknowns

Inverse Transform

$$T^{-1}$$

T^{-1} is the inverse of transform T in both a matrix and geometric sense



Implementing image warping

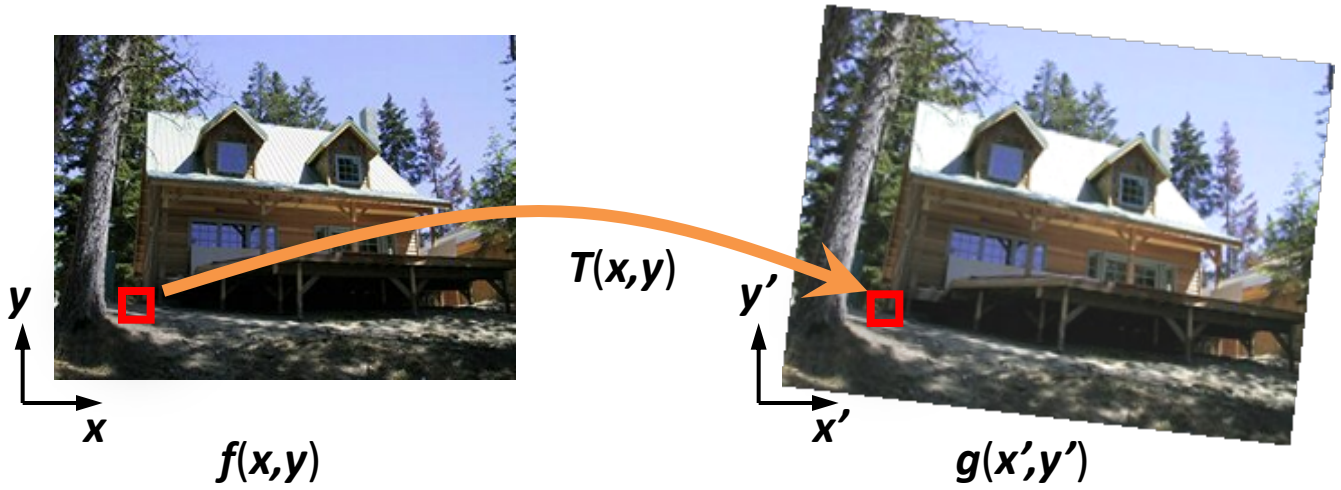
- Given a coordinate transform $(x',y') = T(x,y)$ and a source image $f(x,y)$, how do we compute an transformed image $g(x',y') = f(T(x,y))$?

$f(x,y)$

$g(x',y')$

Forward Warping

- Send each pixel $f(x)$ to its corresponding location $(x',y') = T(x,y)$ in $g(x',y')$

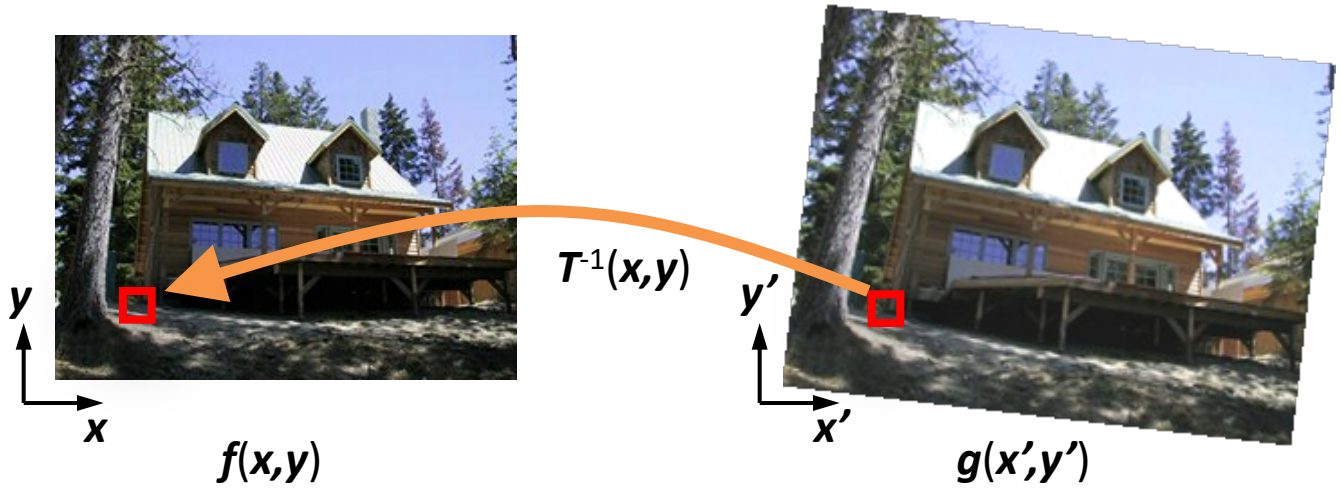


Forward Warping

- What if pixel lands “between” pixels?

Inverse Warping

- Get each pixel $g(x',y')$ from its corresponding location $(x,y) = T^{-1}(x,y)$ in $f(x,y)$



Inverse Warping

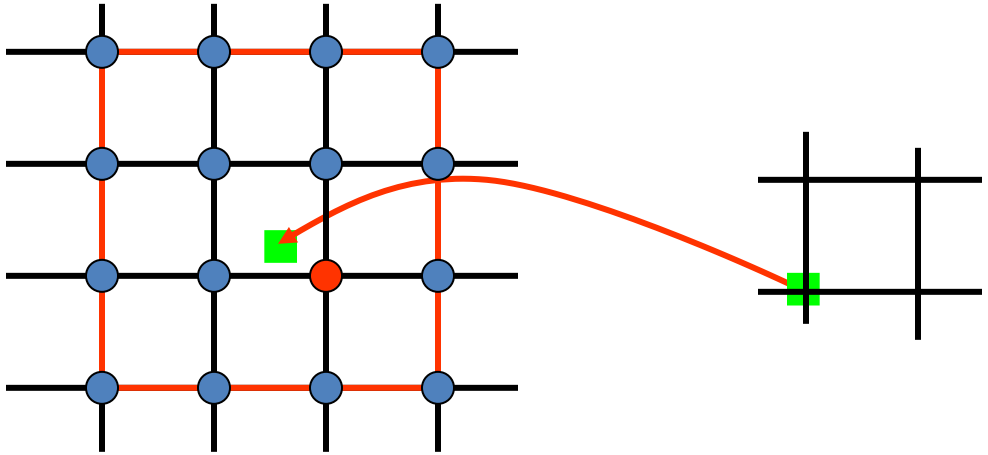
- What if pixel lands “between” pixels?

Answer: interpolate color values from neighboring pixels

Interpolation

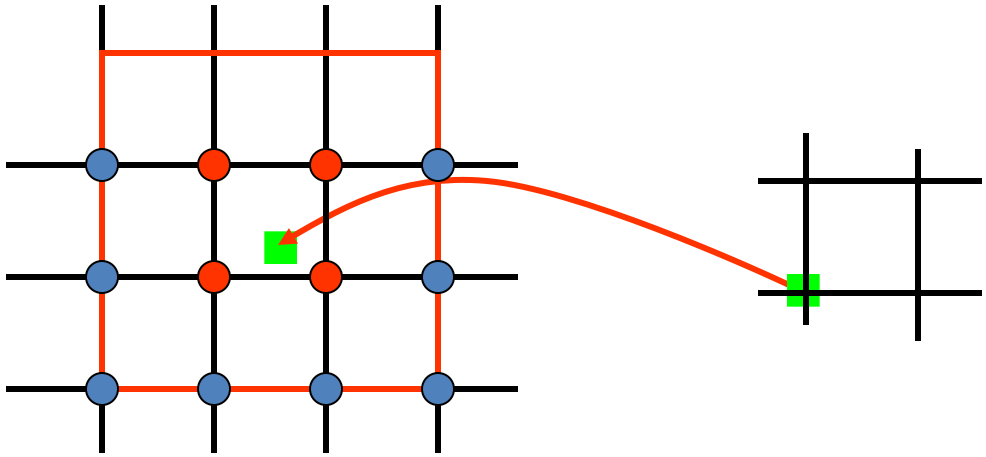
Nearest neighbor

- Copies the color of the pixel with the closest integer coordinate



Interpolation

Weighted sum of four neighboring pixels



Interpolation

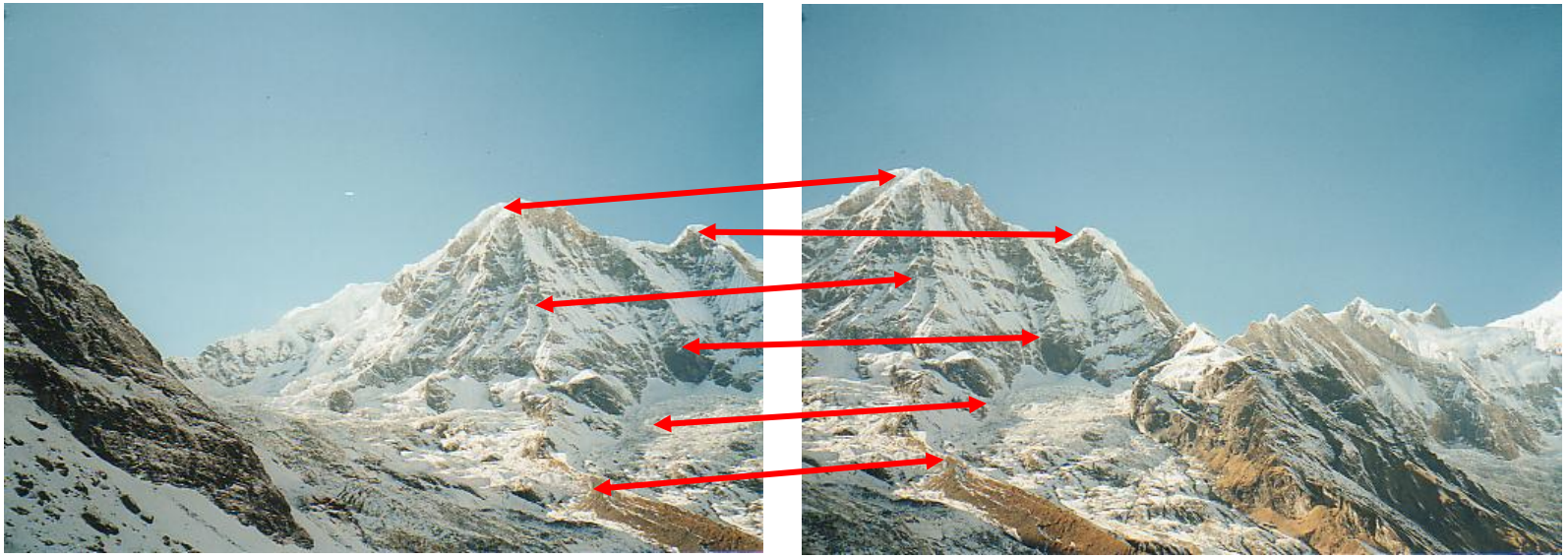
- Possible interpolation filters:
  - nearest neighbor
  - bilinear
  - bicubic
  - sinc

Questions?

