Homographies and image correspondences

15-463, 15-663, 15-862
Computational Photography
Fall 2017, Lecture 19

http://graphics.cs.cmu.edu/courses/15-463

Course announcements

- Homework 5 delayed for ~~Tuesday~~ Wednesday.
  - You will need cameras for <u>the bonus part of</u> that one as well, so keep the ones you picked up for HW4.
  - Will be shorter than HW4 so that it can fit the one week deadline.
- Project proposals were due on Tuesday 31st.
  - Deadline extended by one day.
- One-to-one meetings this week.
  - Sign up for a slot using the spreadsheet posted on Piazza.
  - Make sure to read instructions on course website about elevator pitch presentation.

Overview of today's lecture

- Motivation: panoramas.
- Back to warping: image homographies.
- When can we use homographies?
- Computing with homographies.
- The image correspondence pipeline.
- Detecting interest points.

Slide credits

Most of these slides were adapted from:

- Kris Kitani (15-463, Fall 2016).
- Noah Snavely (Cornell).

Motivation for image alignment: panoramas.

How do you create a panorama?

Panorama: an image of (near) 360^o field of view.

How do you create a panorama?

Panorama: an image of (near) 360^o field of view.

1. Use a very wide-angle lens.

Wide-angle lenses

Fish-eye lens: can produce (near) hemispherical field of view.

What are the pros and cons of this?

How do you create a panorama?

Panorama: an image of (near) 360^o field of view.

1. Use a very wide-angle lens.
- Pros: Everything is done optically, single capture.
- Cons: Lens is super expensive and bulky, lots of distortion (can be dealt-with in post).

Any alternative to this?

How do you create a panorama?

Panorama: an image of (near) 360^{o} field of view.

1. Use a very wide-angle lens.
- Pros: Everything is done optically, single capture.
- Cons: Lens is super expensive and bulky, lots of distortion (can be dealt-with in post).

2. Capture multiple images and combine them.

Panoramas from image stitching

1. Capture multiple images from different viewpoints.

2. Stitch them together into a virtual wide-angle image.

Panoramas from image stitching

1. Capture multiple images from different viewpoints.

2. Stitch them together into a virtual wide-angle image.

How do we stitch images from different viewpoints?

Will standard stitching work?

1. Translate one image relative to another.
2. (Optionally) find an optimal seam.

How do we stitch images from different viewpoints?

Will standard stitching work?

1. Translate one image relative to another.
2. (Optionally) find an optimal seam.

left on top

right on top

Translation-only stitching is not enough to mosaic these images.

How do we stitch images from different viewpoints?

What else can we try?

How do we stitch images from different viewpoints?

Use image warping.

