

# Image stitching

Digital Visual Effects

*Yung-Yu Chuang*

*with slides by Richard Szeliski, Steve Seitz, Matthew Brown and Vaclav Hlavac*

# Image stitching

- Stitching = alignment + blending

geometrical  
registration

photometric  
registration



# Applications of image stitching

---

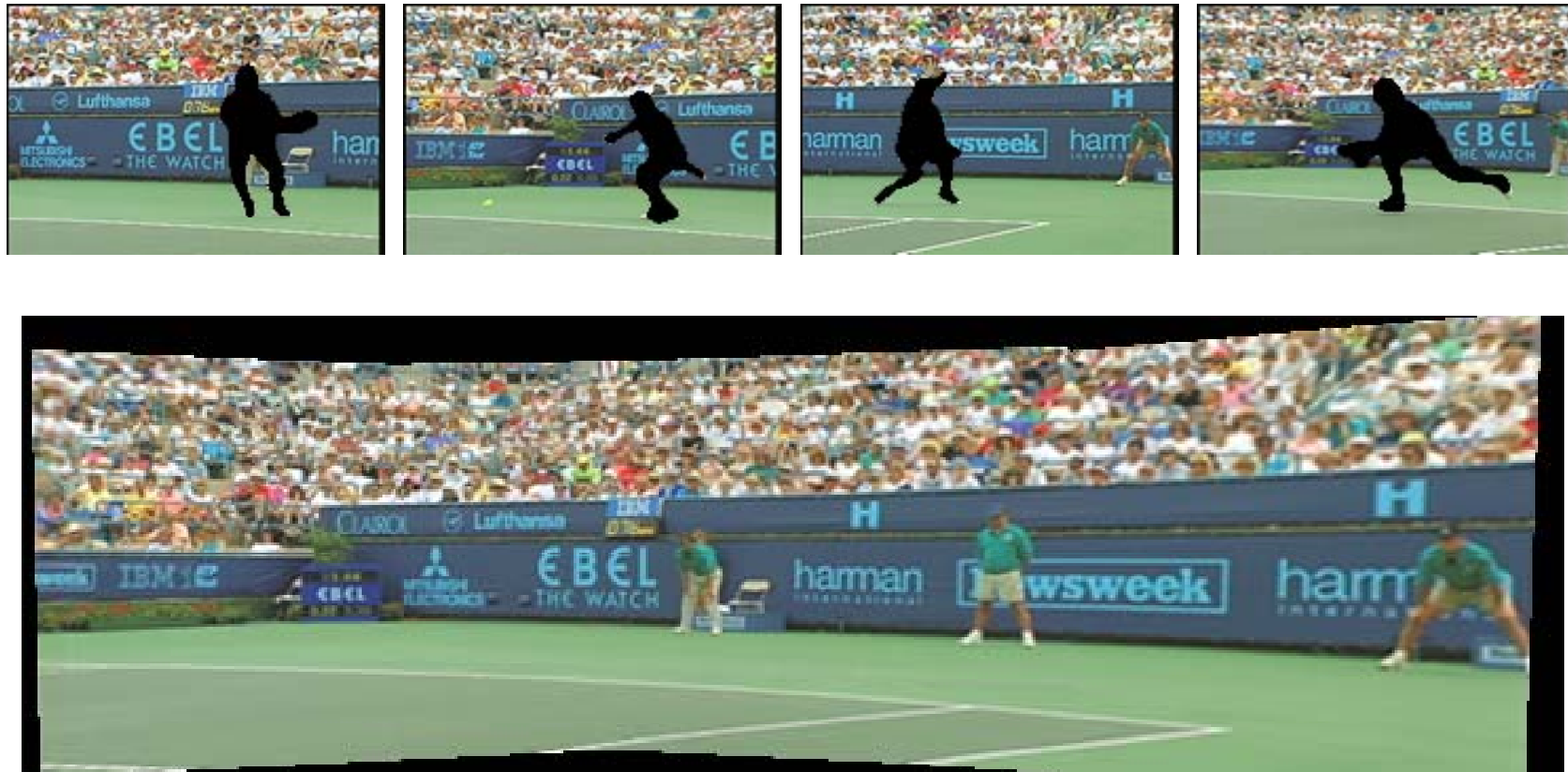
- Video stabilization
- Video summarization
- Video compression
- Video matting
- Panorama creation

# Video summarization



# Video compression

---



# Object removal

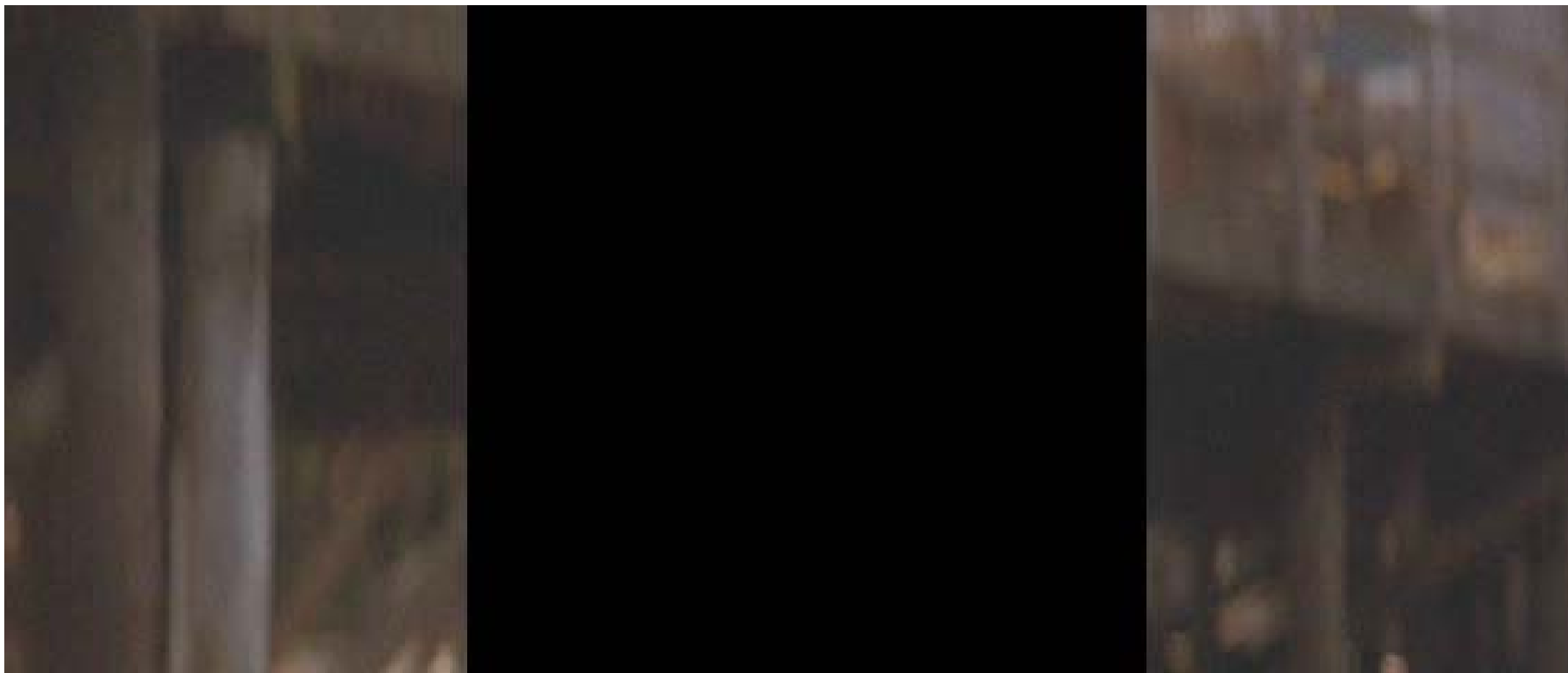
---



input video

# Object removal

---



remove foreground

# Object removal

---



estimate background

# Object removal

---



background estimation

# Panorama creation



# Why panorama?

---

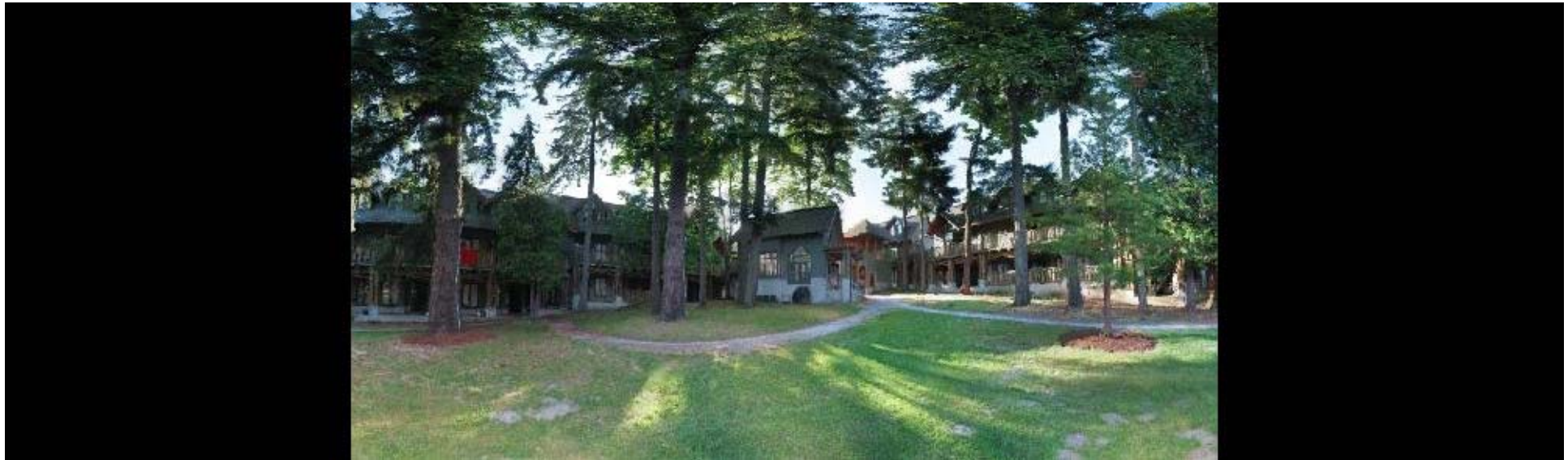
- Are you getting the whole picture?
  - Compact Camera FOV =  $50 \times 35^\circ$



# Why panorama?

---

- Are you getting the whole picture?
  - Compact Camera FOV =  $50 \times 35^\circ$
  - Human FOV =  $200 \times 135^\circ$



# Why panorama?

---

- Are you getting the whole picture?
  - Compact Camera FOV =  $50 \times 35^\circ$
  - Human FOV =  $200 \times 135^\circ$
  - Panoramic Mosaic =  $360 \times 180^\circ$



# Panorama examples

---

- Like HDR, it is a topic of computational photography, seeking ways to build a better camera mostly in software.
- Most consumer cameras have a panorama mode
- Mars:

[http://www.panoramas.dk/fullscreen3/f2\\_mars97.html](http://www.panoramas.dk/fullscreen3/f2_mars97.html)

- Earth:

<http://www.panoramas.dk/new-year-2006/taipei.html>

<http://www.360cities.net/>

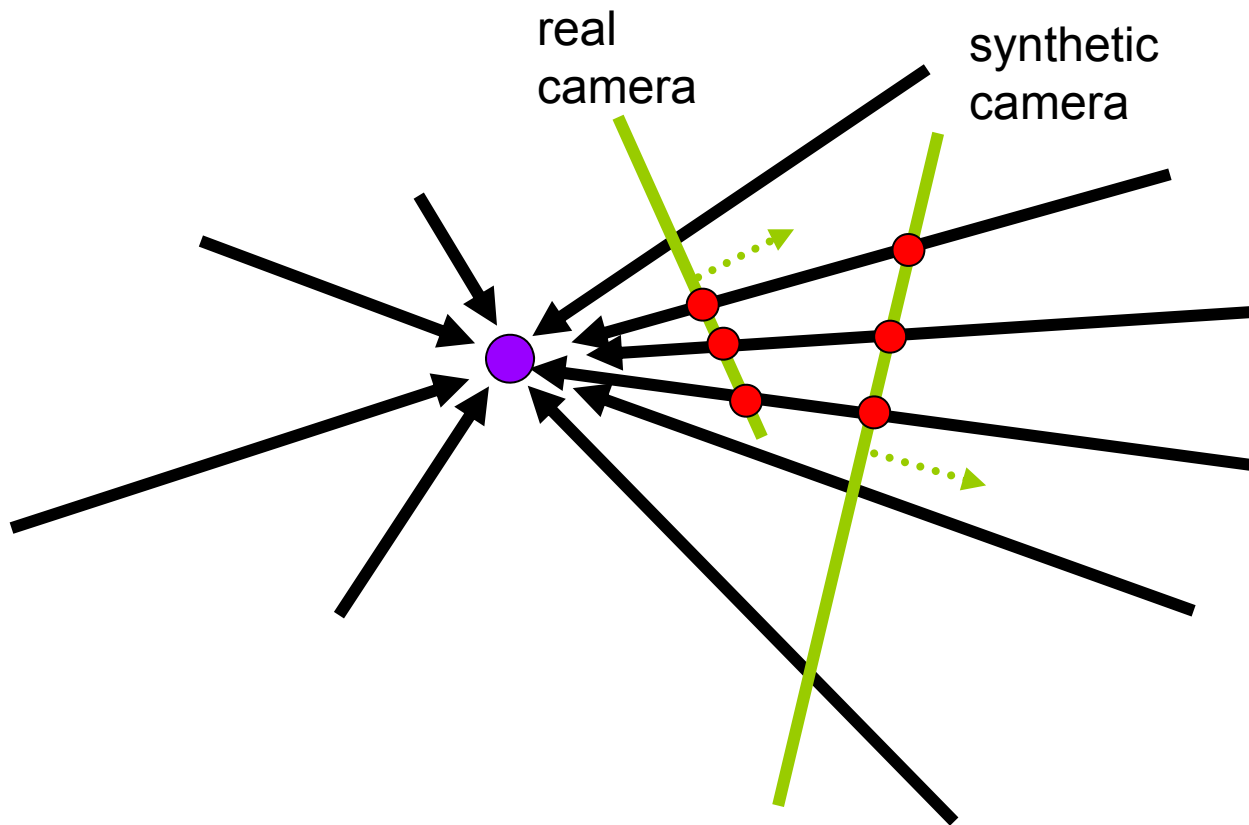
# What can be globally aligned?