Part II
Image Stitching

How to compute transformation?

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \cong T \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



1. Image matching (each match gives an equation)
2. Solve T from the obtained matches

Affine transformations

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} ax + by + c \\ dx + ey + f \\ 1 \end{bmatrix}$$

- How many unknowns?
- How many equations per match?
- How many matches do we need?

Affine transformations

- For each match, we have

$$x' = ax + by + c$$
$$y' = dx + ey + f$$

- Matrix form

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

Affine transformations

- For n matches

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ & & \vdots & & & \\ x_n & y_n & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ \vdots \\ x'_n \\ y'_n \end{bmatrix}$$

$$\underset{2n \times 6}{\mathbf{A}} \qquad \underset{6 \times 1}{\mathbf{t}} = \underset{2n \times 1}{\mathbf{b}}$$

How to solve t?

- Least squares: find **t** that minimizes

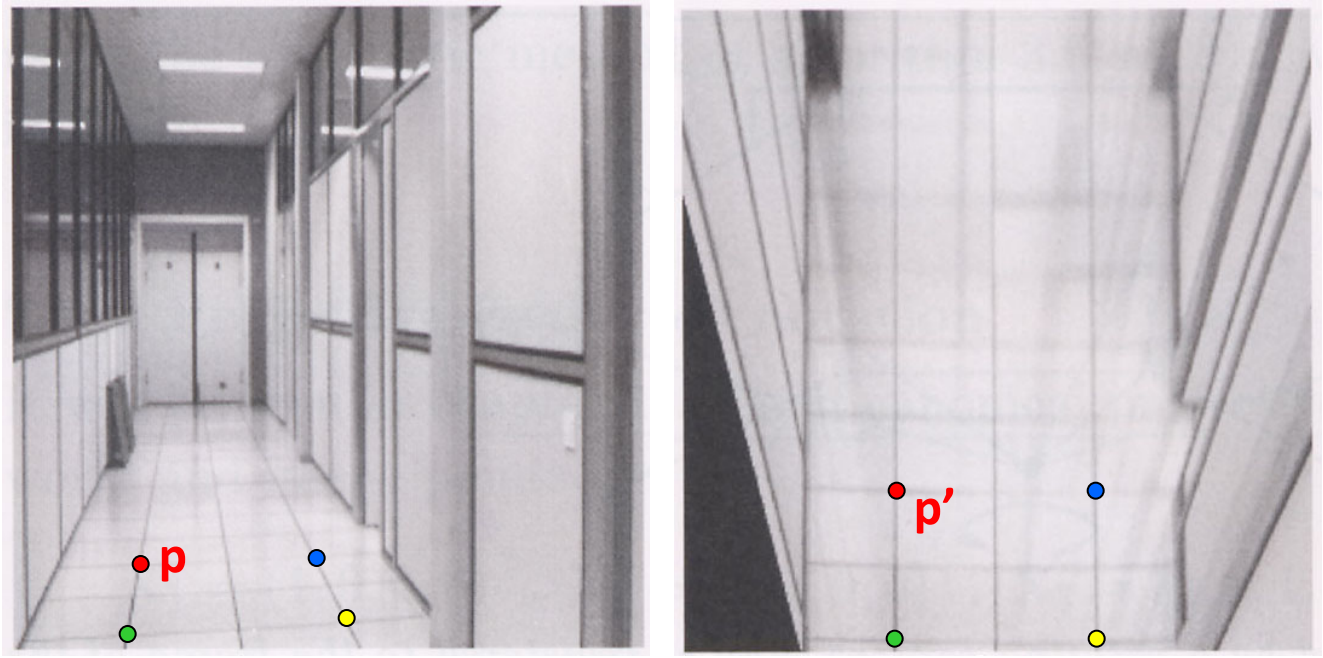
$$||\mathbf{A}\mathbf{t} - \mathbf{b}||^2$$

- To solve, form the *normal equations*

$$\mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{t} = \mathbf{A}^{\mathrm{T}}\mathbf{b}$$

$$\mathbf{t} = \left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{-1}\mathbf{A}^{\mathrm{T}}\mathbf{b}$$

Projective transformations



P’ Homography P

$$\begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix} \sim H_{10} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

Solving for homographies

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

$$x_i' = \frac{h_{00}x_i + h_{01}y_i + h_{02}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

$$y_i' = \frac{h_{10}x_i + h_{11}y_i + h_{12}}{h_{20}x_i + h_{21}y_i + h_{22}}$$

$$\begin{aligned} x_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\ y_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12} \end{aligned}$$

Solving for homographies

$$
\begin{aligned}
x_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{00}x_i + h_{01}y_i + h_{02} \\
y_i'(h_{20}x_i + h_{21}y_i + h_{22}) &= h_{10}x_i + h_{11}y_i + h_{12}
\end{aligned}
$$

$$
\begin{bmatrix}
x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \\
0 & 0 & 0 & x_i & y_i & 1 & -y_i'x_i & -y_i'y_i & -y_i'
\end{bmatrix}
\begin{bmatrix}
h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22}
\end{bmatrix}
=
\begin{bmatrix}
0 \\ 0
\end{bmatrix}
$$

Solving for homographies

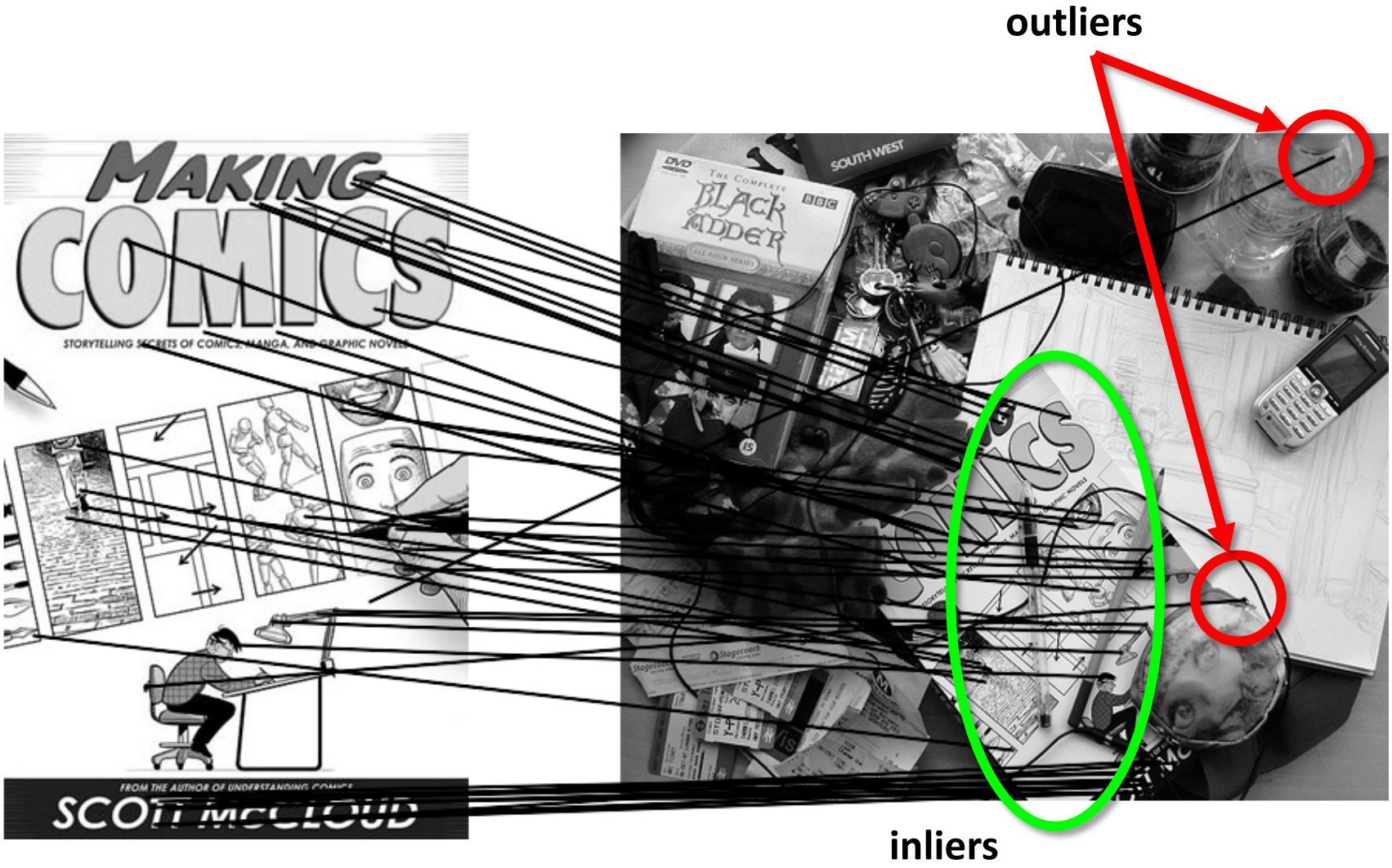
$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1x_1 & -y'_1y_1 & -y'_1 \\ & & & & \vdots & & & & \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_nx_n & -x'_ny_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_nx_n & -y'_ny_n & -y'_n \end{bmatrix} \begin{bmatrix} h_{00} \\ h_{01} \\ h_{02} \\ h_{10} \\ h_{11} \\ h_{12} \\ h_{20} \\ h_{21} \\ h_{22} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

$$\underset{2n \times 9}{\mathbf{A}} \qquad \underset{9}{\mathbf{h}} \qquad \underset{2n}{\mathbf{0}}$$

Defines a least squares problem: minimize $\|\mathbf{A}\mathbf{h} - \mathbf{0}\|^2$