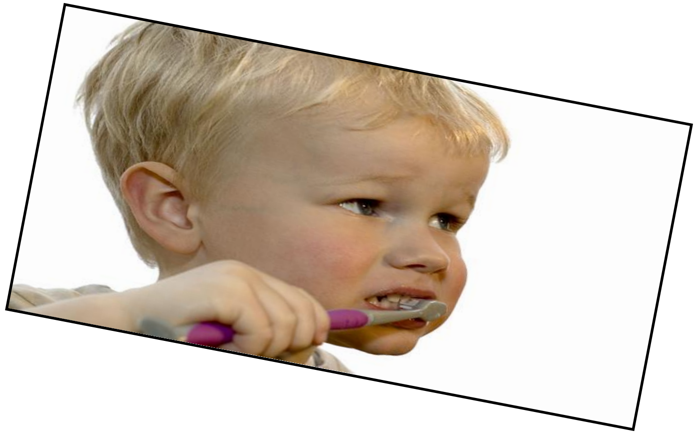
Back to warping: image homographies

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

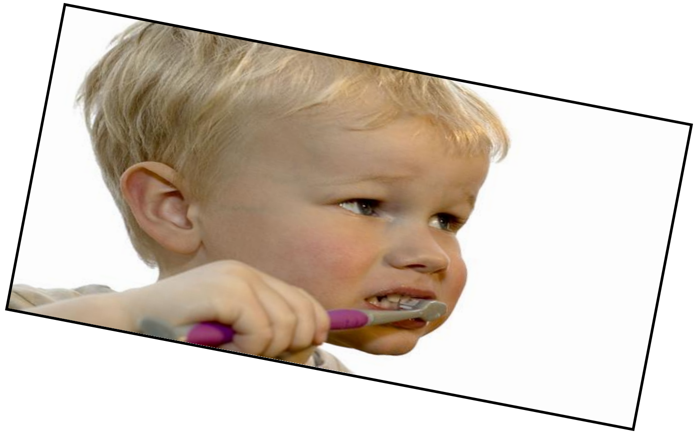
changes *range* of image function

F

Warping

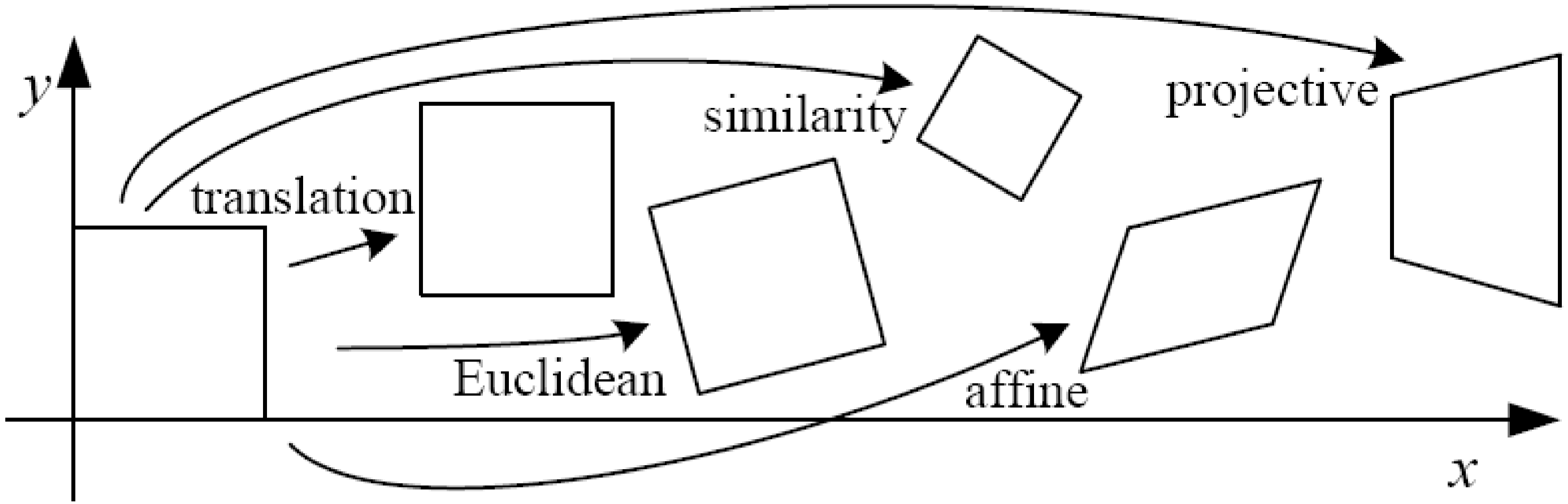
$$G(\boldsymbol{x}) = F(h\{\boldsymbol{x}\})$$