---

- In image stitching, we seek for a matrix to globally warp one image into another. Are any two images of the same scene can be aligned this way?
  - Images captured with the same center of projection
  - A planar scene or far-away scene

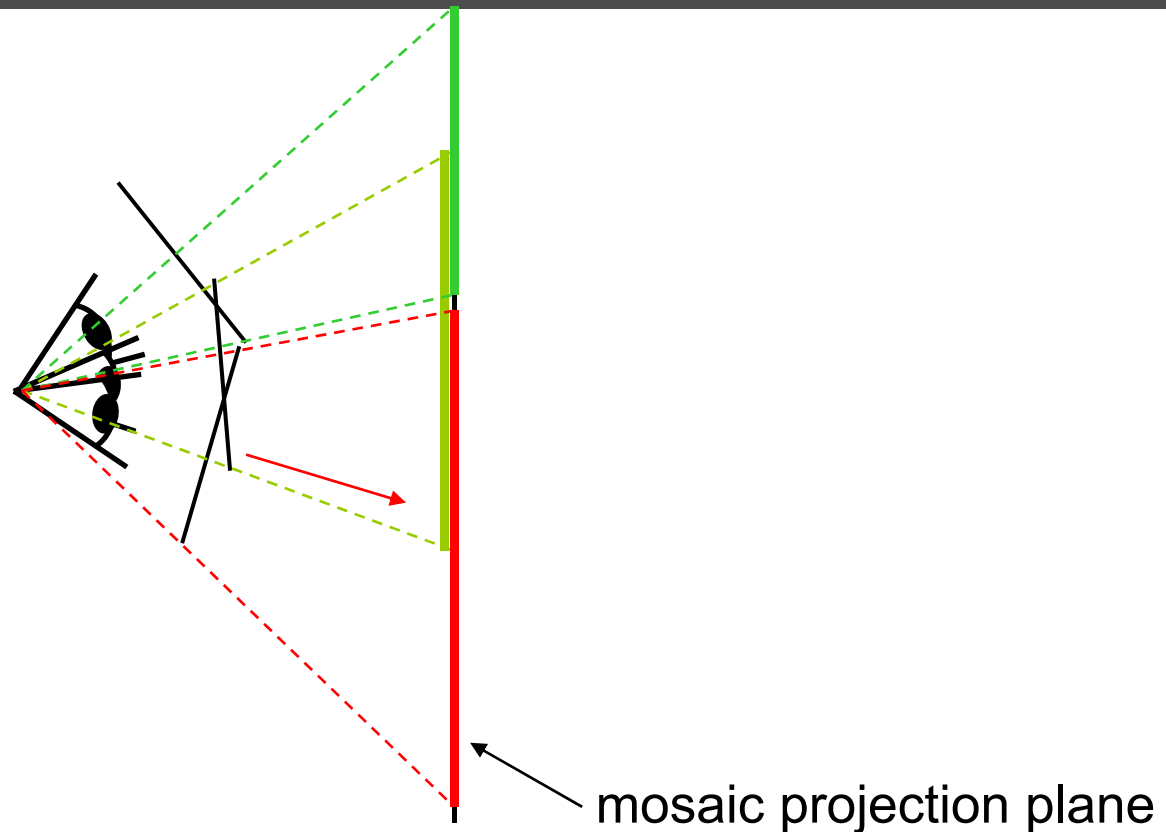
# A pencil of rays contains all views

---



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

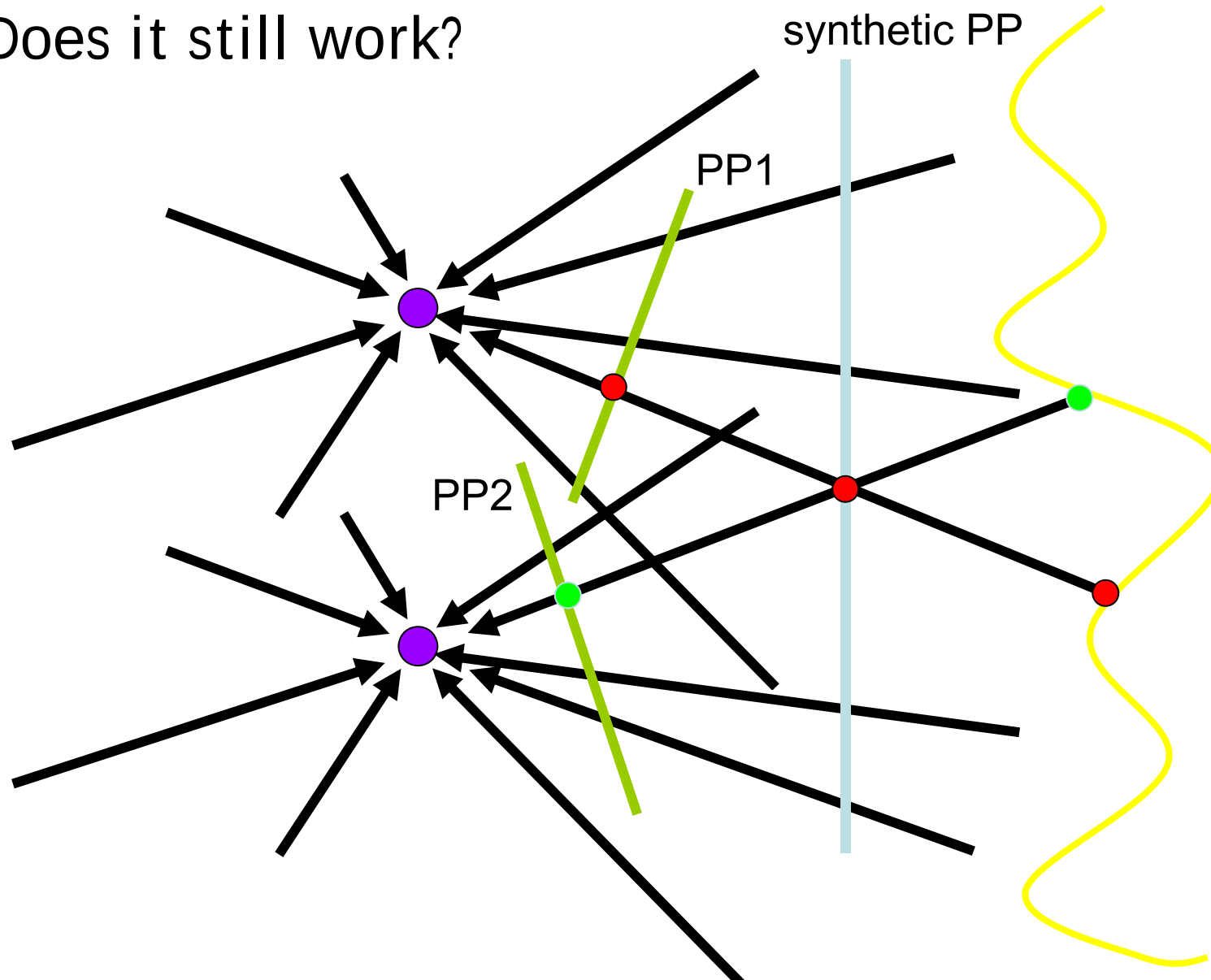
# Mosaic as an image reprojection



- The images are reprojected onto a common plane
- The mosaic is formed on this plane
- Mosaic is a *synthetic wide-angle camera*

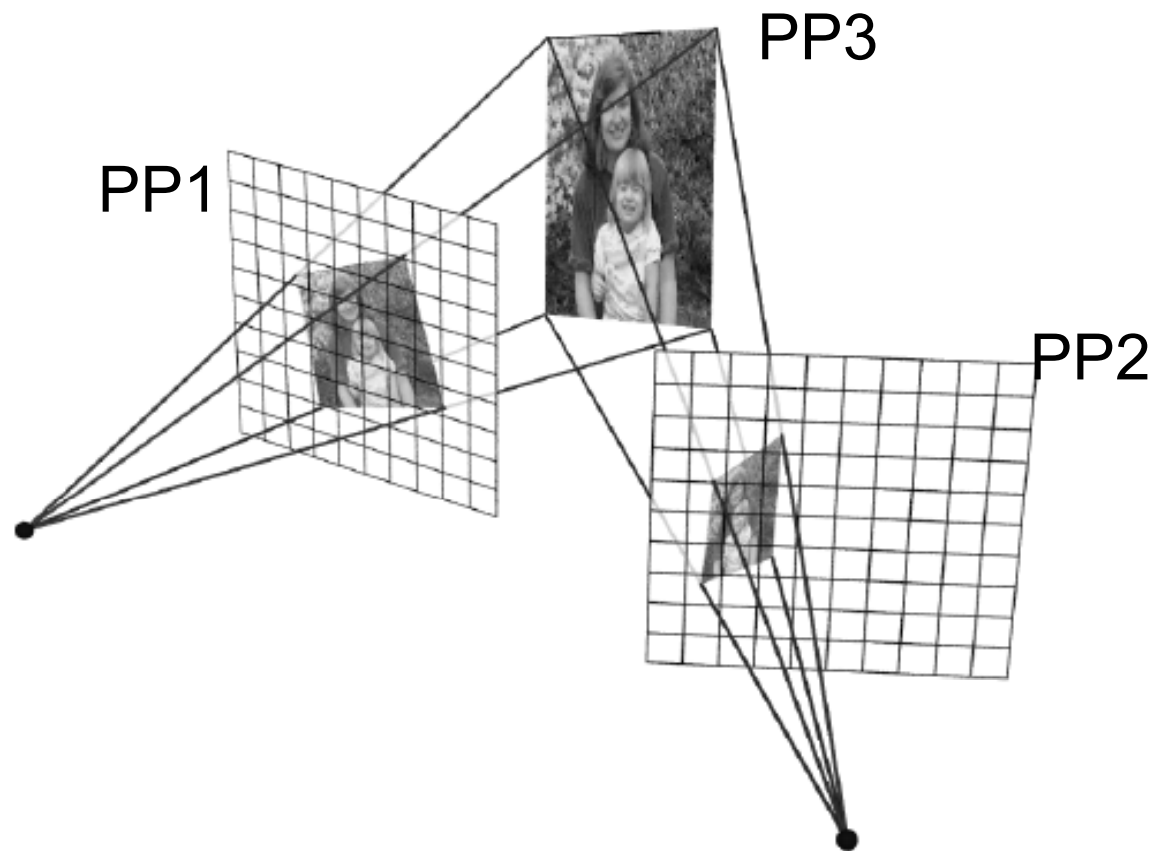
# Changing camera center

- Does it still work?



# Planar scene (or a faraway one)

---



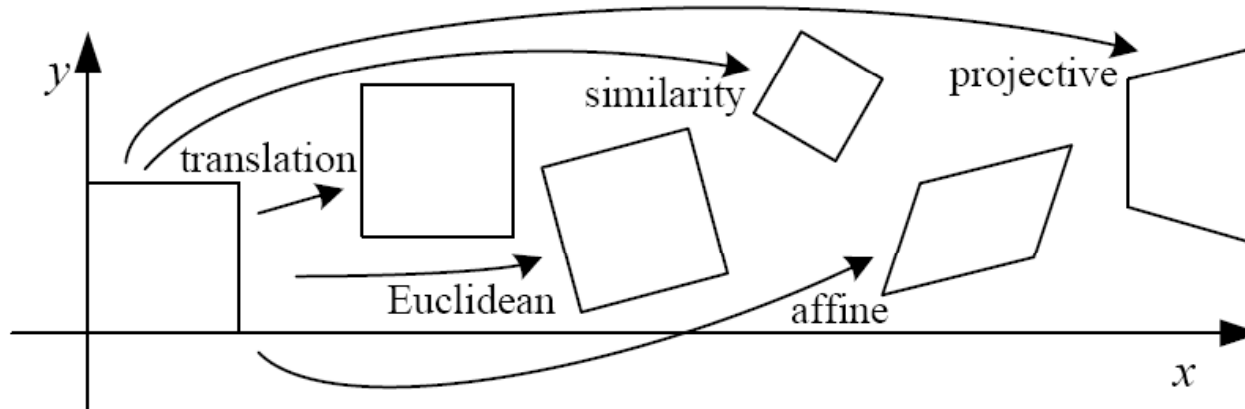
- PP3 is a projection plane of both centers of projection, so we are OK!
- This is how big aerial photographs are made

# Motion models

---

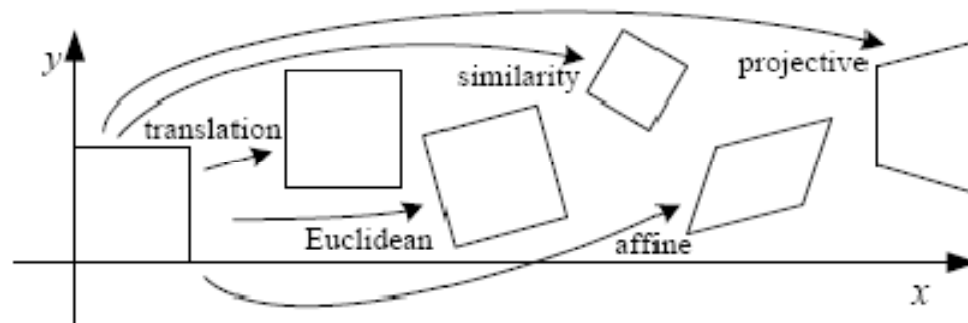
- Parametric models as the assumptions on the relation between two images.