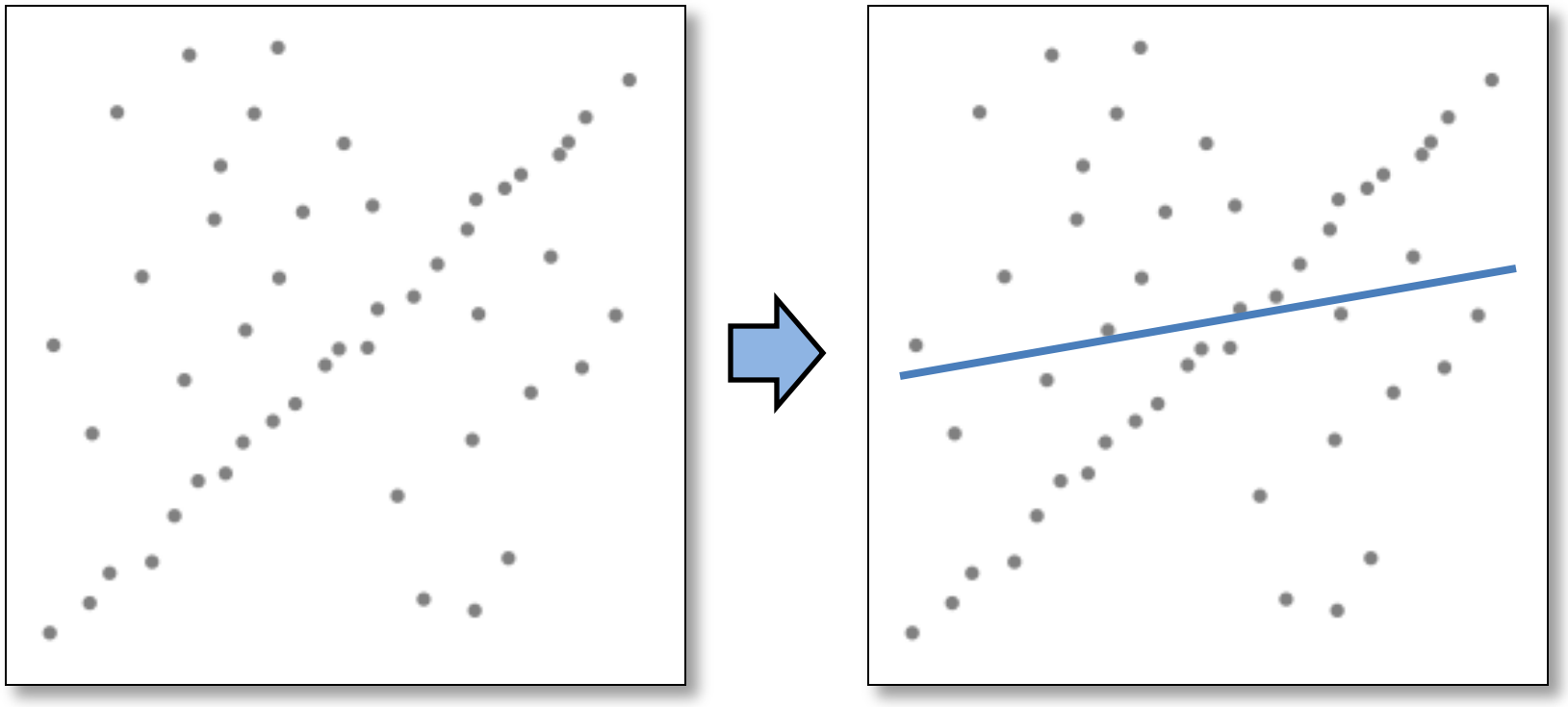
- Since $\mathbf{h}$ is only defined up to scale, solve for unit vector $\hat{\mathbf{h}}$
- Solution: $\hat{\mathbf{h}}$ = eigenvector of $\mathbf{A}^T\mathbf{A}$ with smallest eigenvalue
- Works with 4 or more points

Outliers


outliers
MAKING COMICS
STORYTELLING SECRETS OF COMICS, MANGA, AND GRAPHIC NOVELS
FROM THE AUTHOR OF UNDERSTANDING COMICS
SCOTT McCLOUD
THE COMPLETE BLACK ADDER
BBC
SOUTH WEST
inliers

Robustness

- Let's consider a simpler example… linear regression

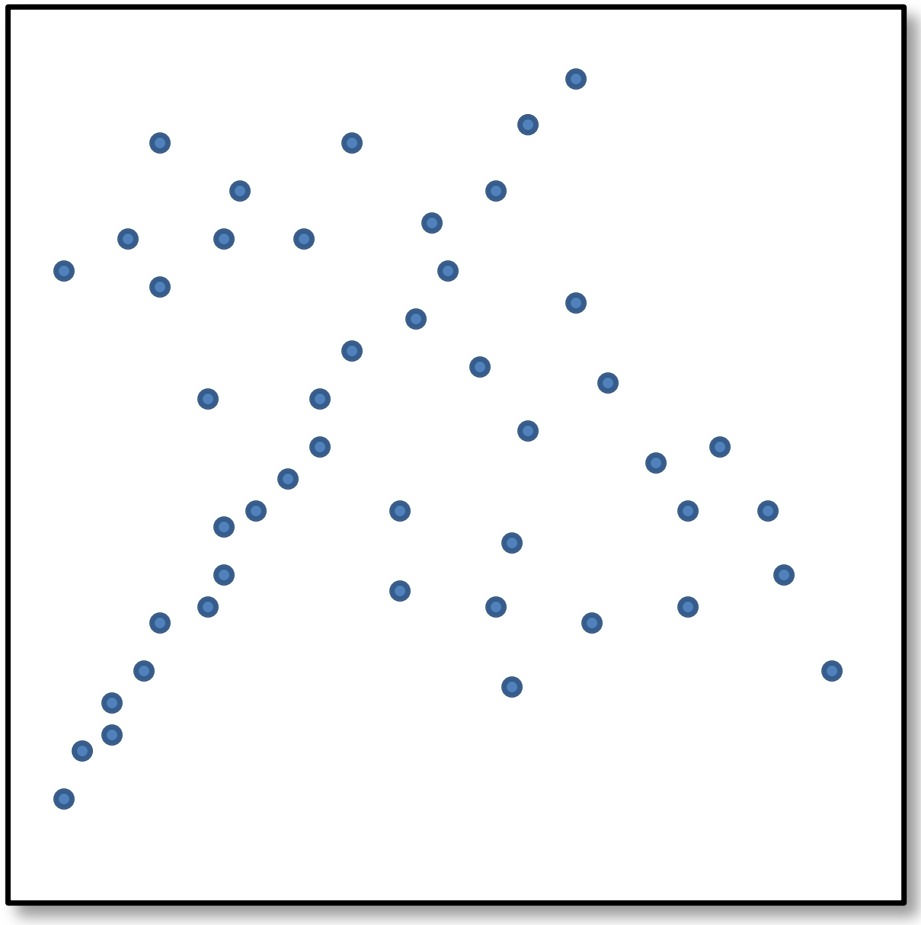


Problem: Fit a line to these datapoints

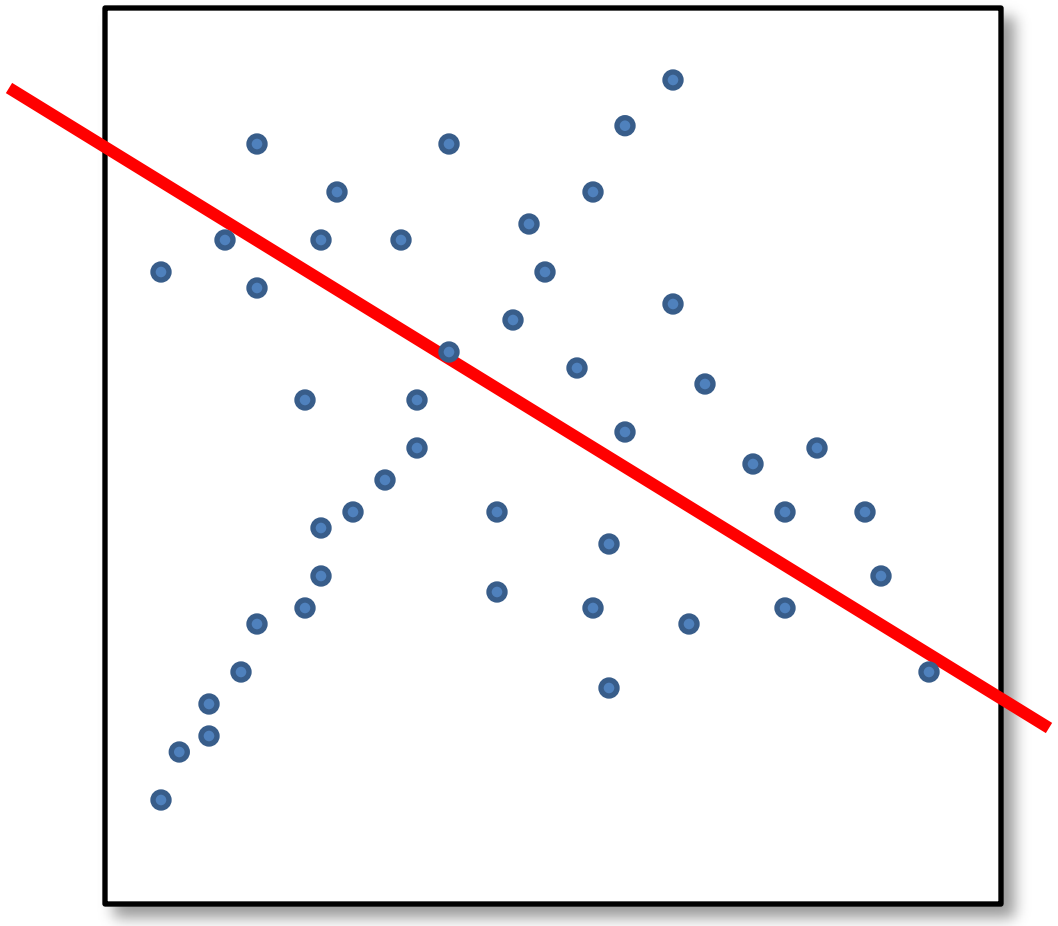
Least squares fit

- How can we fix this?