G



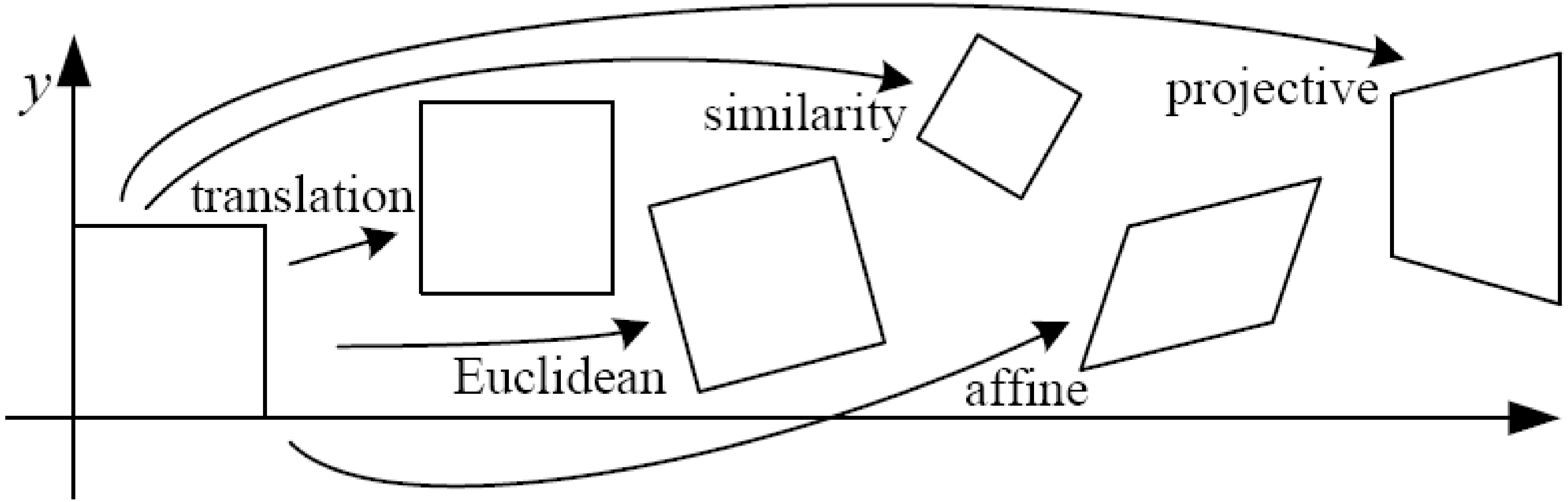
changes *domain* of image function

Classification of 2D transformations

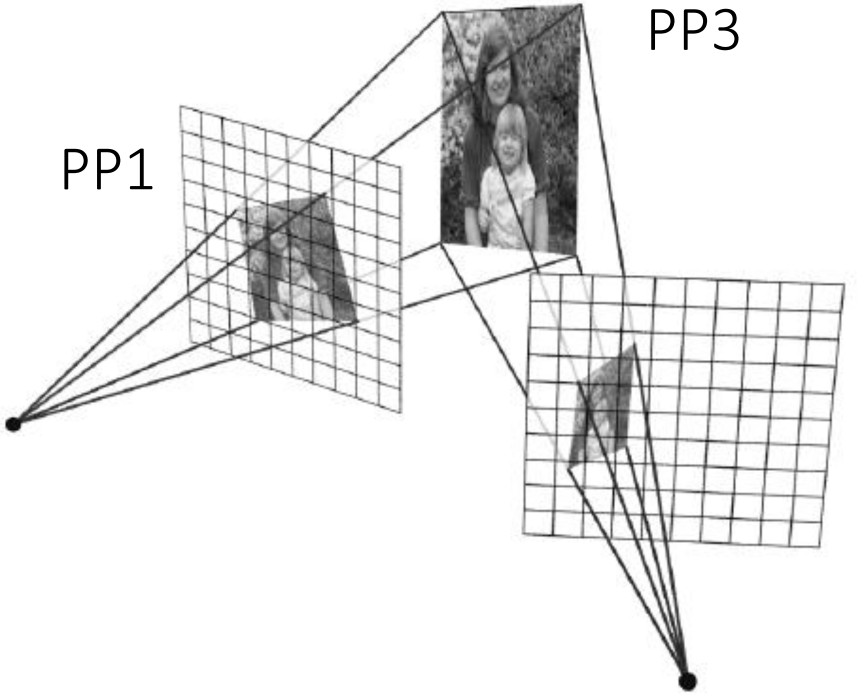


Name	Matrix	# D.O.F.
translation	$\left[\, \boldsymbol{I} \mid \boldsymbol{t} \,\right]_{2\times 3}$	2
rigid (Euclidean)	$\left[\, \boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	3
similarity	$\left[\, s\boldsymbol{R} \mid \boldsymbol{t} \,\right]_{2\times 3}$	4
affine	$\left[\, \boldsymbol{A} \,\right]_{2\times 3}$	6
projective	$\left[\, \tilde{\boldsymbol{H}} \,\right]_{3\times 3}$	8