# 2D Motion models

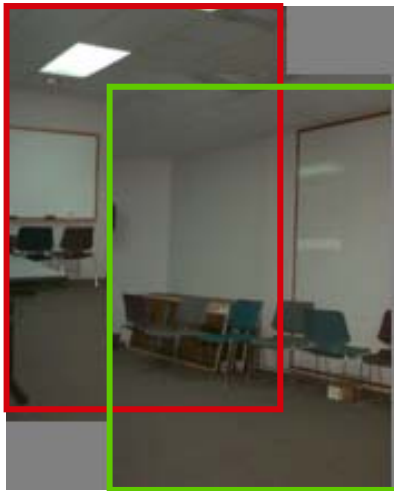


| Name              | Matrix                                                 | # D.O.F. | Preserves:        | Icon                                                                                  |
|-------------------|--------------------------------------------------------|----------|-------------------|---------------------------------------------------------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

# Motion models

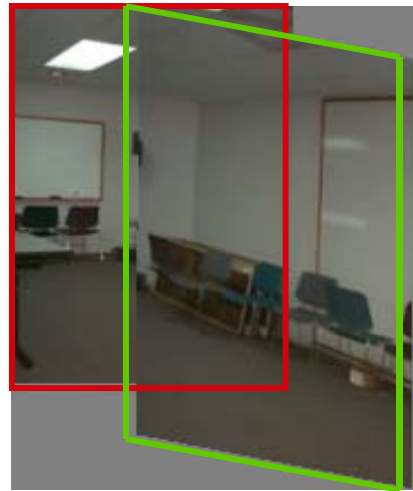


**Translation**



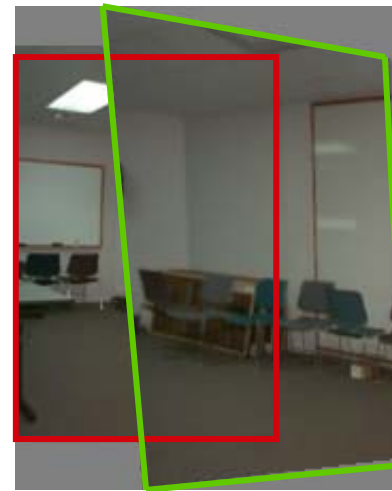
**2 unknowns**

**Affine**



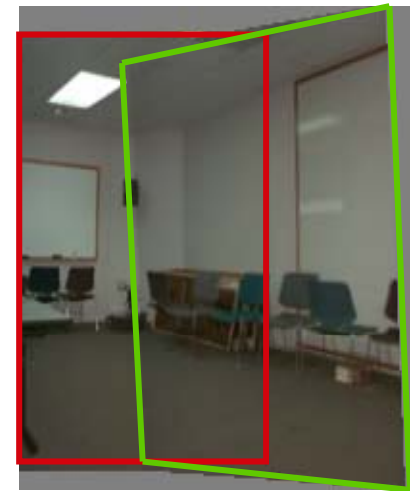
**6 unknowns**

**Perspective**



**8 unknowns**

**3D rotation**

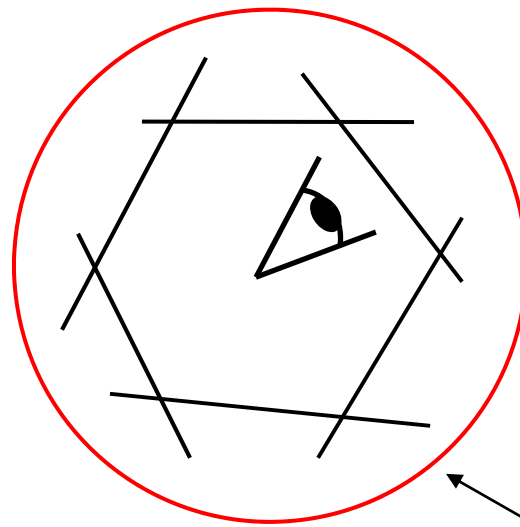


**3 unknowns**

# A case study: cylindrical panorama

---

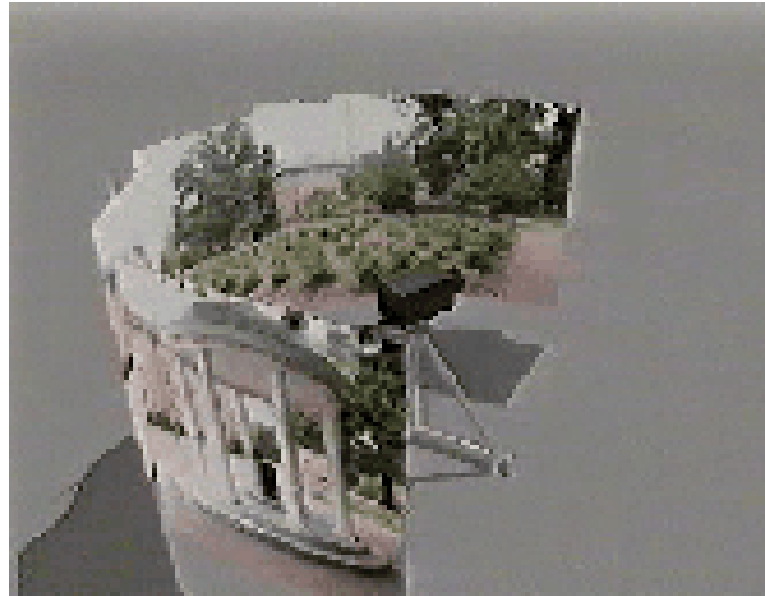
- What if you want a 360° field of view?



mosaic projection cylinder

# Cylindrical panoramas

---



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

# Cylindrical panorama

---

1. Take pictures on a tripod (or handheld)
2. Warp to cylindrical coordinate
3. Compute pairwise alignments
4. Fix up the end-to-end alignment
5. Blending
6. Crop the result and import into a viewer

It is required to do radial distortion correction for better stitching results!

# Taking pictures



Kaidan panoramic tripod head

# Translation model

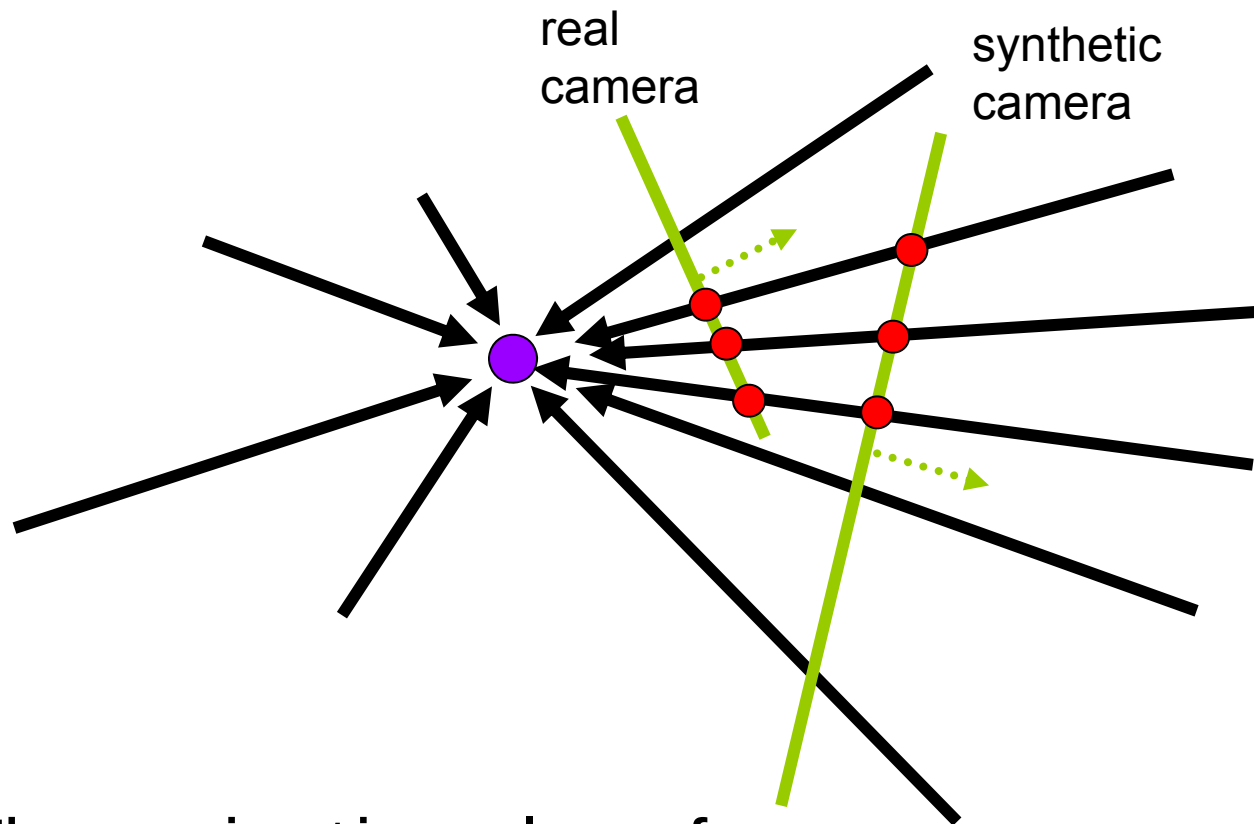
---



Try to align this in PaintShop Pro

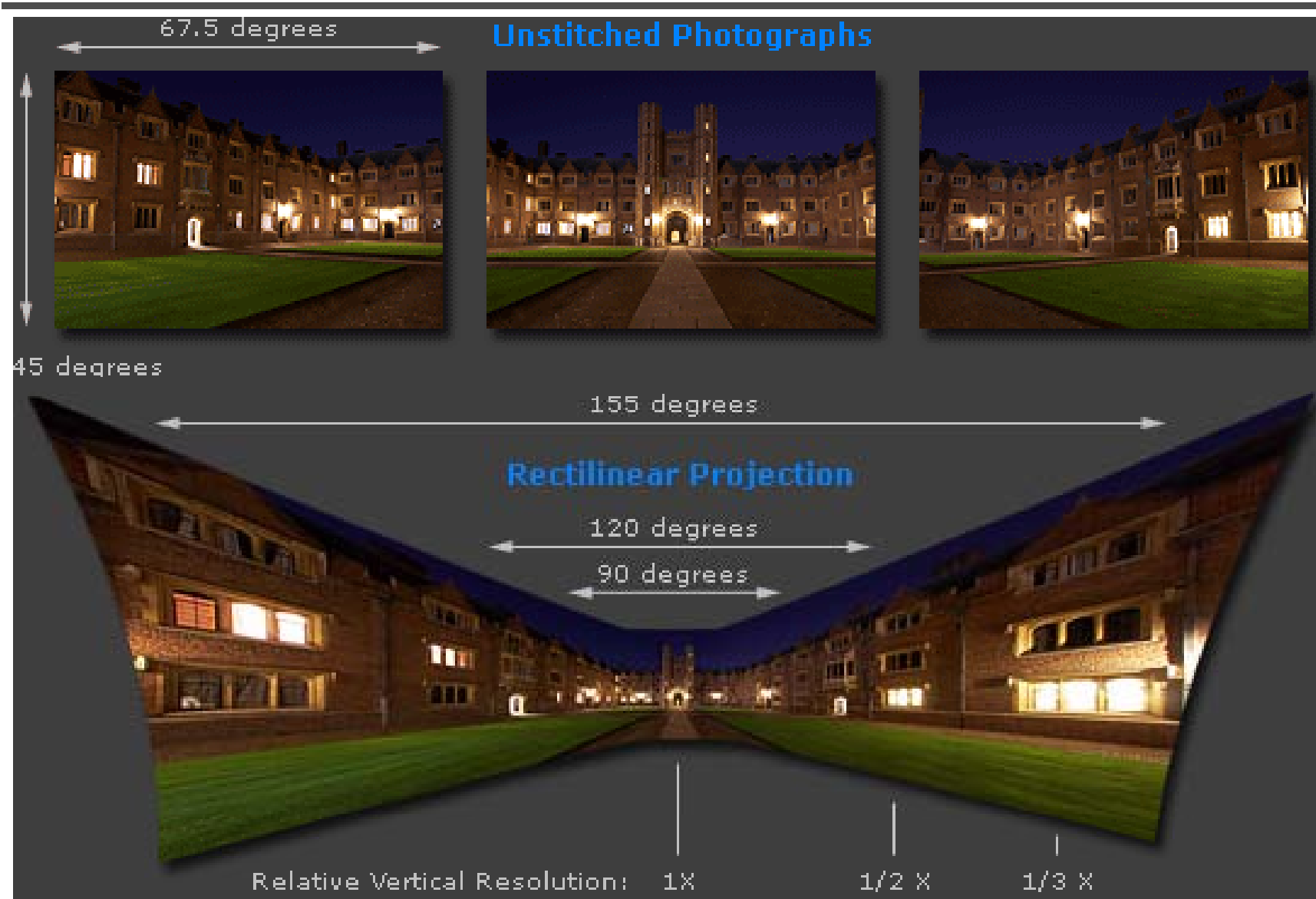
# Where should the synthetic camera be

---



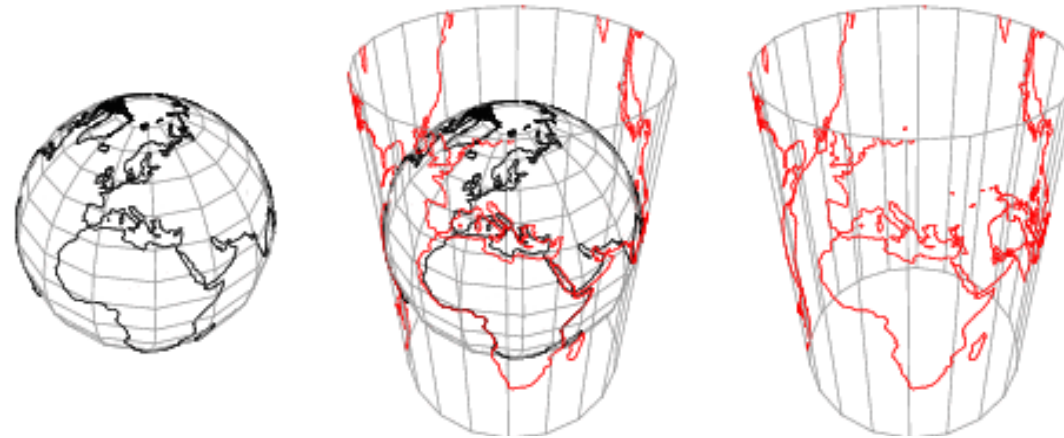
- The projection plan of some camera
- Onto a cylinder