RANSAC

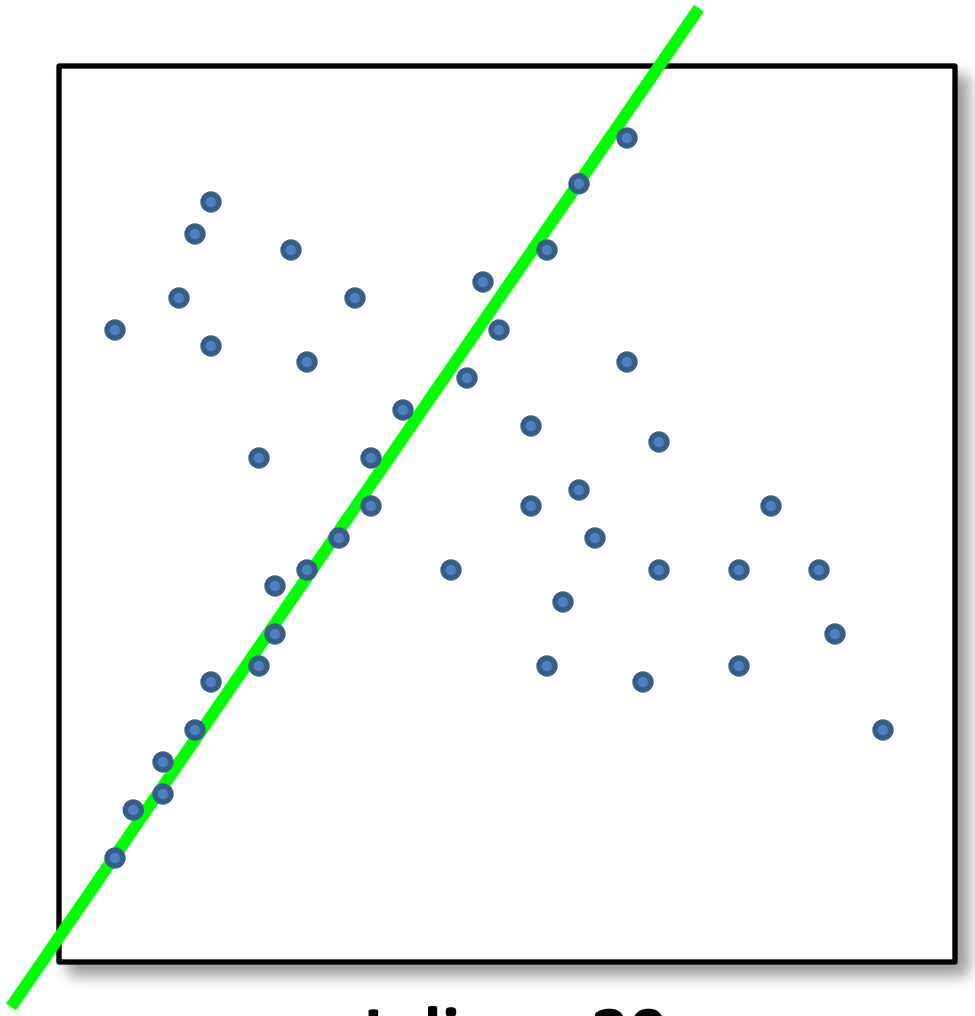


RANSAC



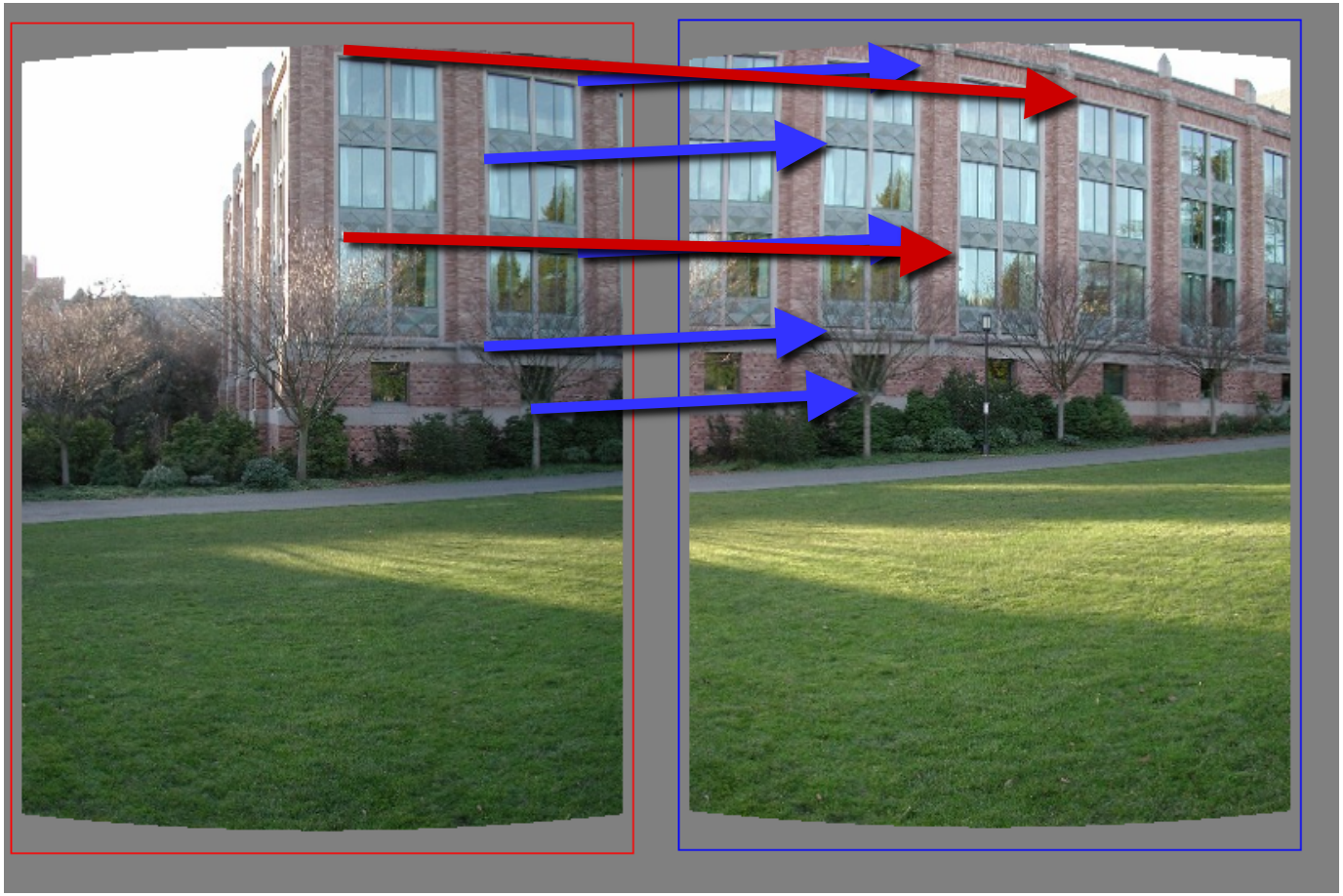
Inliers: 3

RANSAC

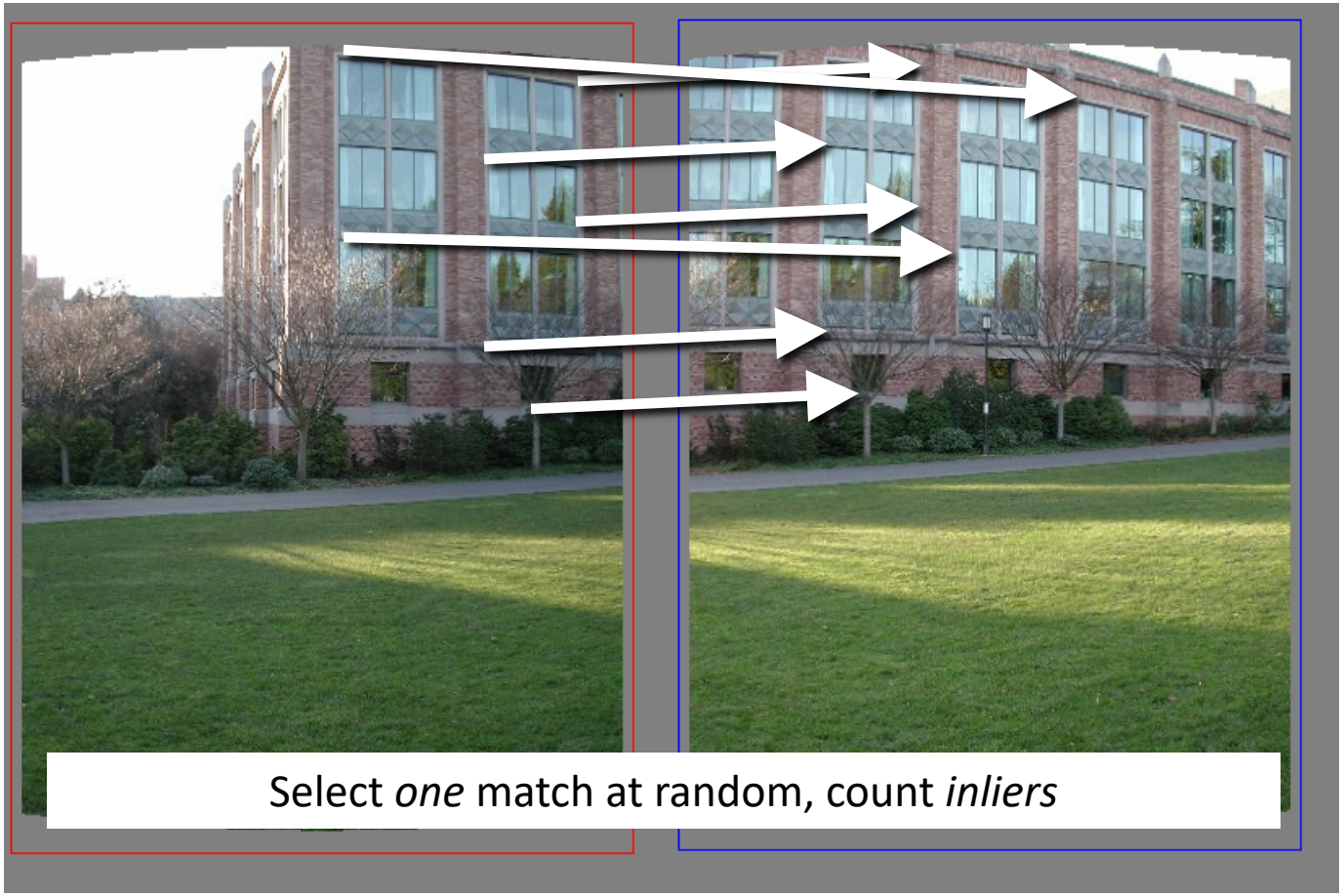


Inliers: 20

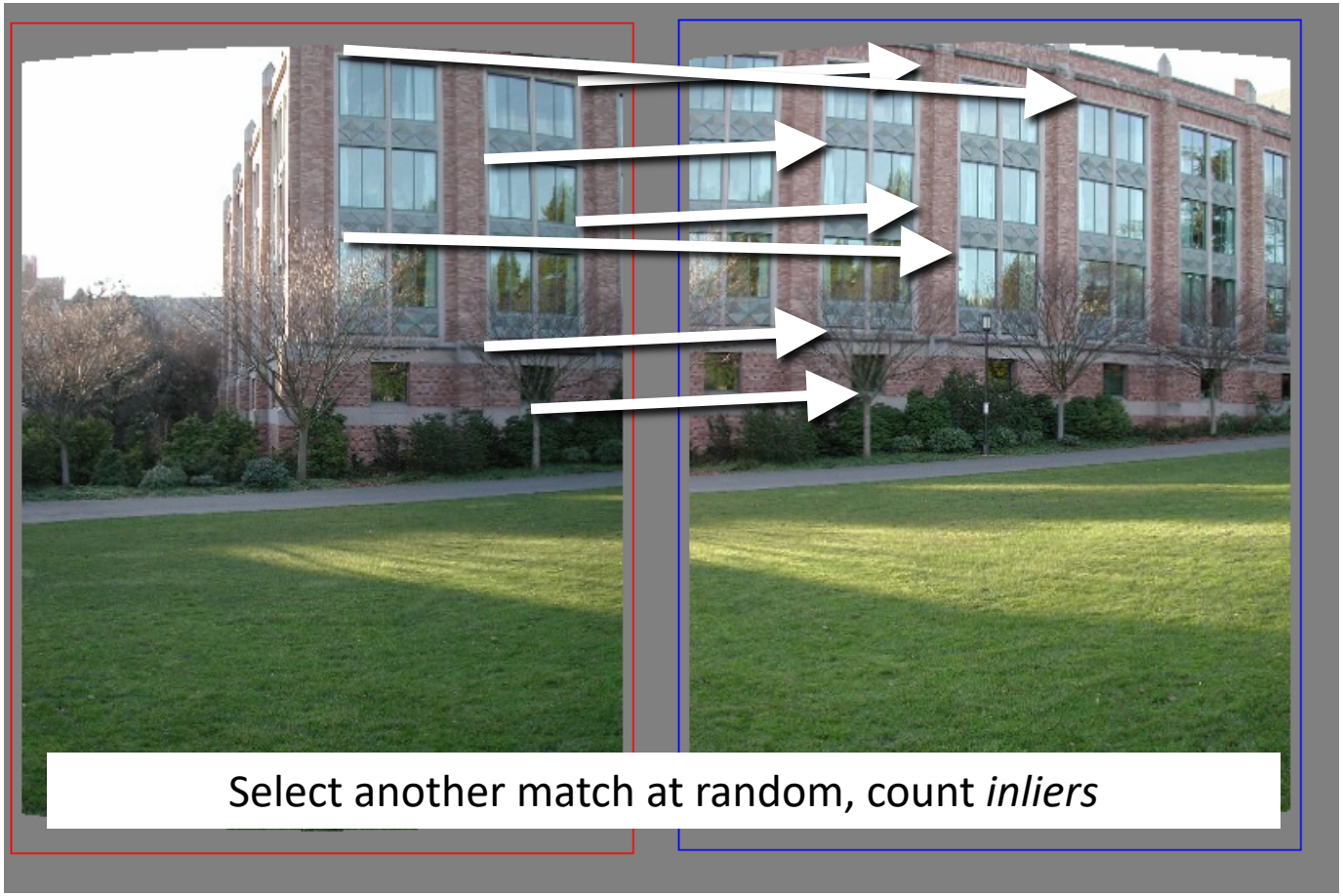
RANSAC for Translation



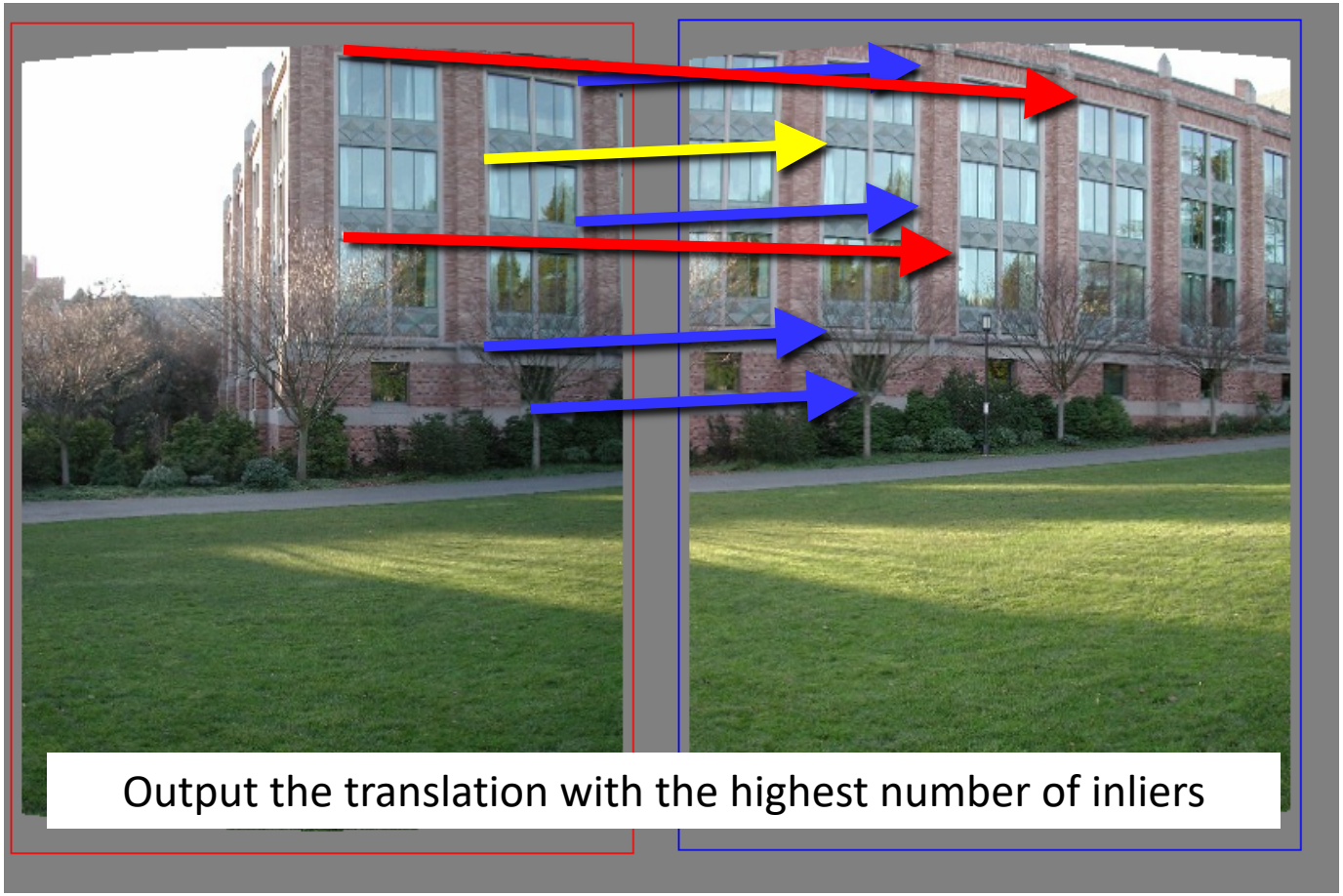
RANSAC for Translation



RANSAC for Translation