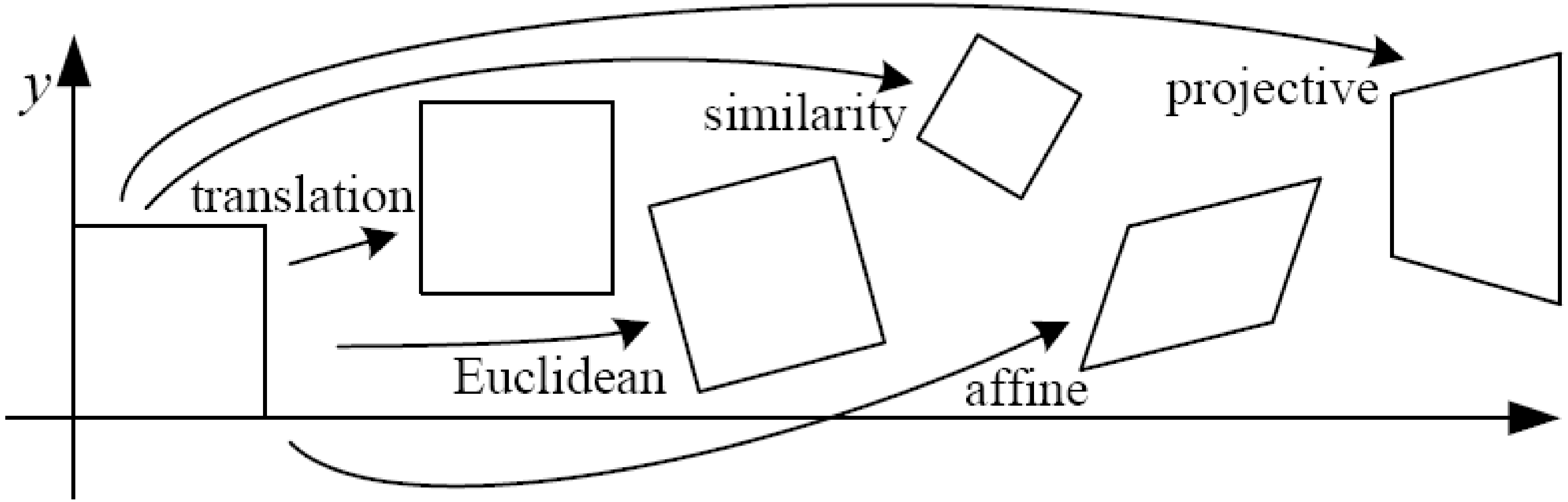
Classification of 2D transformations



Which kind transformation is needed to warp projective plane 1 into projective plane 2?