# Cylindrical projection

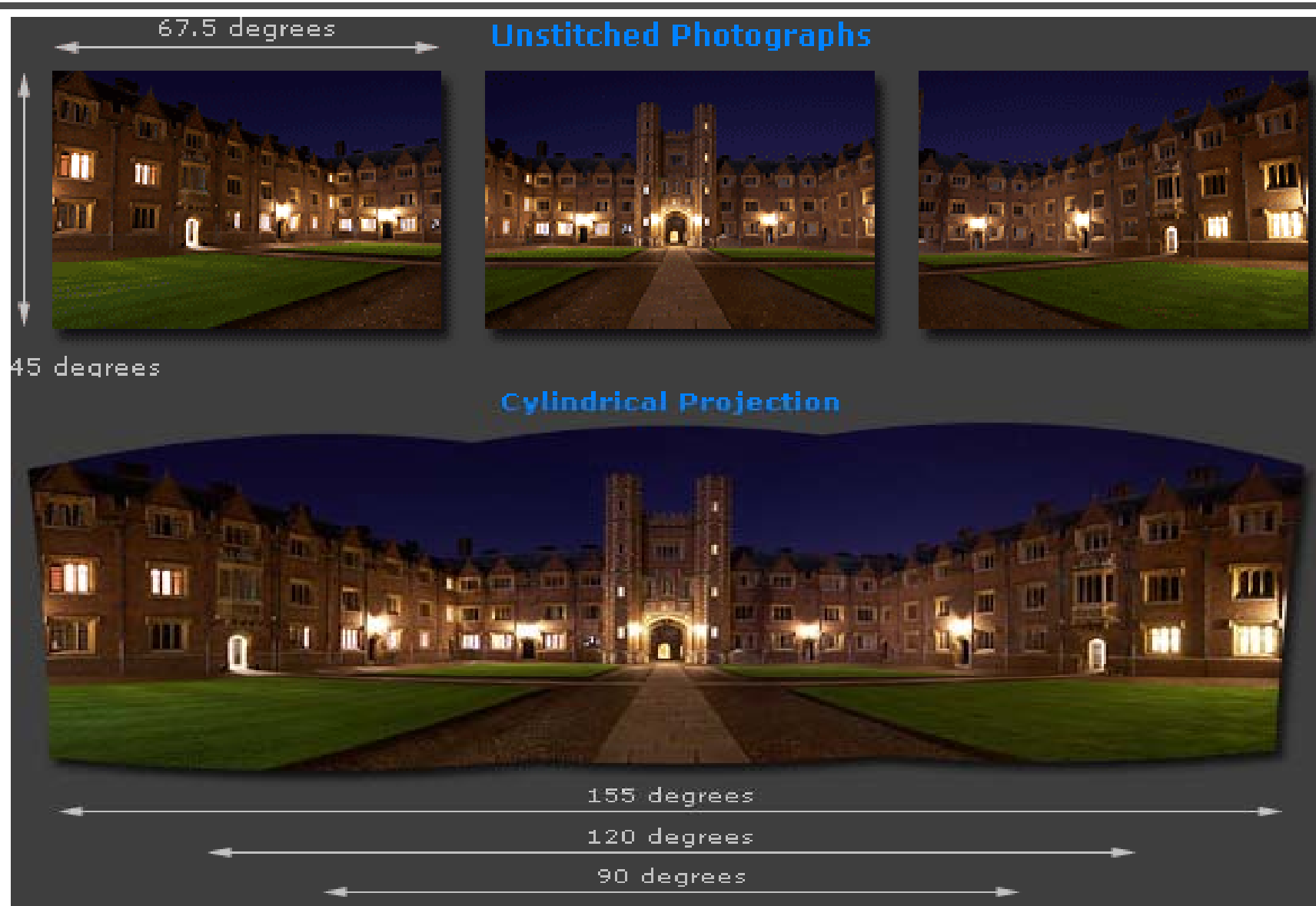


# Cylindrical projection

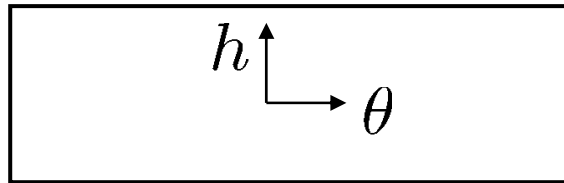
---



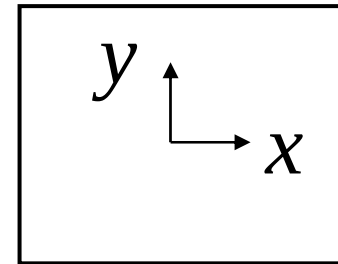
# Cylindrical projection



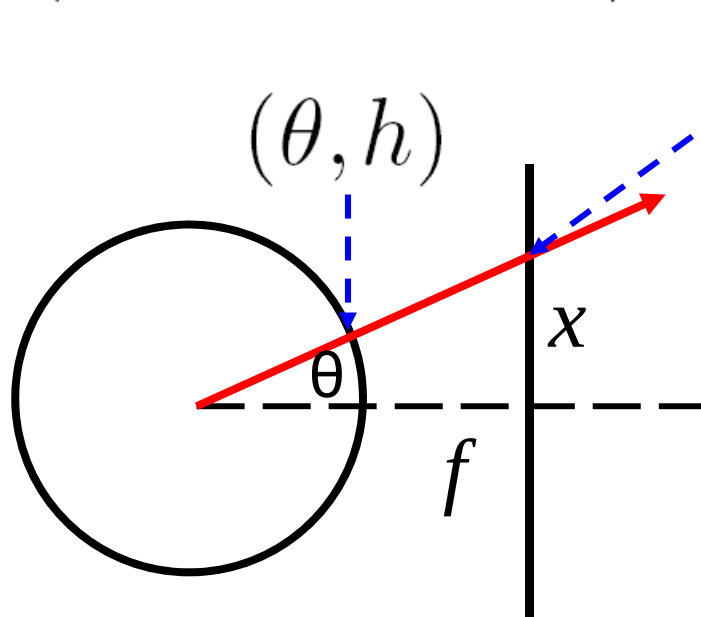
# Cylindrical projection



unwrapped cylinder

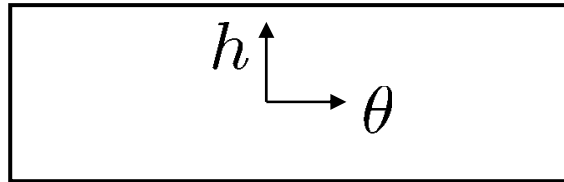


$$(\sin \theta, h, \cos \theta) \propto (x, y, f)$$

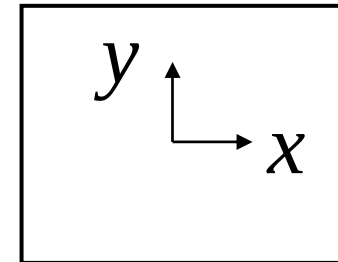


$$\theta = \tan^{-1} \frac{x}{f}$$

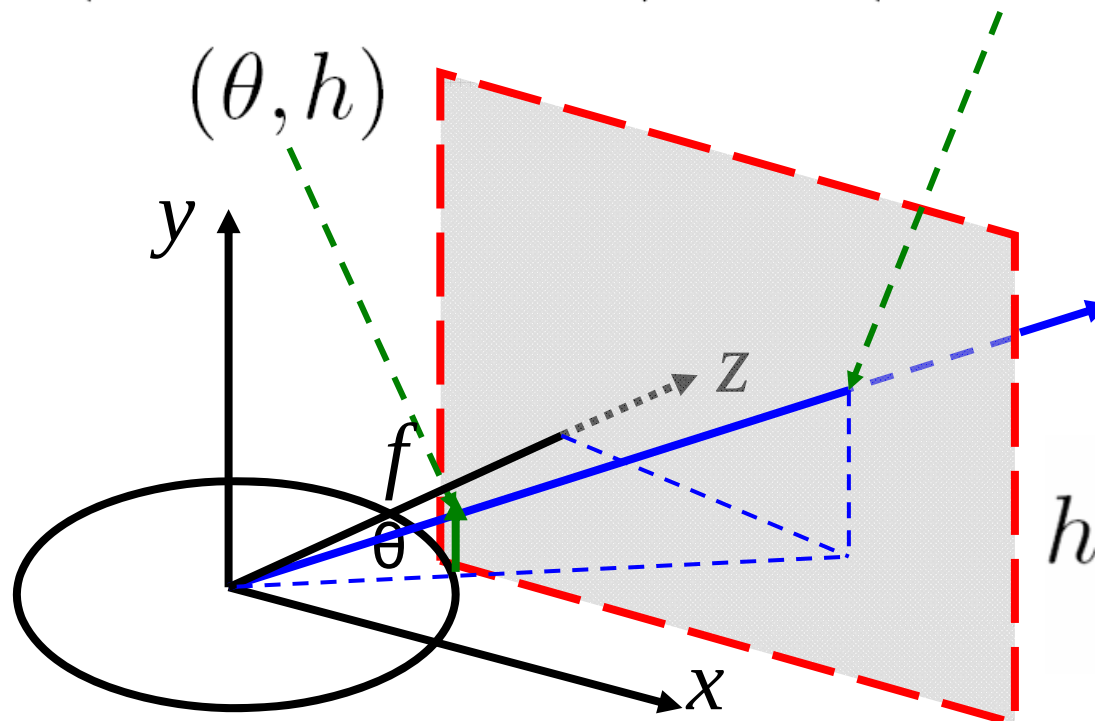
# Cylindrical projection



unwrapped cylinder

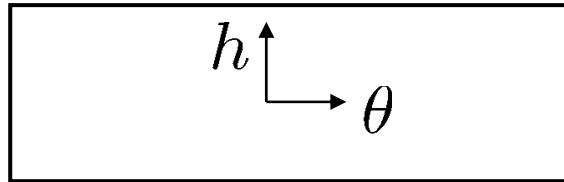


$$(\sin \theta, h, \cos \theta) \propto (x, y, f)$$

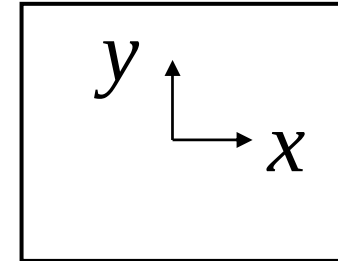


$$h = \frac{y}{\sqrt{x^2 + f^2}}$$

# Cylindrical projection



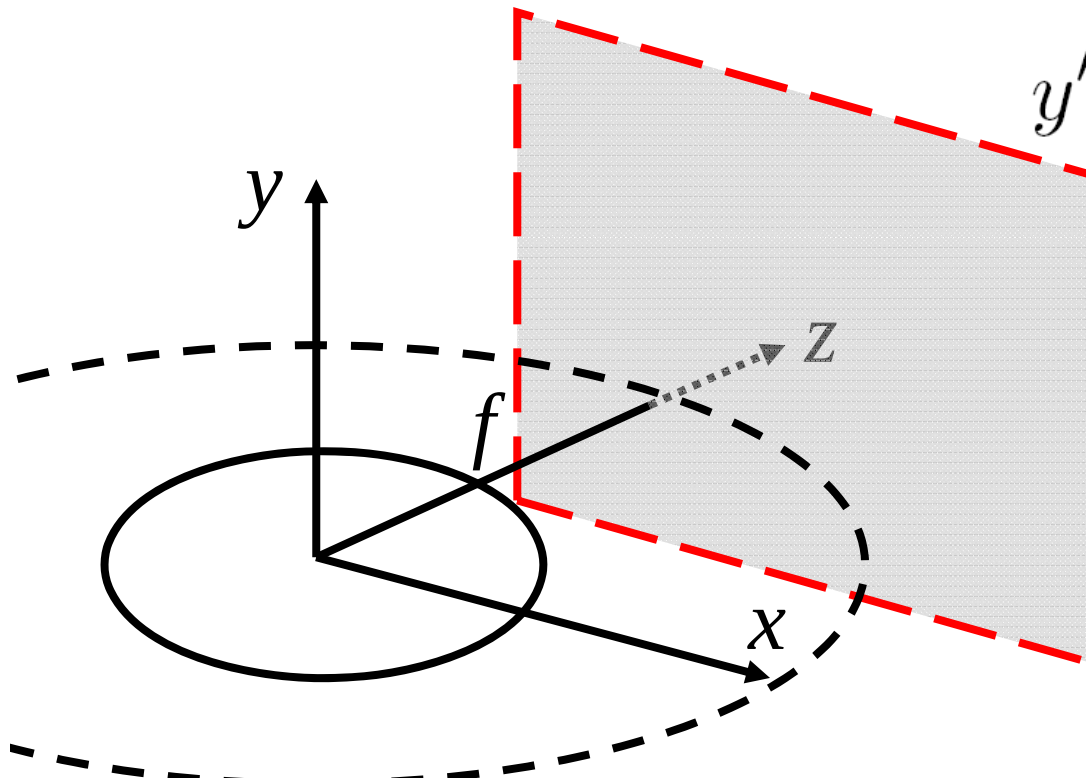
unwrapped cylinder



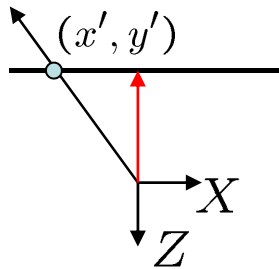
$$x' = s\theta = s \tan^{-1} \frac{x}{f}$$

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

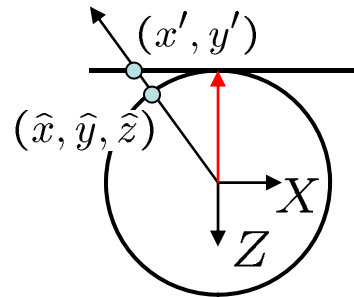
$s=f$  gives less distortion



# Cylindrical reprojection



top-down view



**Focal length** – the dirty secret...

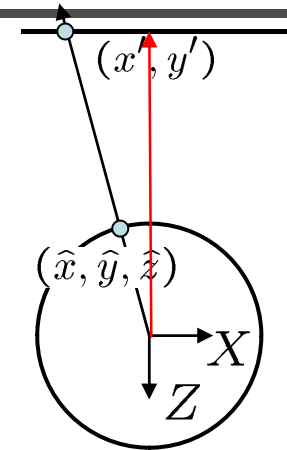
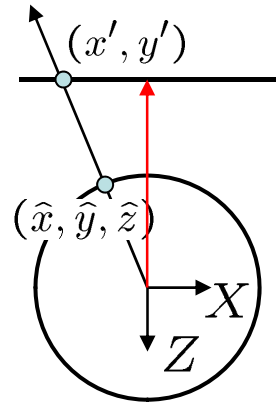