RANSAC for Translation



RANSAC

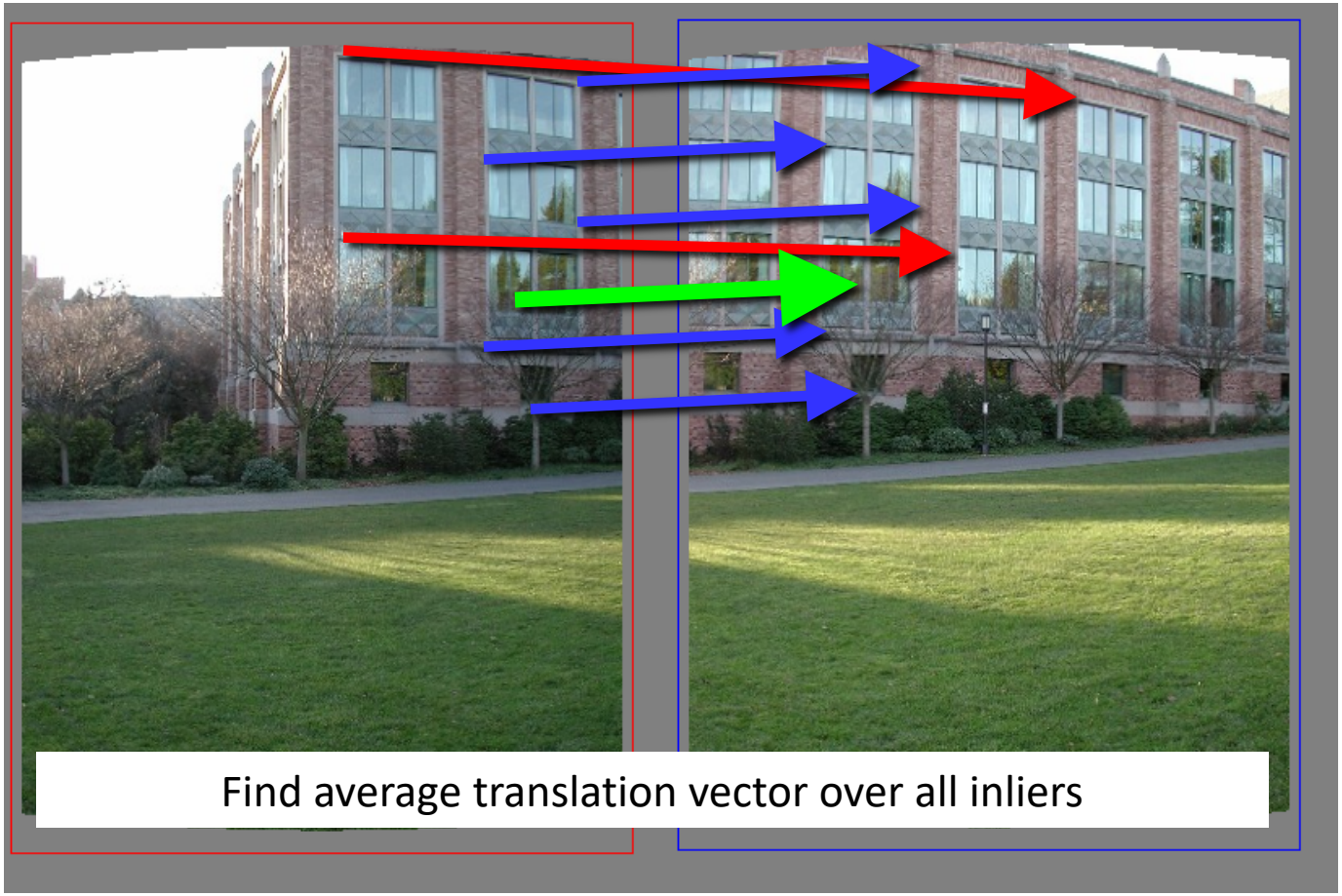
- Idea:
  - All the inliers will agree with each other on the translation vector;
  - The outliers will disagree with each other
    - RANSAC only has guarantees if there are < 50% outliers
  - “All good matches are alike; every bad match is bad in its own way.”