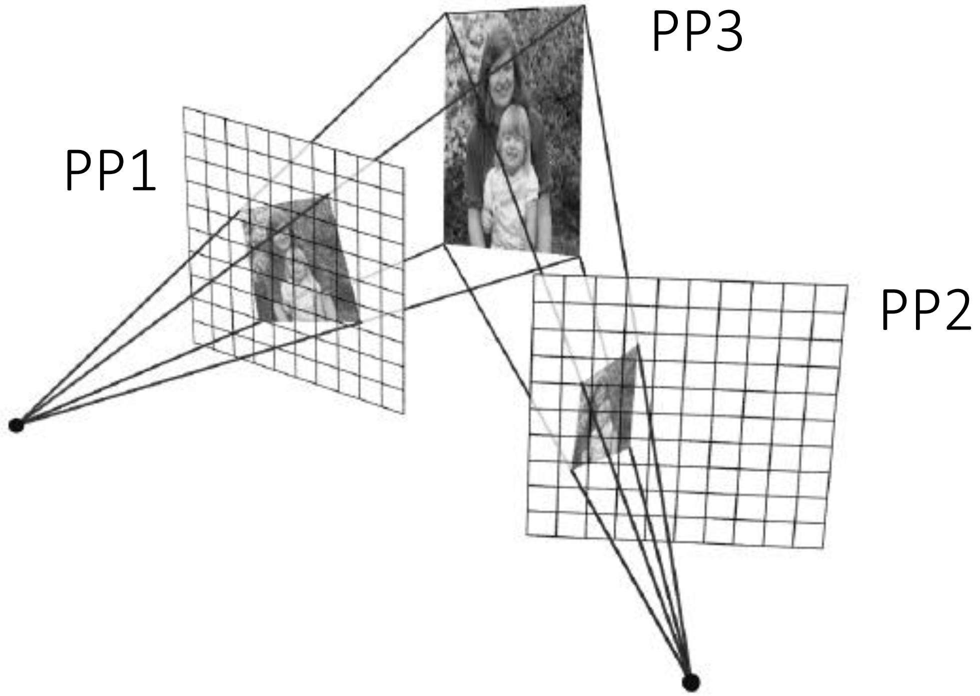


Classification of 2D transformations



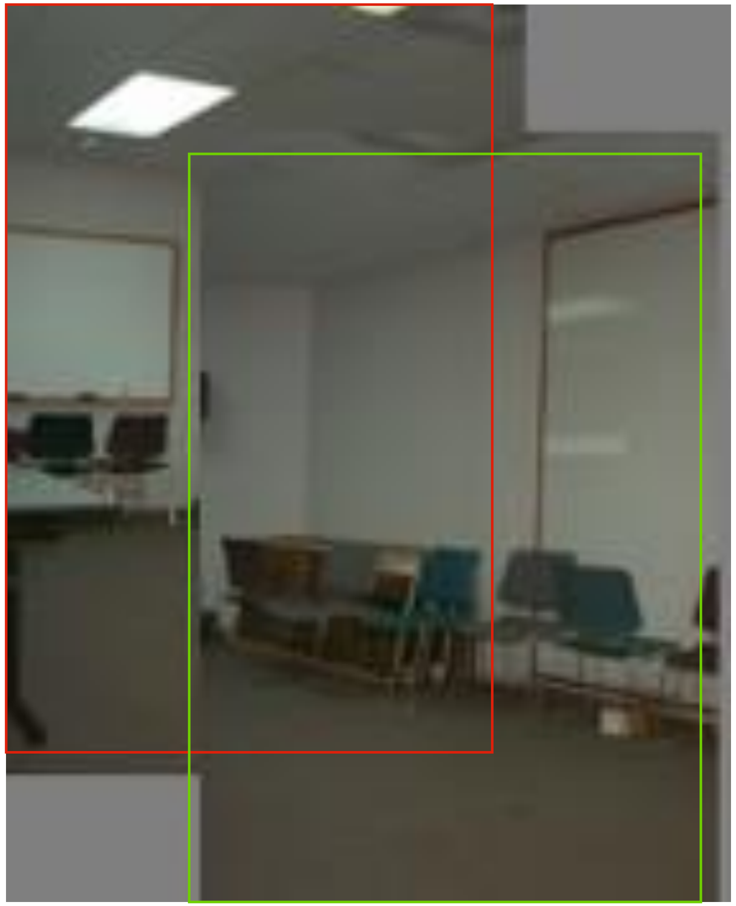
Which kind transformation is needed to warp projective plane 1 into projective plane 2?

- A projective transformation (a.k.a. a homography).