Image 384x300



$f = 180$  (pixels)

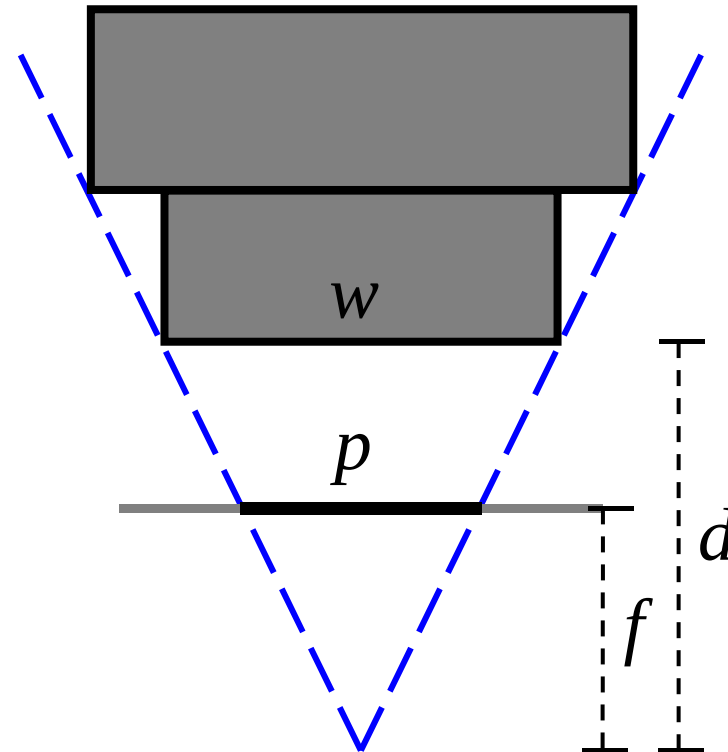
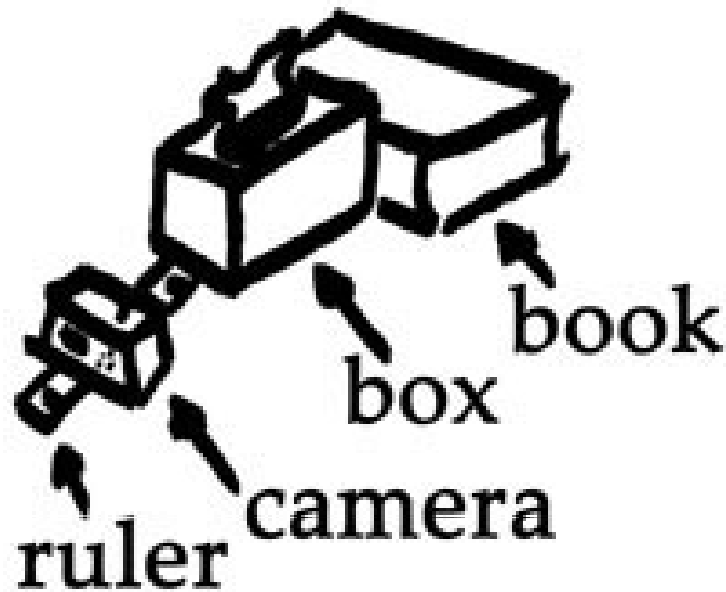


$f = 280$



$f = 380$

# A simple method for estimating $f$



Or, you can use other software, such as AutoStich, to help.

# Input images

---



# Cylindrical warping

---



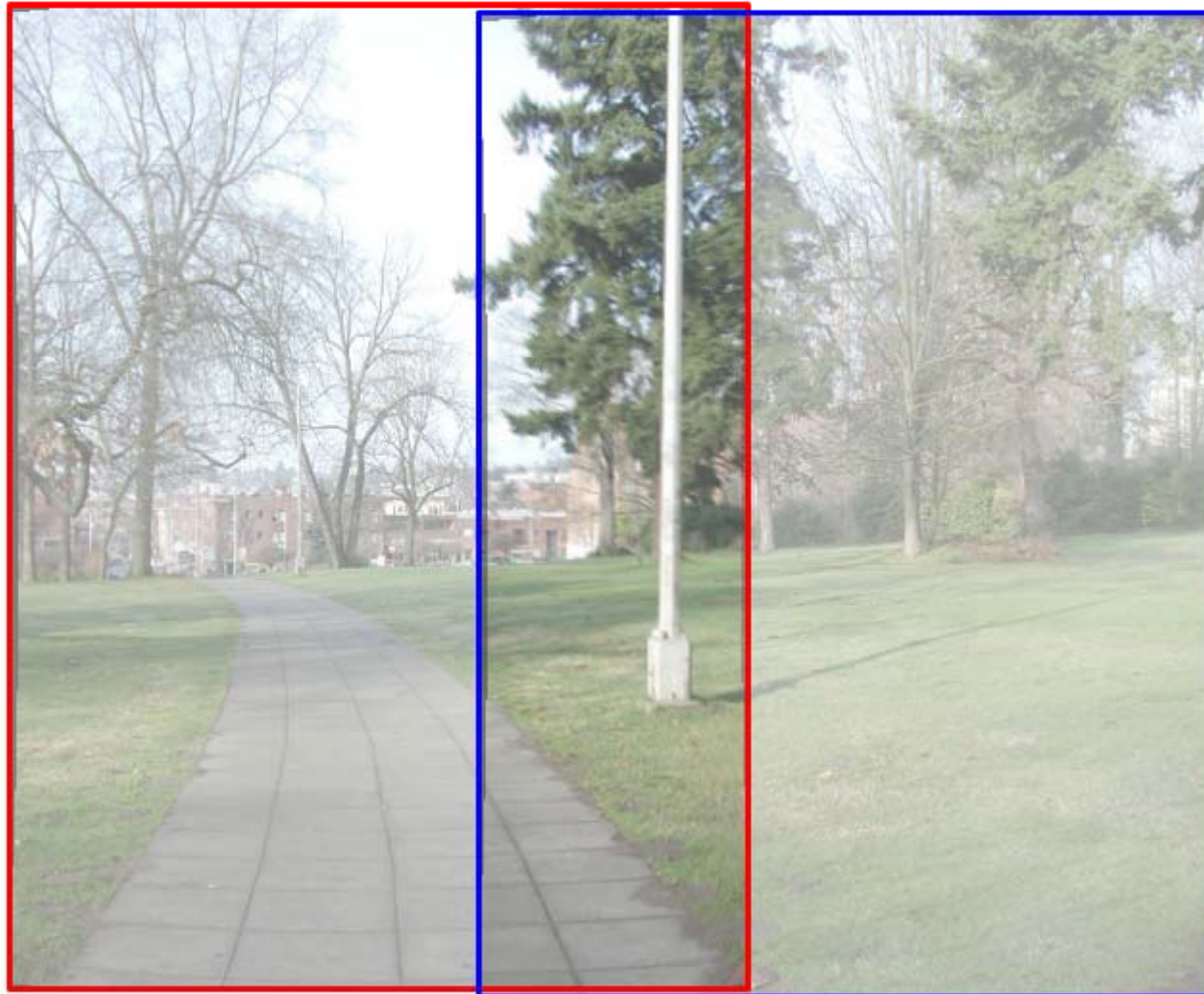
# Blending

---

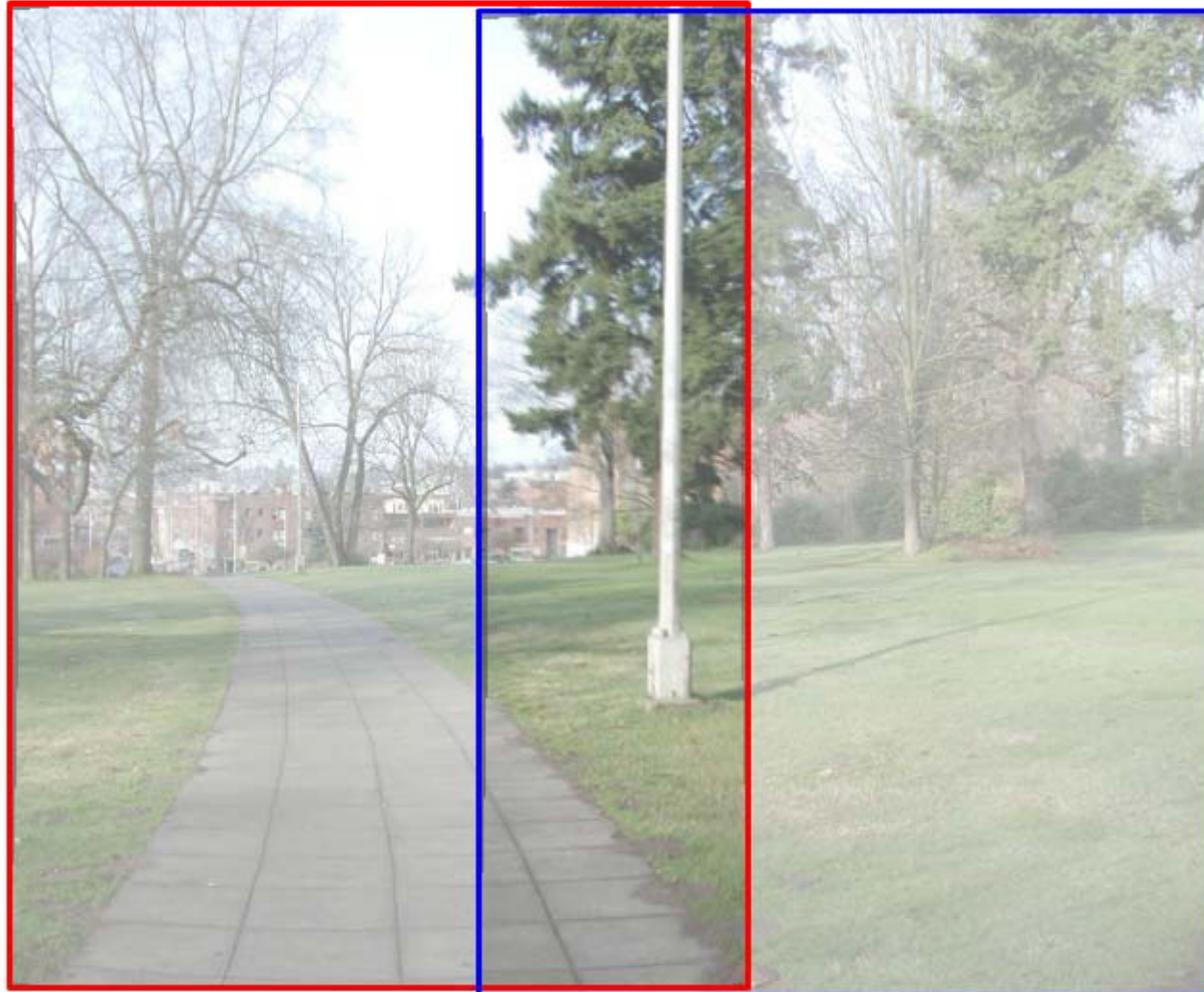
- Why blending: parallax, lens distortion, scene motion, exposure difference