    – Tolstoy via Alyosha Efros

RANSAC

- General version:
  1. Randomly choose *s* samples
     - Typically *s* = minimum sample size that lets you fit a model
  2. Fit a model (e.g., transformation matrix) to those samples
  3. Count the number of inliers that approximately fit the model
  4. Repeat *N* times
  5. Choose the model that has the largest set of inliers

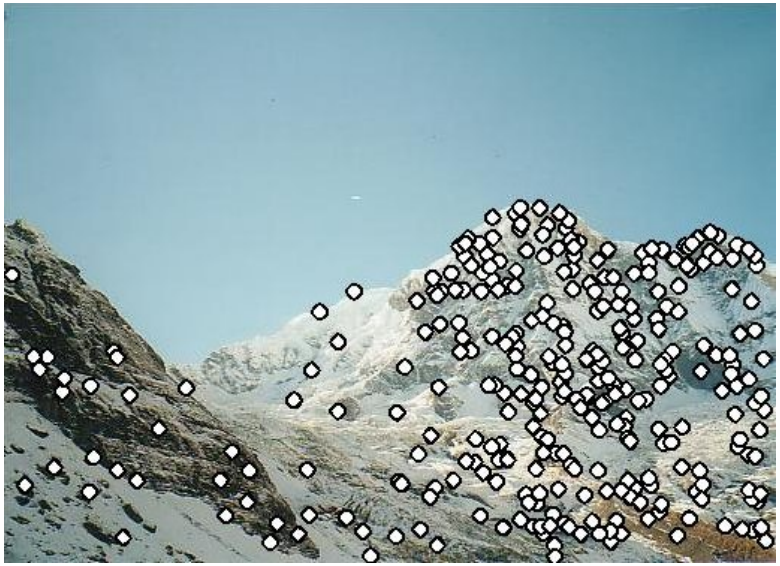
Final step: least squares fit

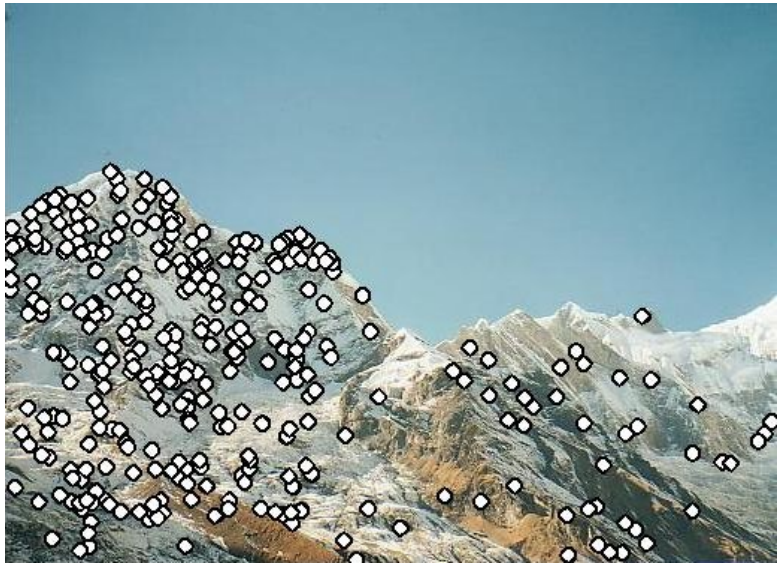


图像拼接

输入图像

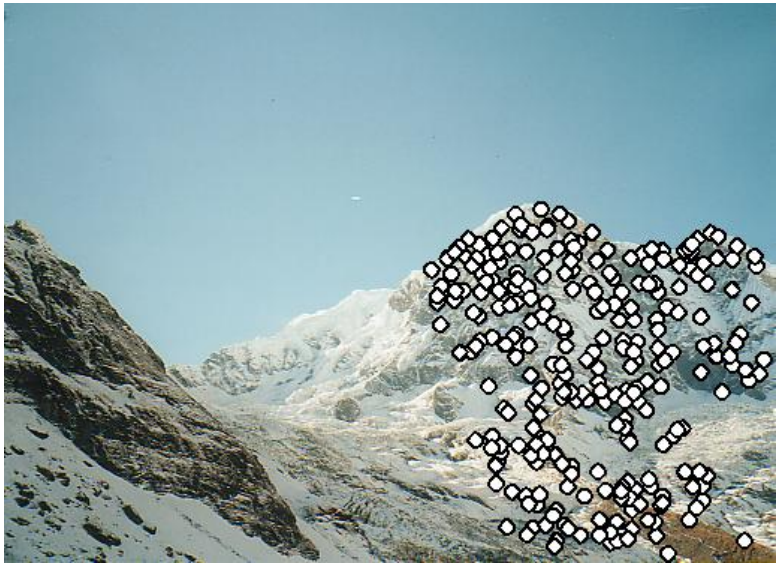
图像拼接

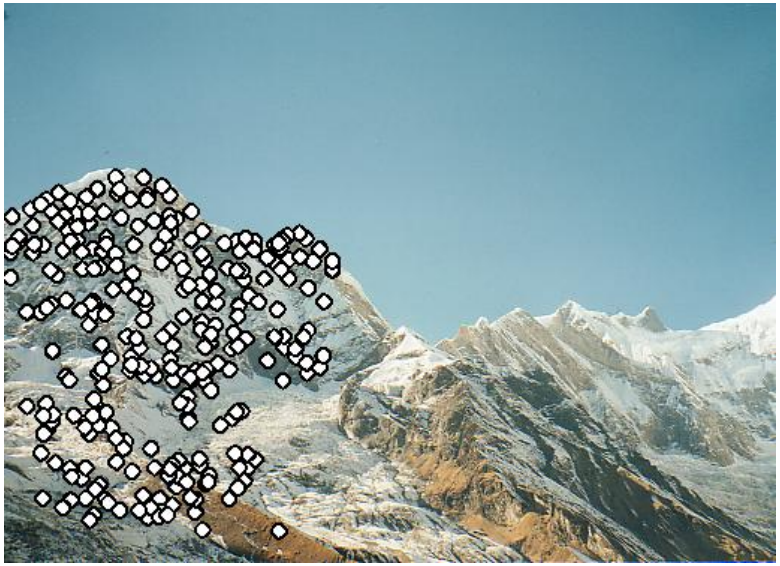




特征匹配

图像拼接





RANSAC计算变换矩阵

图像拼接

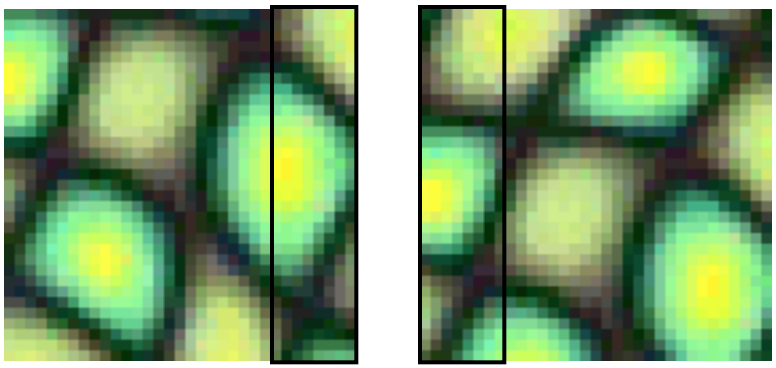