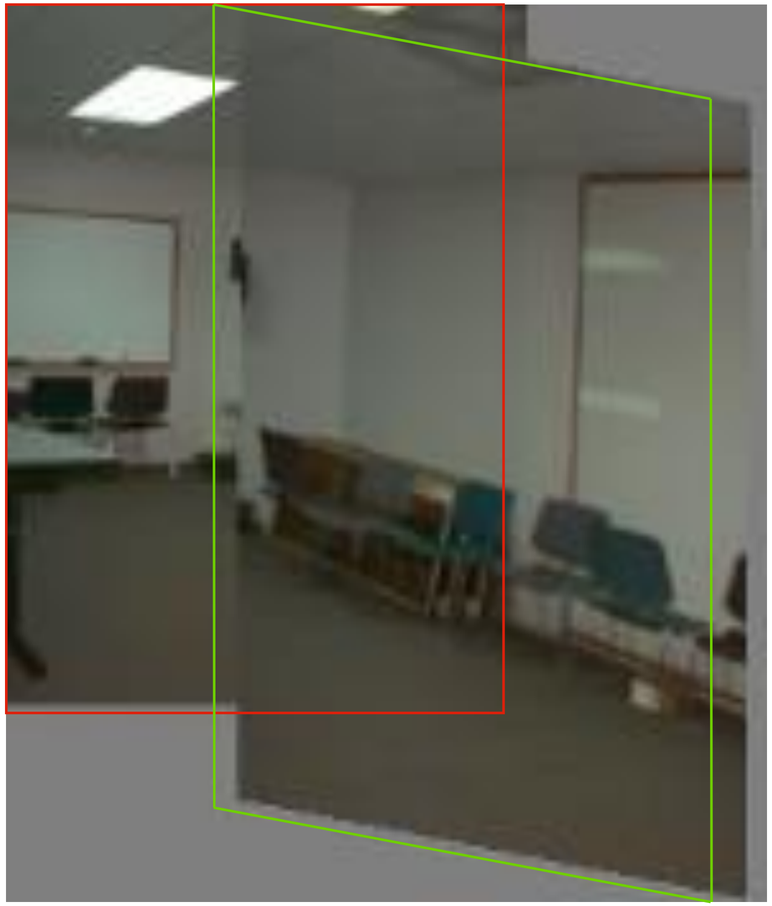


Warping with different transformations

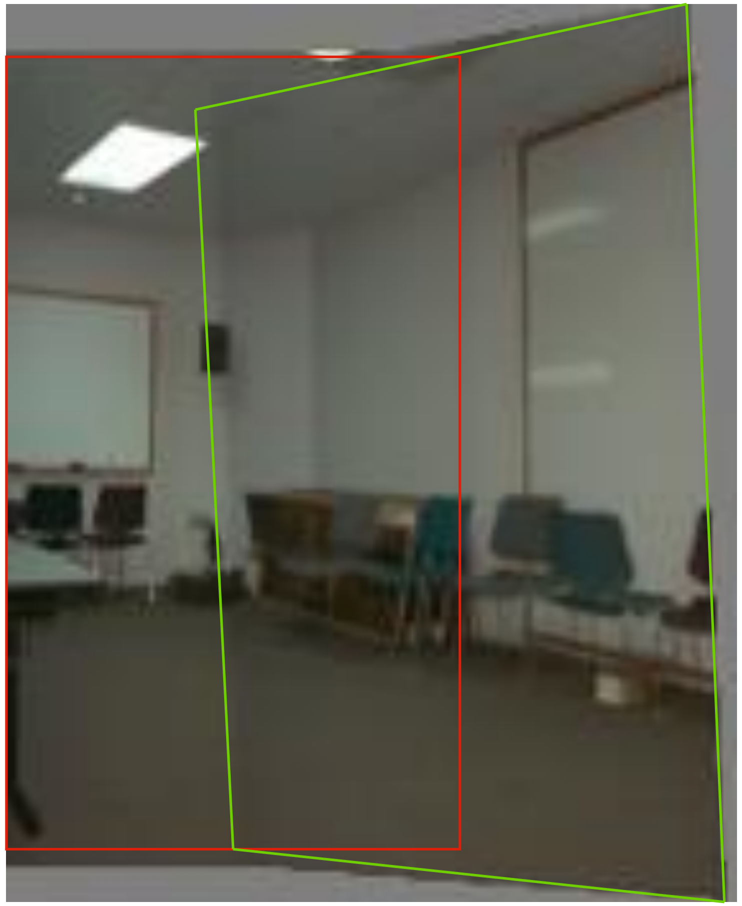
translation



affine



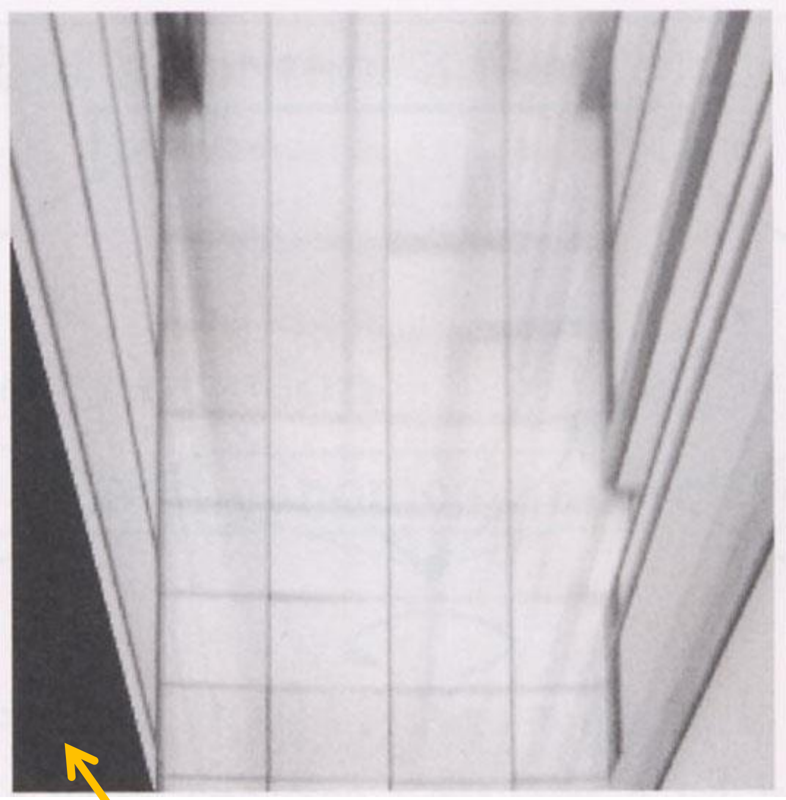
pProjective (homography)



View warping

original view

synthetic top view



synthetic side view

What are these black areas near the boundaries?