# Blending

---



# Blending



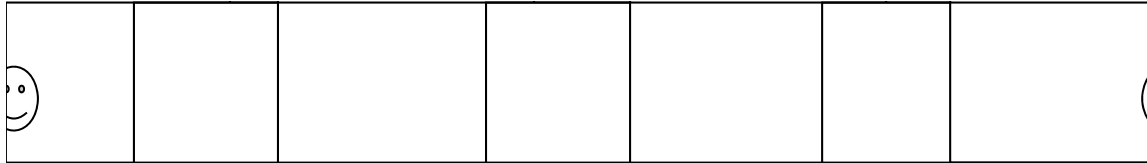
# Blending

---



# Assembling the panorama

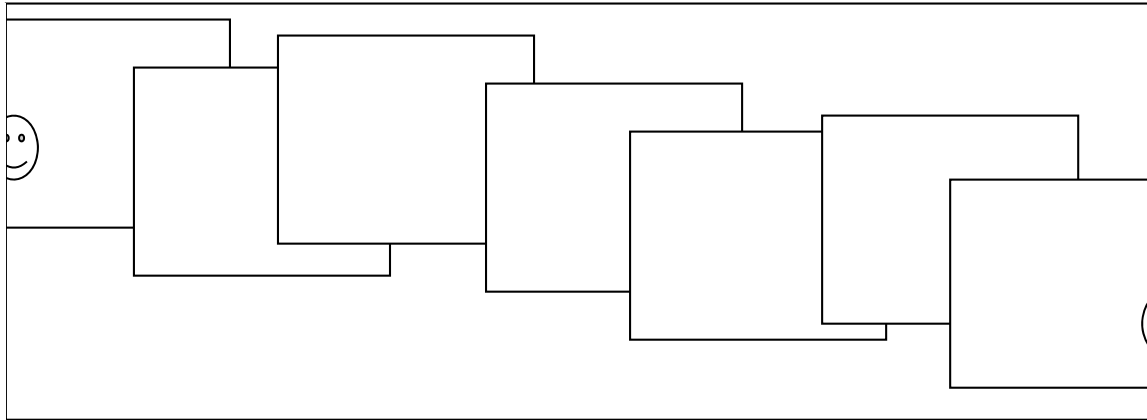
---



- Stitch pairs together, blend, then crop

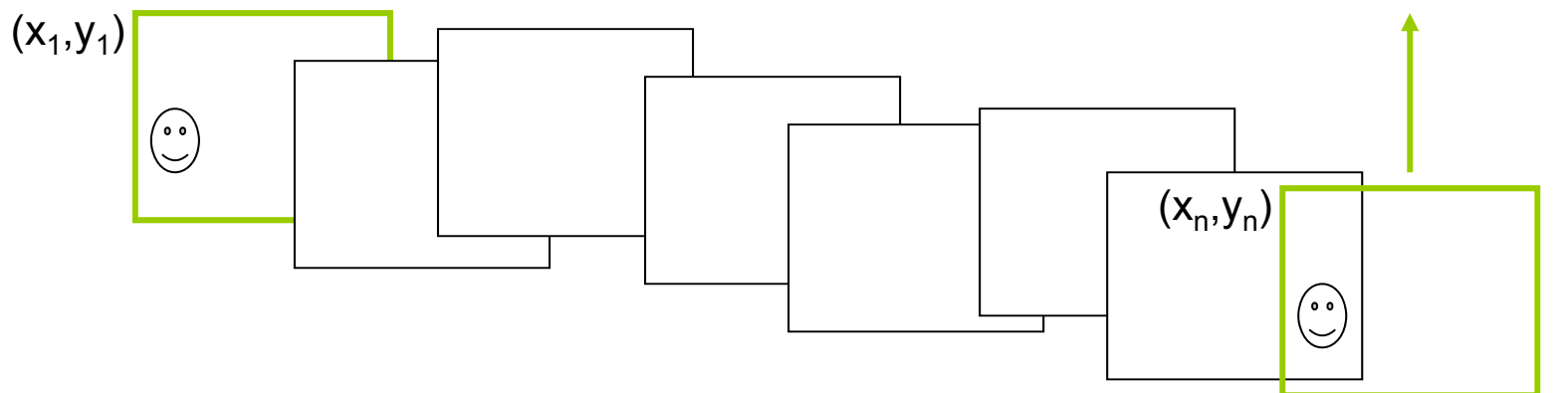
# Problem: Drift

---



- Error accumulation
  - small errors accumulate over time

# Problem: Drift



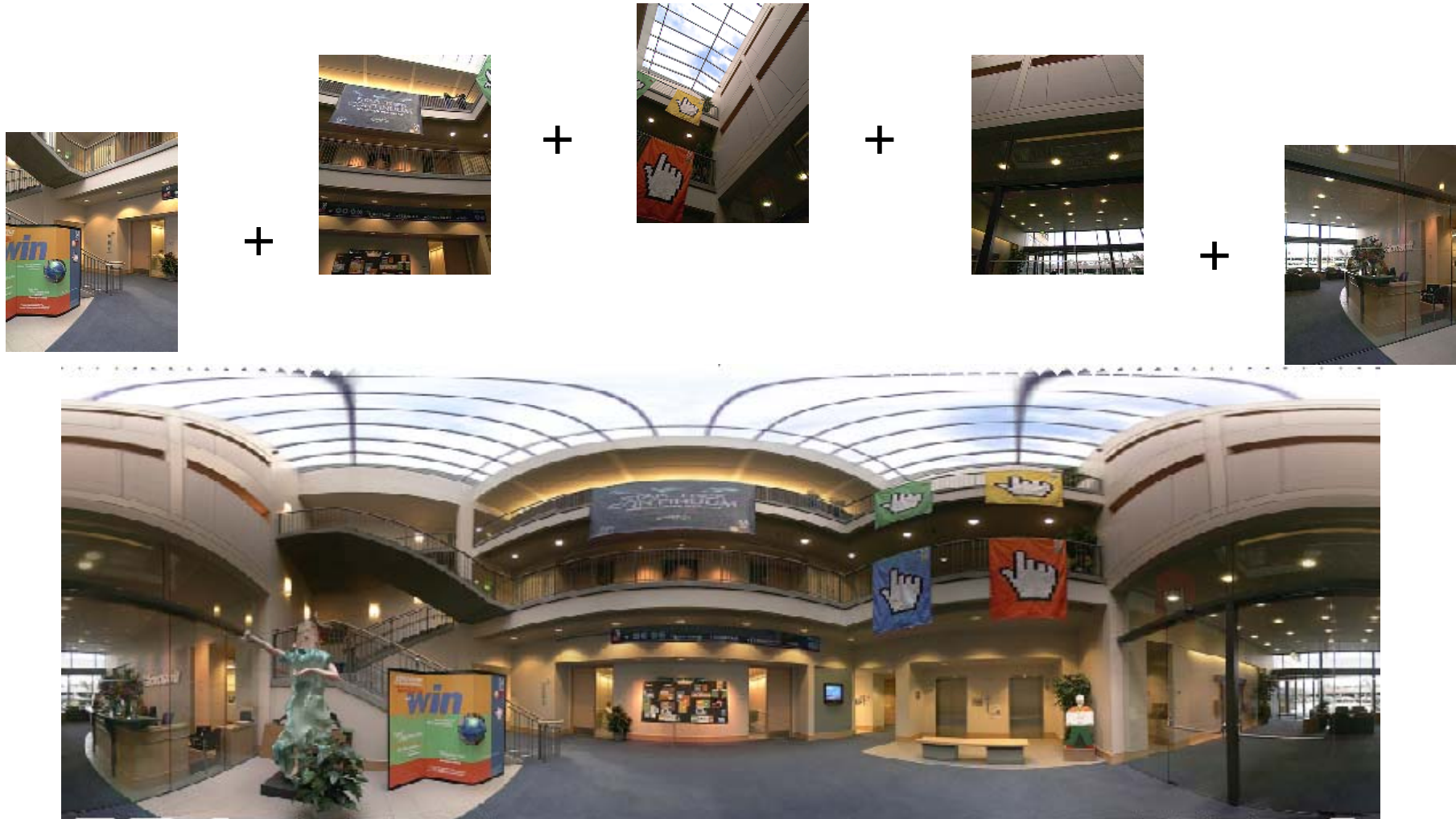
- Solution
  - add another copy of first image at the end
  - there are a bunch of ways to solve this problem
    - add displacement of  $(y_1 - y_n) / (n - 1)$  to each image after the first
    - compute a global warp:  $y' = y + ax$
    - run a big optimization problem, incorporating this constraint
      - best solution, but more complicated
      - known as “bundle adjustment”

# End-to-end alignment and crop

---



# Viewer: panorama



example: <http://www.cs.washington.edu/education/courses/cse590ss/01wi/projects/project1/students/dougz/index.html>

# Viewer: texture mapped model

---



example: <http://www.panoramas.dk/>

# Cylindrical panorama

---

1. Take pictures on a tripod (or handheld)
2. Warp to cylindrical coordinate
3. Compute pairwise alignments
4. Fix up the end-to-end alignment
5. Blending
6. Crop the result and import into a viewer

# Determine pairwise alignment?

---

- Feature-based methods: only use feature points to estimate parameters
- We will study the “Recognising panorama” paper published in ICCV 2003
- Run SIFT (or other feature algorithms) for each image, find feature matches.

# Determine pairwise alignment

---

- $p' = Mp$ , where  $M$  is a transformation matrix,  $p$  and  $p'$  are feature matches
- It is possible to use more complicated models such as affine or perspective
- For example, assume  $M$  is a 2x2 matrix

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

- Find  $M$  with the least square error

$$\sum_{i=1}^n (Mp - p')^2$$

# Determine pairwise alignment

---

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad \begin{aligned} x_1 m_{11} + y_1 m_{12} &= x'_1 \\ x_1 m_{21} + y_1 m_{22} &= y'_1 \end{aligned}$$

- Overdetermined system

$$\begin{pmatrix} x_1 & y_1 & 0 & 0 \\ 0 & 0 & x_1 & y_1 \\ x_2 & y_2 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 0 & 0 \\ 0 & 0 & x_n & y_n \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{21} \\ m_{22} \end{pmatrix} = \begin{pmatrix} x'_1 \\ y'_1 \\ x'_2 \\ \vdots \\ x'_n \\ y'_n \end{pmatrix}$$

# Normal equation

---

Given an overdetermined system

$$\mathbf{Ax} = \mathbf{b}$$

the normal equation is that which minimizes the sum of the square differences between left and right sides

$$\mathbf{A}^T \mathbf{Ax} = \mathbf{A}^T \mathbf{b}$$

Why?

# Normal equation

---

$$E(\mathbf{x}) = (\mathbf{Ax} - \mathbf{b})^2$$

$$\begin{bmatrix} a_{11} & \dots & a_{1m} \\ \vdots & & \vdots \\ \vdots & & \vdots \\ \vdots & & \vdots \\ a_{n1} & \dots & a_{nm} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ \vdots \\ b_n \end{bmatrix}$$

$n \times m$ ,  $n$  equations,  $m$  variables

# Normal equation

---

$$\mathbf{Ax} - \mathbf{b} = \begin{bmatrix} \sum_{j=1}^m a_{1j} x_j \\ \vdots \\ \sum_{j=1}^m a_{ij} x_j \\ \vdots \\ \sum_{j=1}^m a_{nj} x_j \end{bmatrix} - \begin{bmatrix} b_1 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{bmatrix} = \begin{bmatrix} \left( \sum_{j=1}^m a_{1j} x_j \right) - b_1 \\ \vdots \\ \left( \sum_{j=1}^m a_{ij} x_j \right) - b_i \\ \vdots \\ \left( \sum_{j=1}^m a_{nj} x_j \right) - b_n \end{bmatrix}$$
$$E(\mathbf{x}) = (\mathbf{Ax} - \mathbf{b})^2 = \sum_{i=1}^n \left[ \left( \sum_{j=1}^m a_{ij} x_j \right) - b_i \right]^2$$

# Normal equation

---

$$E(\mathbf{x}) = (\mathbf{Ax} - \mathbf{b})^2 = \sum_{i=1}^n \left[ \left( \sum_{j=1}^m a_{ij} x_j \right) - b_i \right]^2$$

$$\begin{aligned} 0 = \frac{\partial E}{\partial x_1} &= \sum_{i=1}^n 2 \left[ \left( \sum_{j=1}^m a_{ij} x_j \right) - b_i \right] a_{i1} \\ &= 2 \sum_{i=1}^n a_{i1} \sum_{j=1}^m a_{ij} x_j - 2 \sum_{i=1}^n a_{i1} b_i \end{aligned}$$

$$0 = \frac{\partial E}{\partial \mathbf{x}} = 2(\mathbf{A}^T \mathbf{Ax} - \mathbf{A}^T \mathbf{b}) \rightarrow \mathbf{A}^T \mathbf{Ax} = \mathbf{A}^T \mathbf{b}$$

# Normal equation

---

$$(\mathbf{Ax} - \mathbf{b})^2$$

$$= (\mathbf{Ax} - \mathbf{b})^T (\mathbf{Ax} - \mathbf{b})$$

$$= ((\mathbf{Ax})^T - \mathbf{b}^T)(\mathbf{Ax} - \mathbf{b})$$

$$= (\mathbf{x}^T \mathbf{A}^T - \mathbf{b}^T)(\mathbf{Ax} - \mathbf{b})$$

$$= \mathbf{x}^T \mathbf{A}^T \mathbf{Ax} - \mathbf{b}^T \mathbf{Ax} - \mathbf{x}^T \mathbf{A}^T \mathbf{b} + \mathbf{b}^T \mathbf{b}$$

$$= \mathbf{x}^T \mathbf{A}^T \mathbf{Ax} - (\mathbf{A}^T \mathbf{b})^T \mathbf{x} - (\mathbf{A}^T \mathbf{b})^T \mathbf{x} + \mathbf{b}^T \mathbf{b}$$

$$\frac{\partial E}{\partial \mathbf{x}} = 2\mathbf{A}^T \mathbf{Ax} - 2\mathbf{A}^T \mathbf{b}$$

# Determine pairwise alignment

---

- $p' = Mp$ , where  $M$  is a transformation matrix,  $p$  and  $p'$  are feature matches
- For translation model, it is easier.

$$E = \sum_{i=1}^n \left[ (m_1 + x_i - x'_i)^2 + (m_2 + y_i - y'_i)^2 \right]$$

$$0 = \frac{\partial E}{\partial m_1}$$

- What if the match is false? Avoid impact of outliers.

# RANSAC

---

- RANSAC = Random Sample Consensus
- An algorithm for robust fitting of models in the presence of many data outliers
- Compare to robust statistics
- Given  $N$  data points  $x_i$ , assume that majority of them are generated from a model with parameters  $\Theta$ , try to recover  $\Theta$ .

# RANSAC algorithm

---

Run  $k$  times:

← How many times?

(1) draw  $n$  samples randomly

← How big?

Smaller is better

(2) fit parameters  $\Theta$  with these  $n$  samples

(3) for each of other  $N-n$  points, calculate its distance to the fitted model, count the number of inlier points,  $c$

Output  $\Theta$  with the largest  $c$

← How to define?

Depends on the problem.