固定第一幅图，变换第二幅图

无缝融合

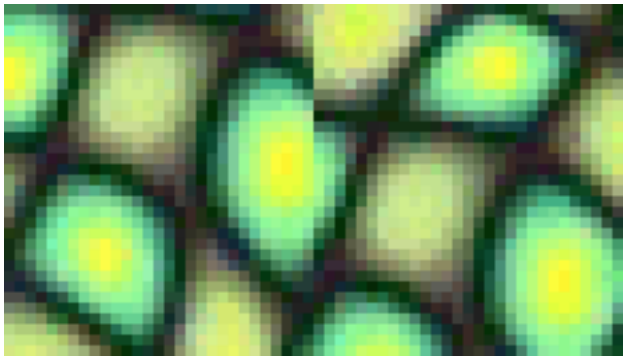
- Graphcut
- Poisson Image Editing

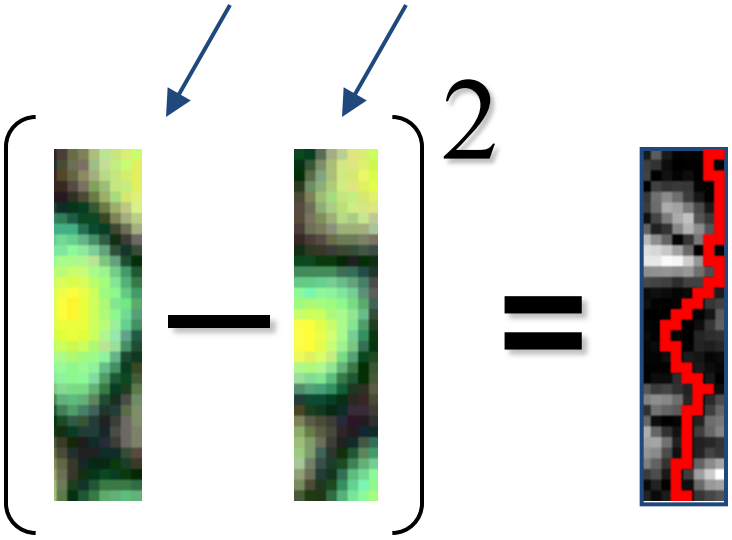
无缝融合

重叠的图像

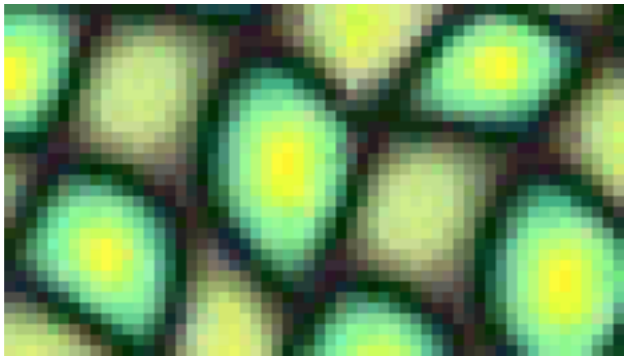


简单的接缝



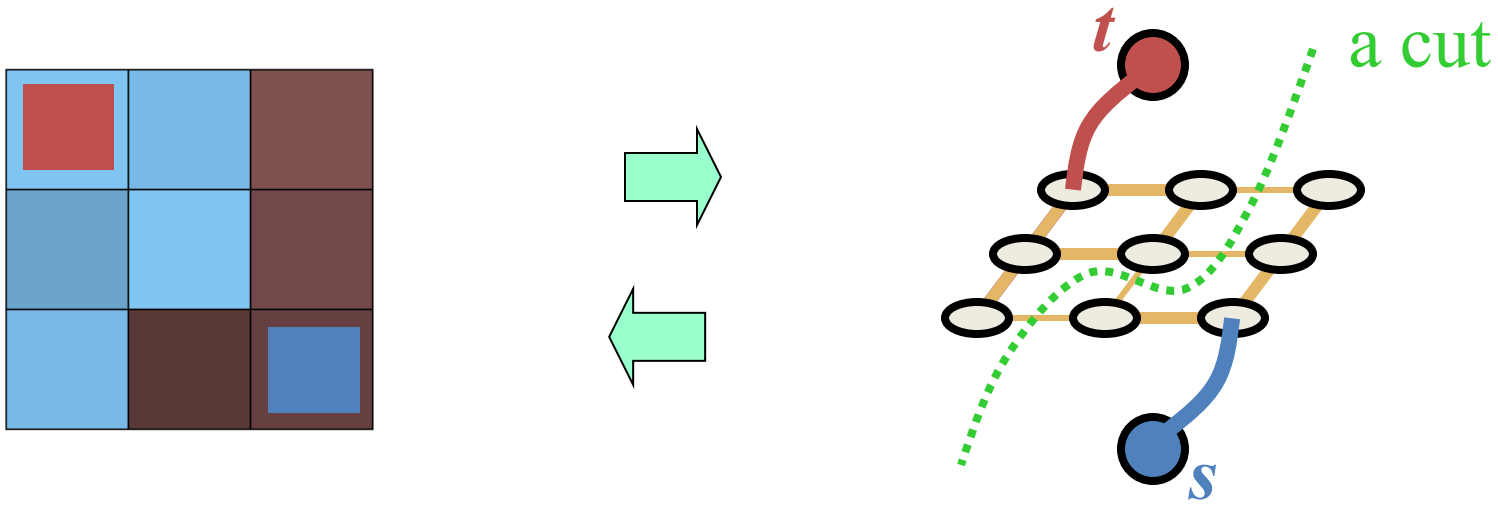


颜色差



最小化颜色差的接缝

无缝融合



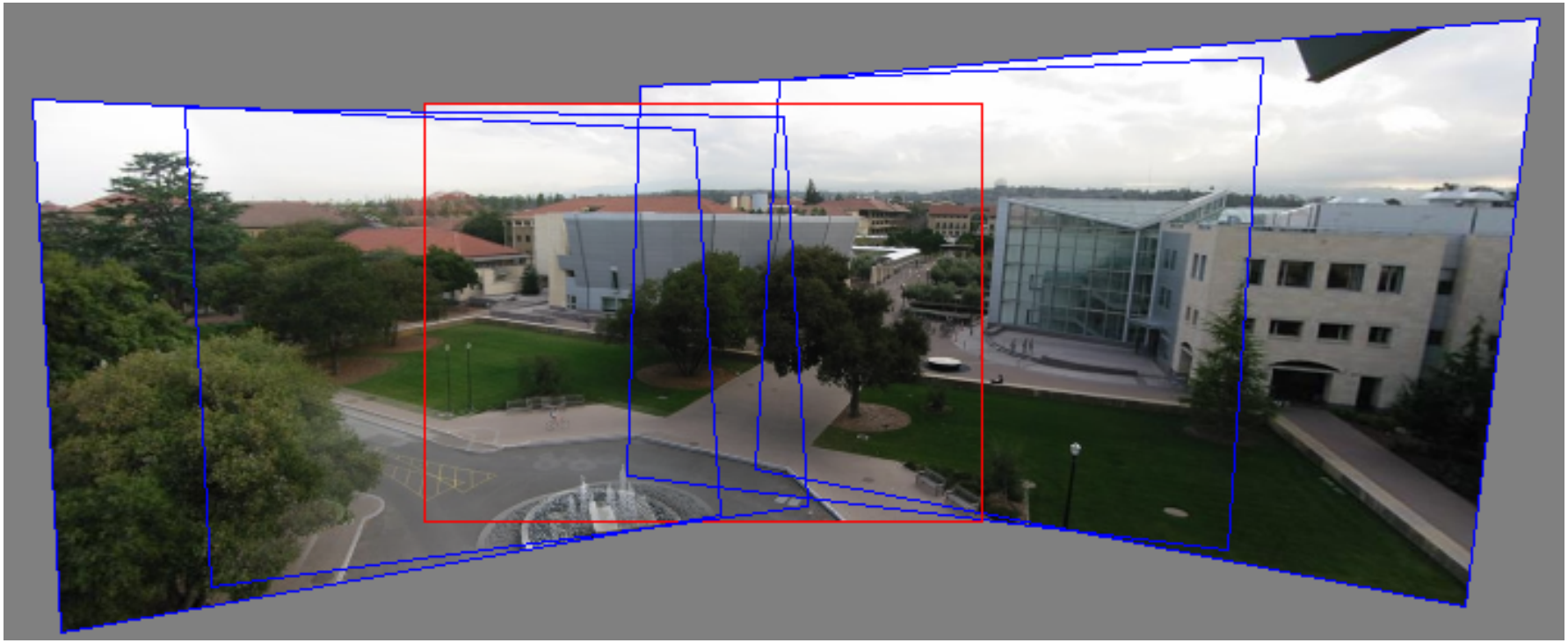
最大流最小割算法

多项式时间

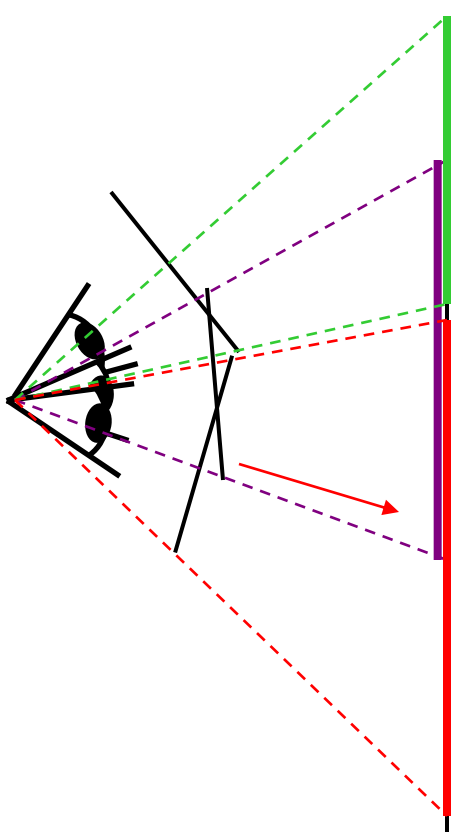
Panoramas

- Now we know how to create panoramas!

  1) Warp all images to a reference image; 2) merge them



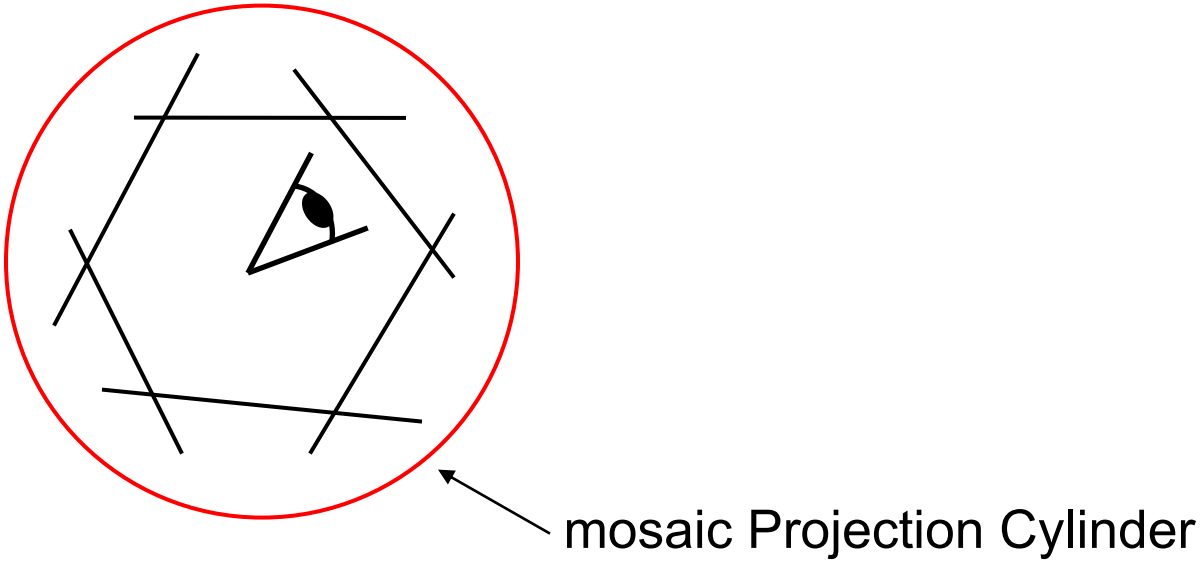
Do we have to project onto a plane?



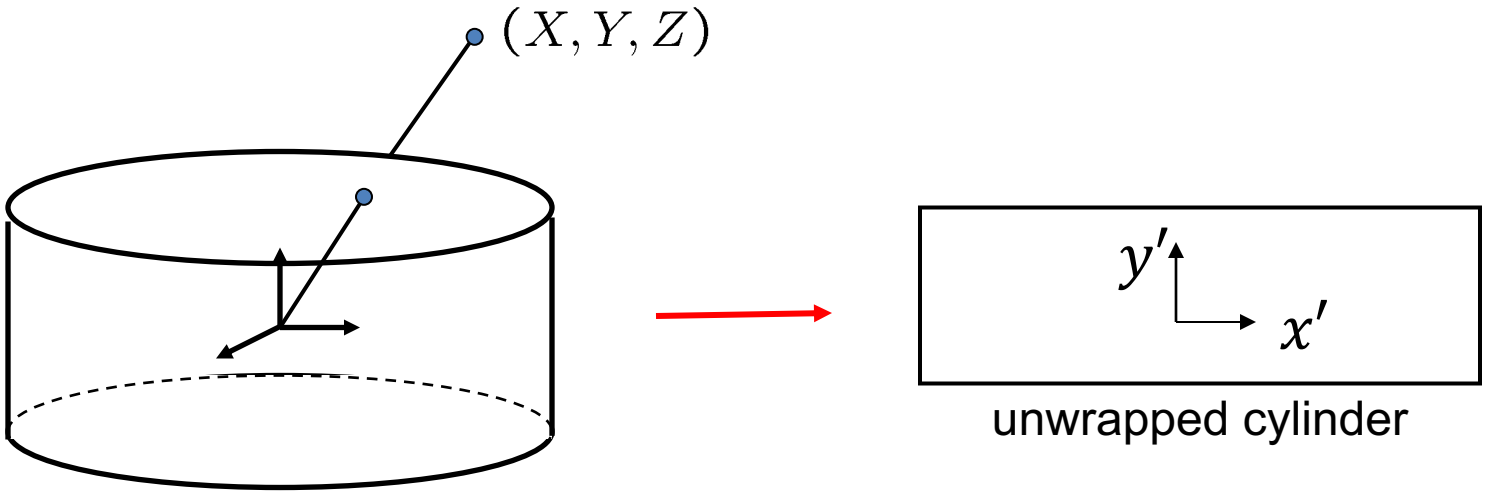
mosaic PP

Full Panoramas

- What if you want a 360° field of view?