# How to determine k

$p$ : probability of real inliers

$P$ : probability of success after k trials

$$P = 1 - (1 - p^n)^k$$

$p^n$  n samples are all inliers

a failure

failure after k trials

$$k = \frac{\log(1 - P)}{\log(1 - p^n)}$$

for  $P=0.99$

| $n$ | $p$ | $k$ |
|-----|-----|-----|
| 3   | 0.5 | 35  |
| 6   | 0.6 | 97  |
| 6   | 0.5 | 293 |

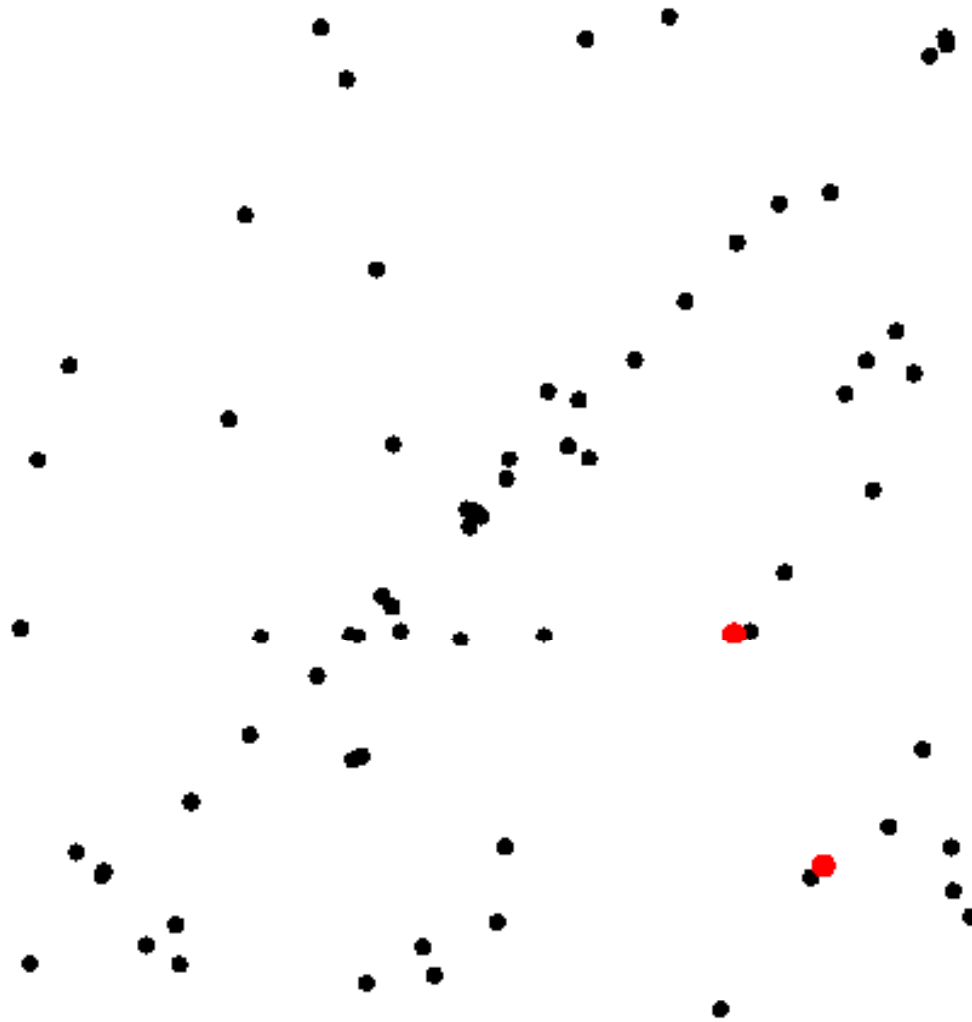
# Example: line fitting

---



# Example: line fitting

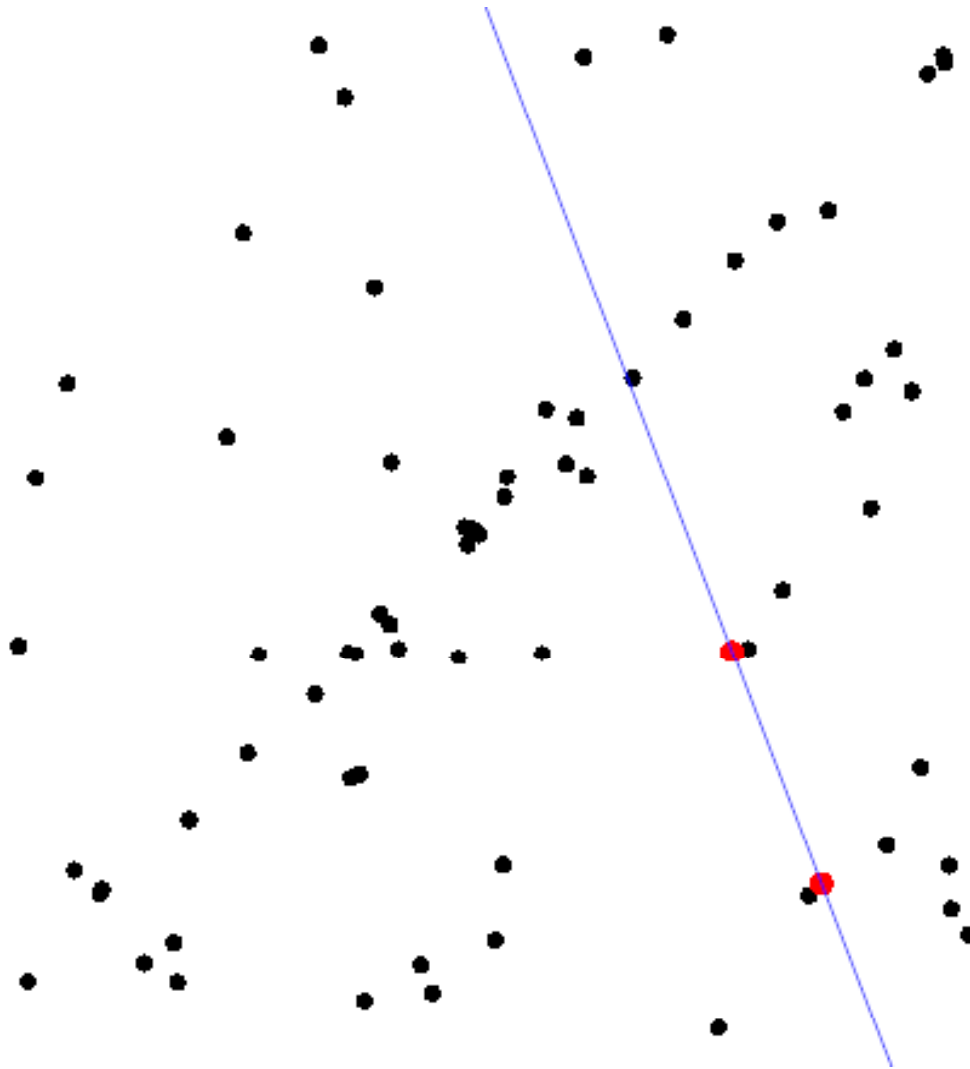
---



$n=2$

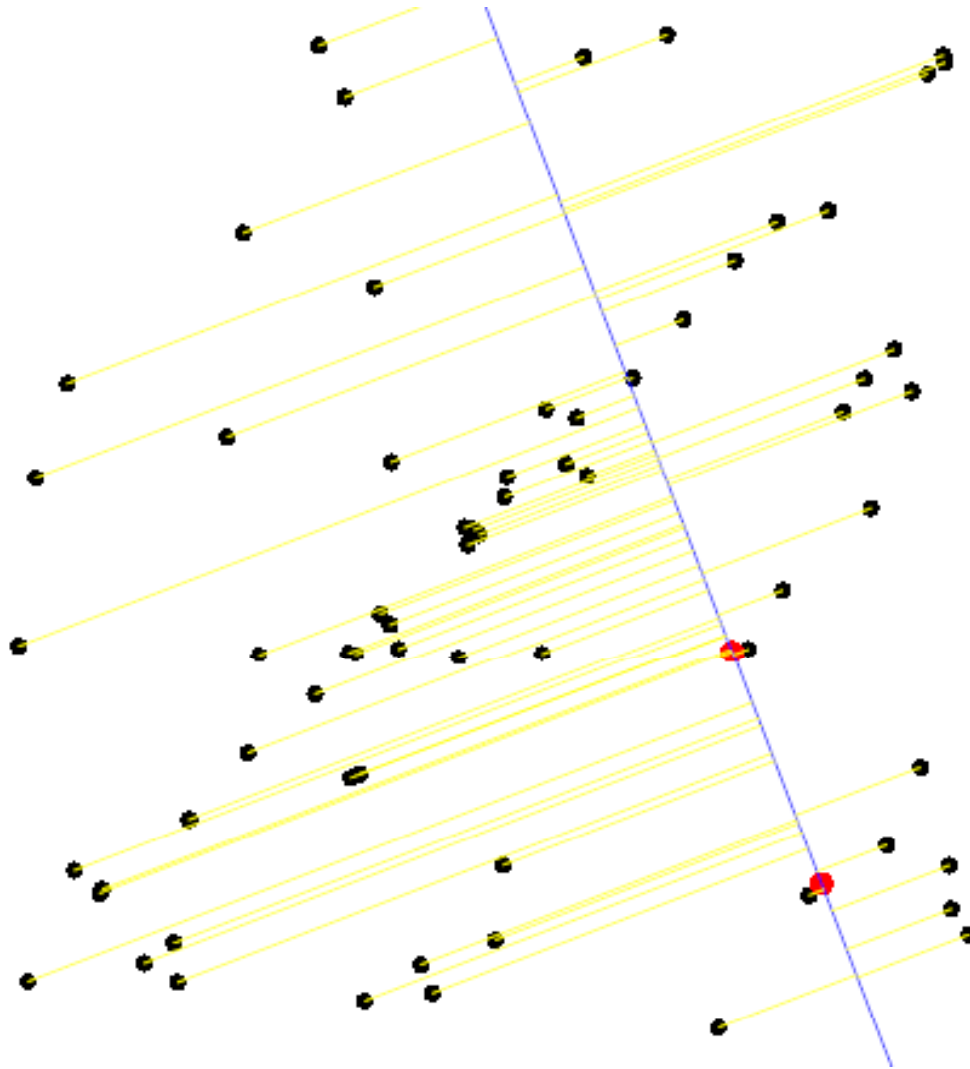
# Model fitting

---



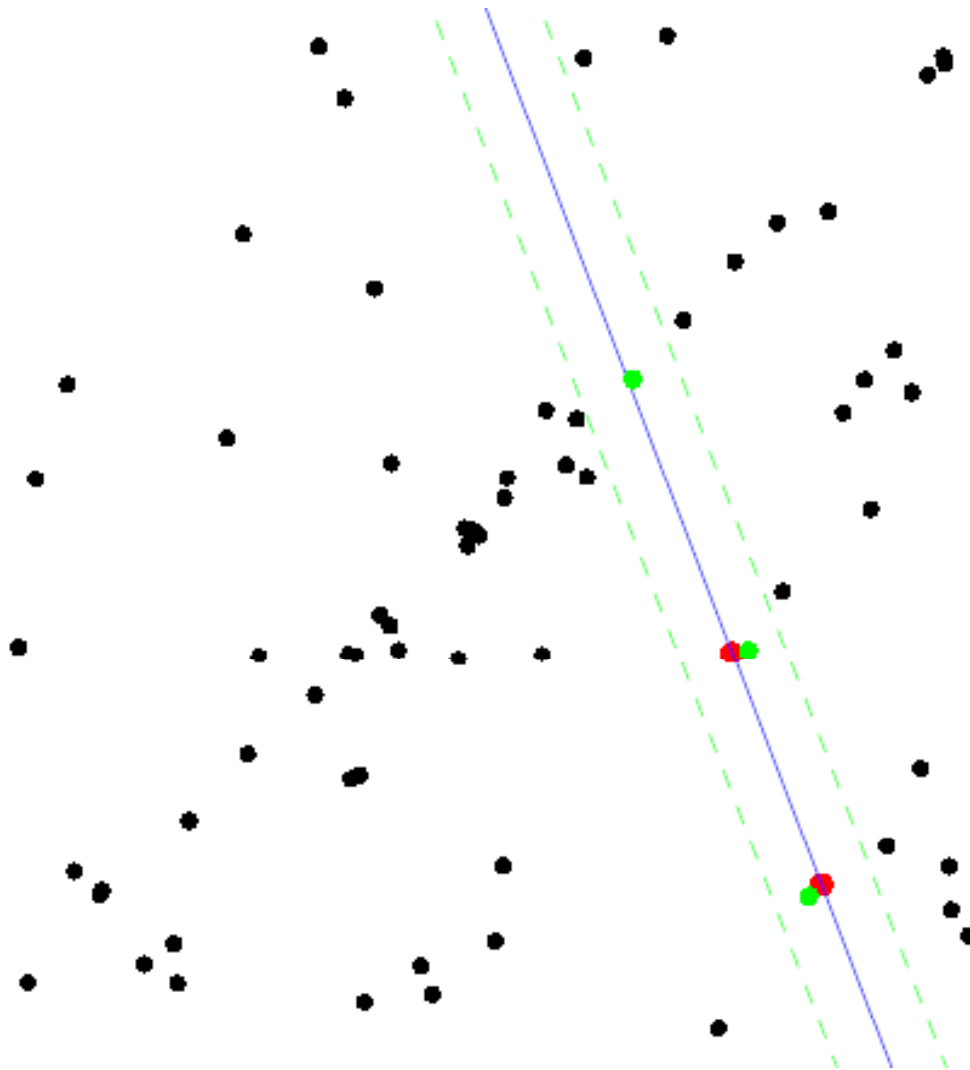
# Measure distances

---



# Count inliers

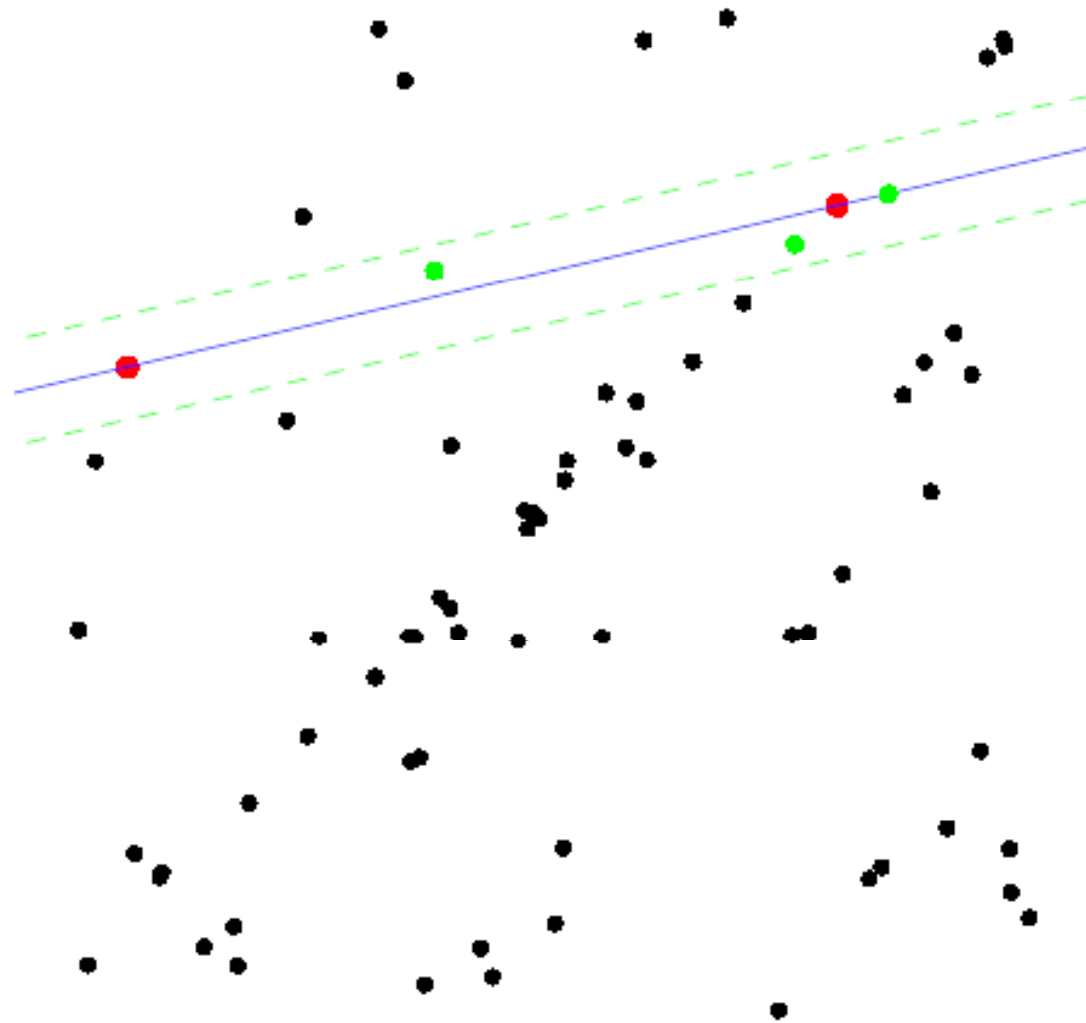
---



$c=3$

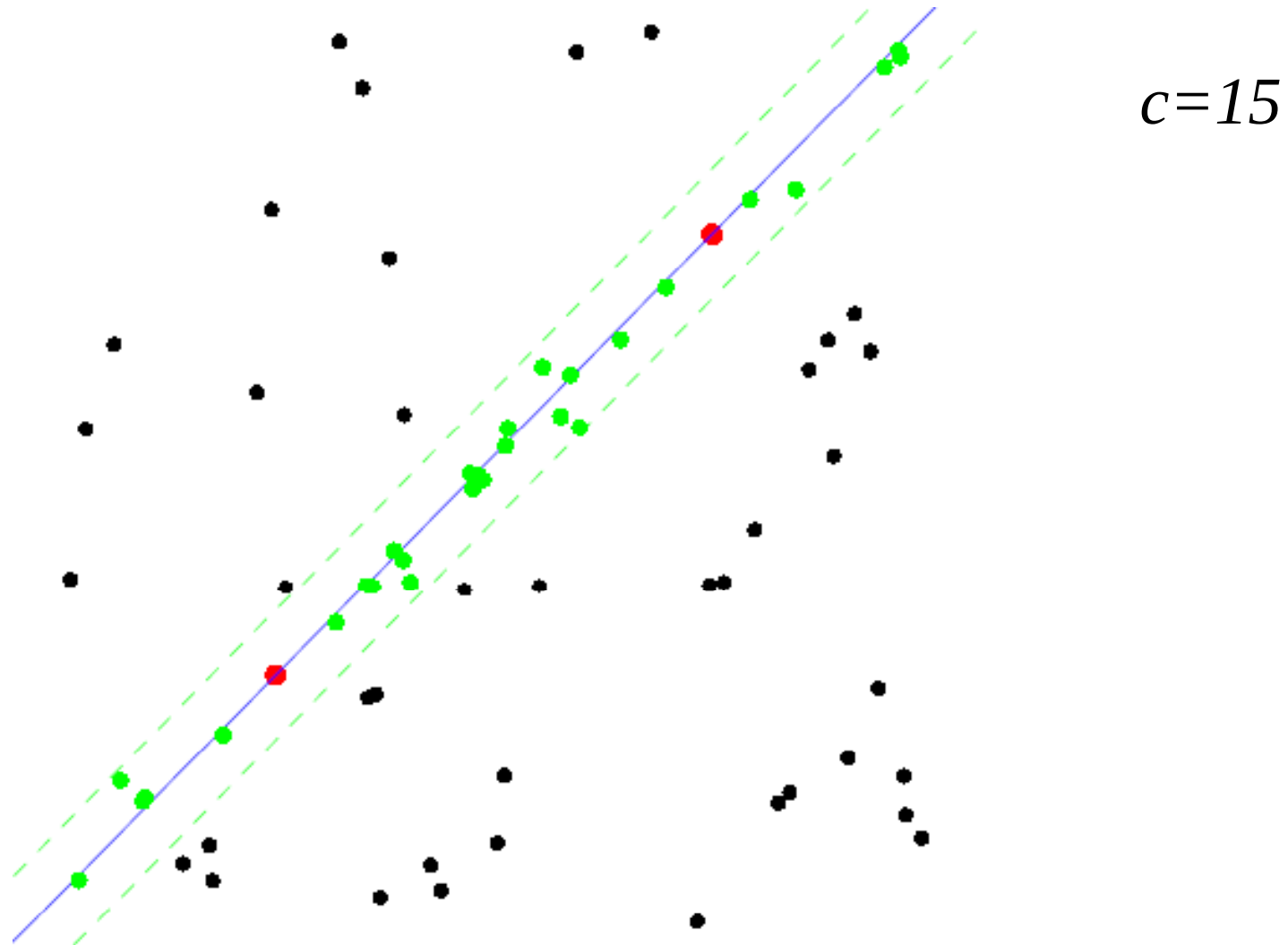
# Another trial

---

 $c=3$

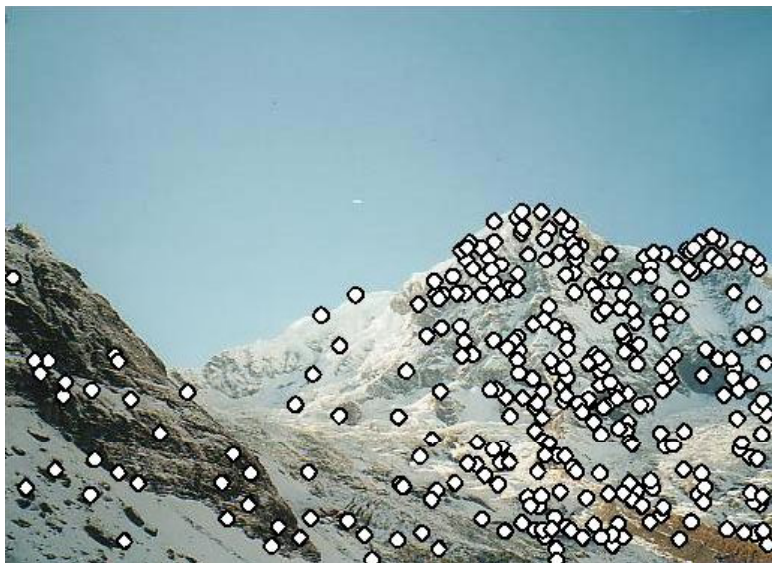
# The best model

---



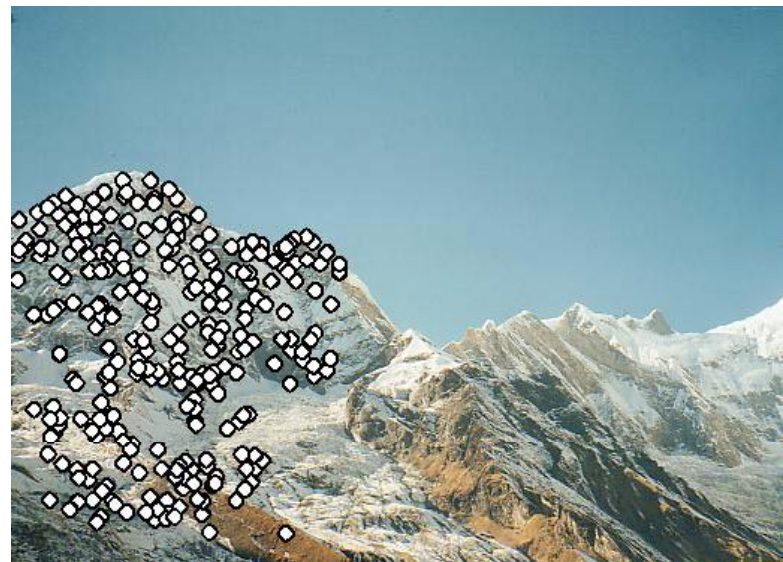
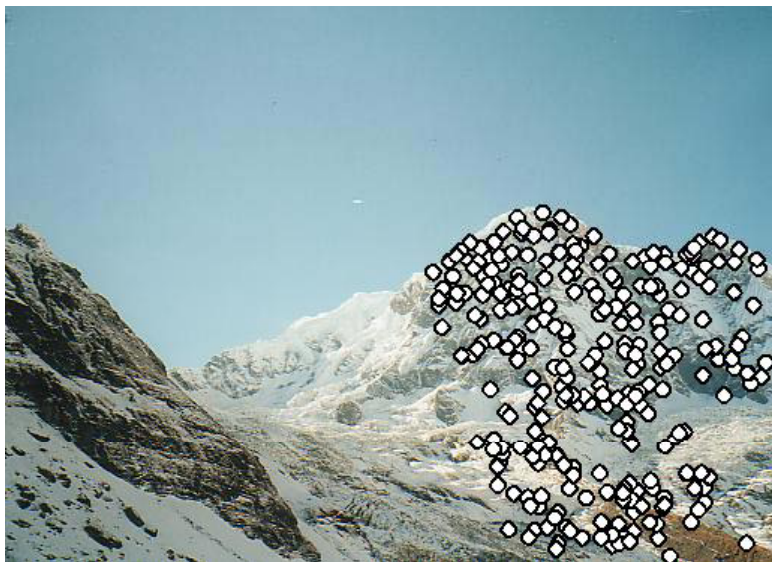
# RANSAC for Homography

---



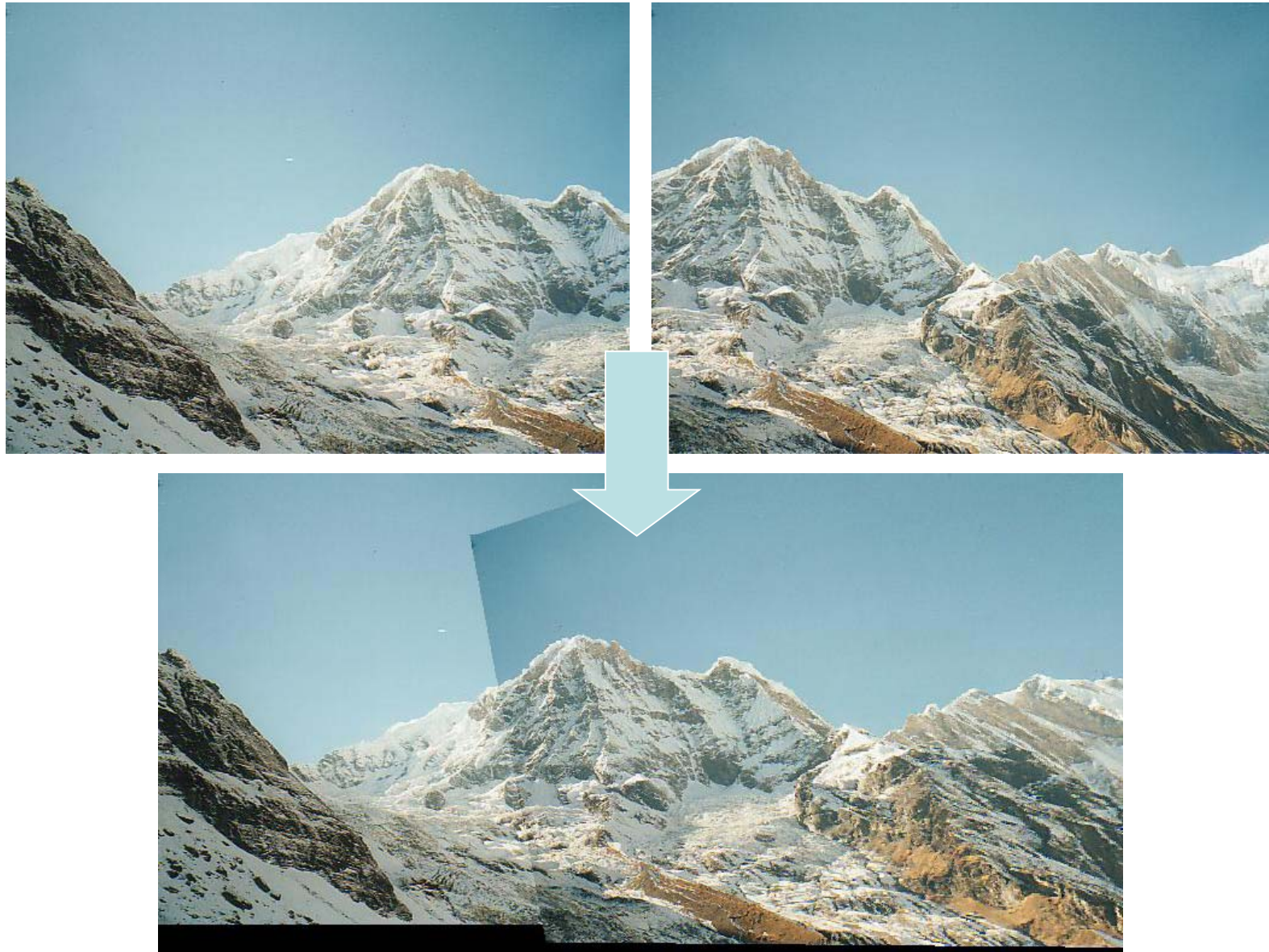
# RANSAC for Homography

---



# RANSAC for Homography

---



# Applications of panorama in VFX

---

- Background plates
- Image-based lighting

# Troy (image-based lighting)

---



[http://www.cgnetworks.com/story\\_custom.php?story\\_id=2195&page=4](http://www.cgnetworks.com/story_custom.php?story_id=2195&page=4)

# Spiderman 2 (background plate)