Cylindrical projection

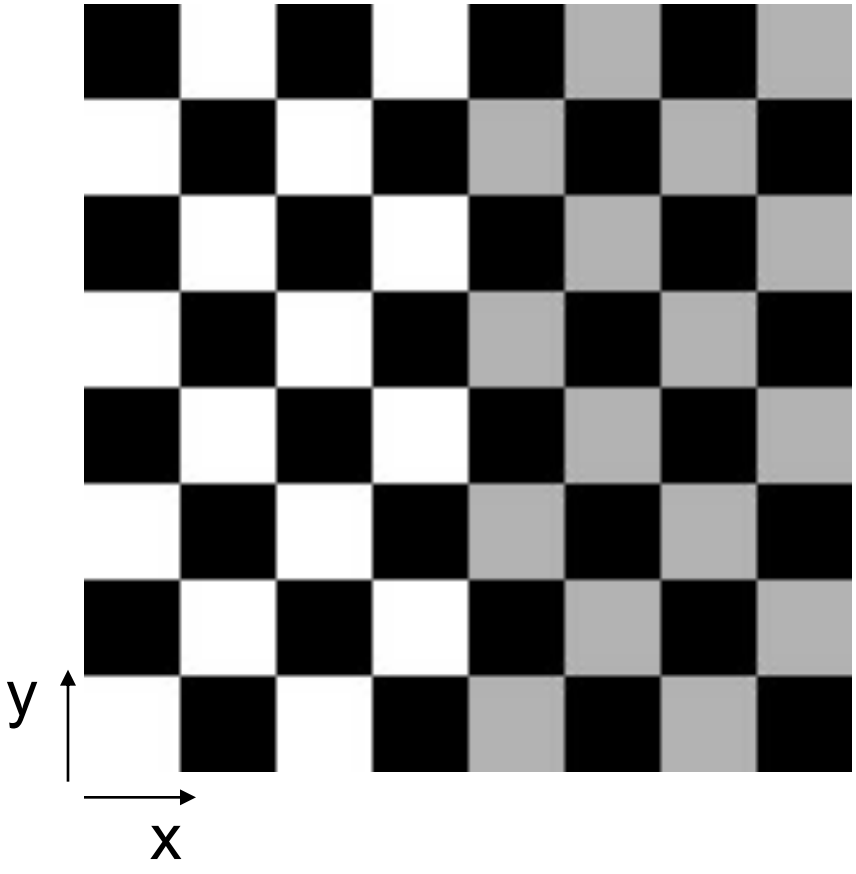


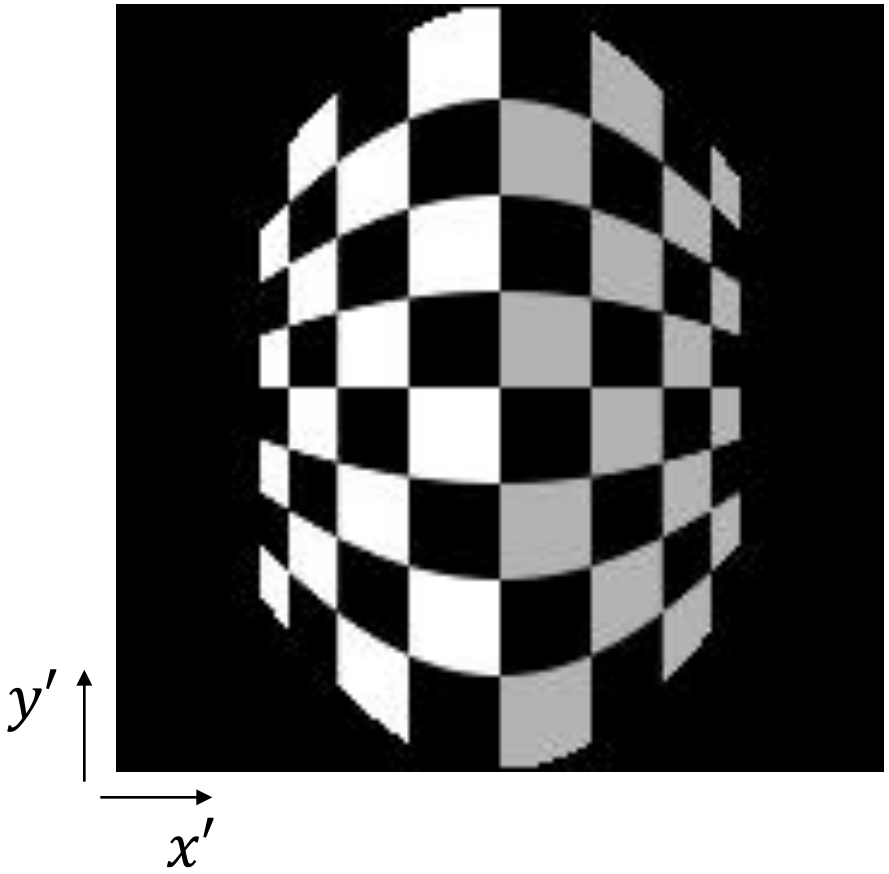
$$x' = rtan^{-1}\left(\frac{x}{f}\right)$$

$$y' = \frac{ry}{\sqrt{x^2+f^2}}$$

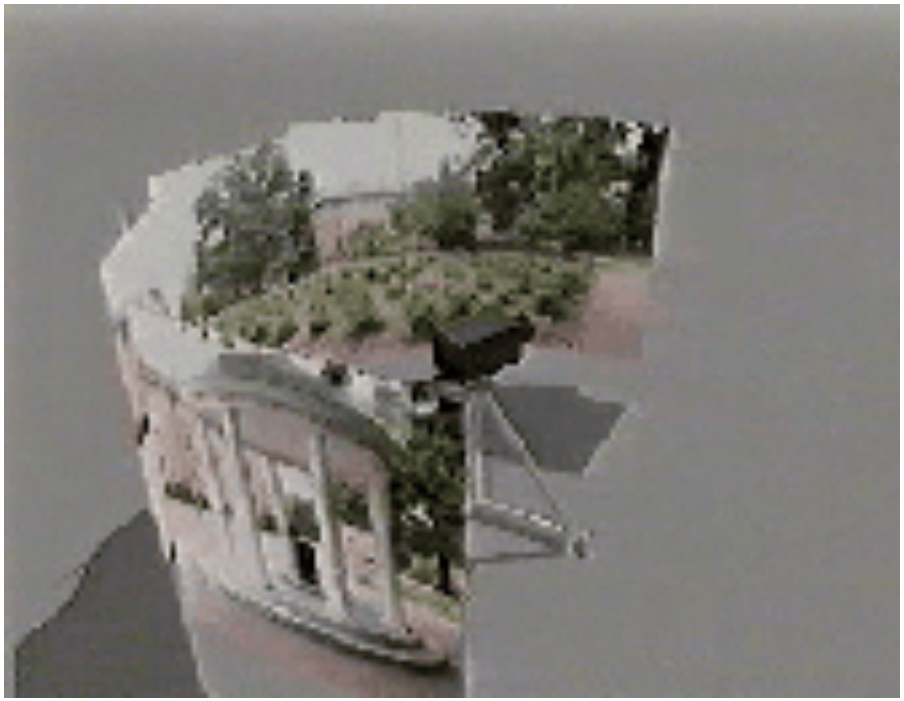
其中(x', y')为柱面上的坐标,(x, y)为平面图像坐标，其坐标原点都已移至图像中心，r 为柱面半径，f 为焦距。

Cylindrical projection





Cylindrical panoramas

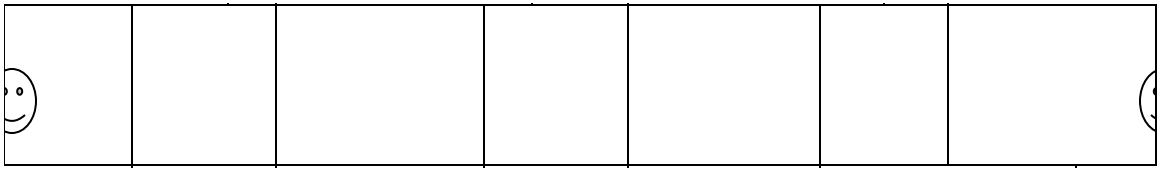


- Steps
  - Reproject each image onto a cylinder
  - Blend
  - Output the resulting mosaic

Cylindrical image stitching

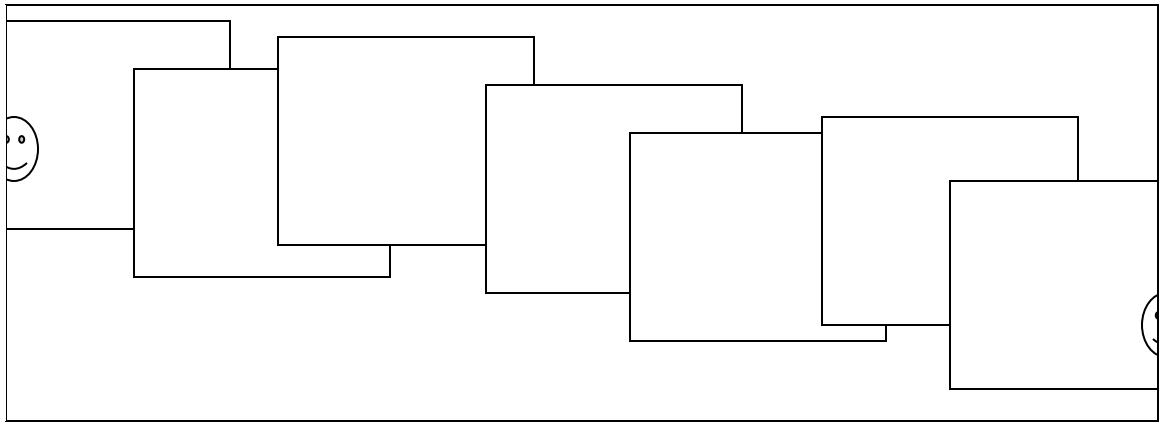
- What if you don’t know the camera rotation?
  - Solve for the camera rotations
    - A rotation of the camera is a **translation** of the cylinder!

Assembling the panorama



- Stitch pairs together, blend, then crop

Problem: Drift



- small (vertical) errors accumulate over time
- apply correction so that sum = 0 (for 360° pan.)

Full-view (360°) panoramas

VIA 9GAG.COM

Questions?