Virtual camera rotations

original view

synthetic rotations

Image rectification

two original images

rectified and stitched

Street art

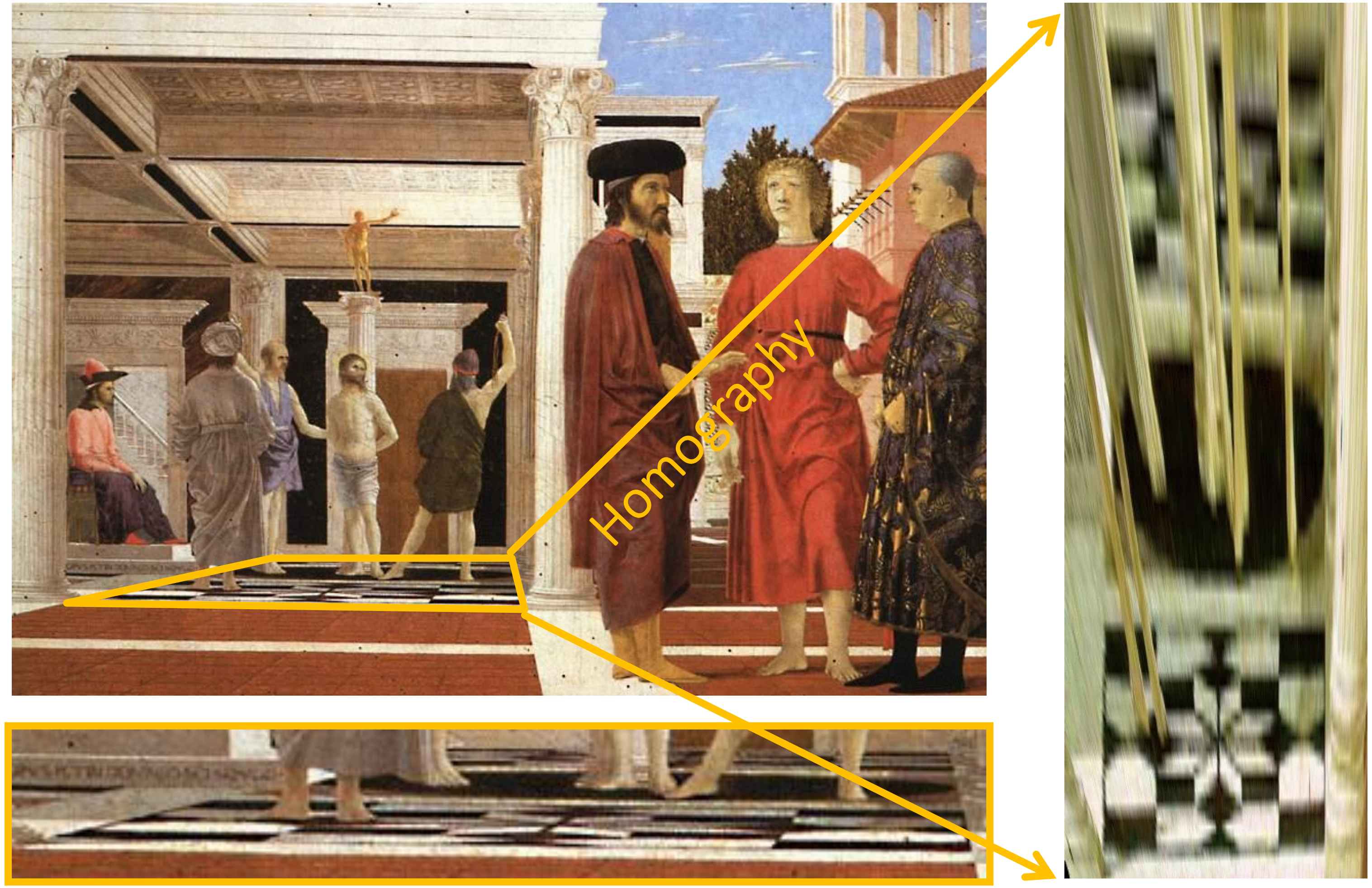
Understanding geometric patterns

What is the pattern on the floor?

magnified view of floor

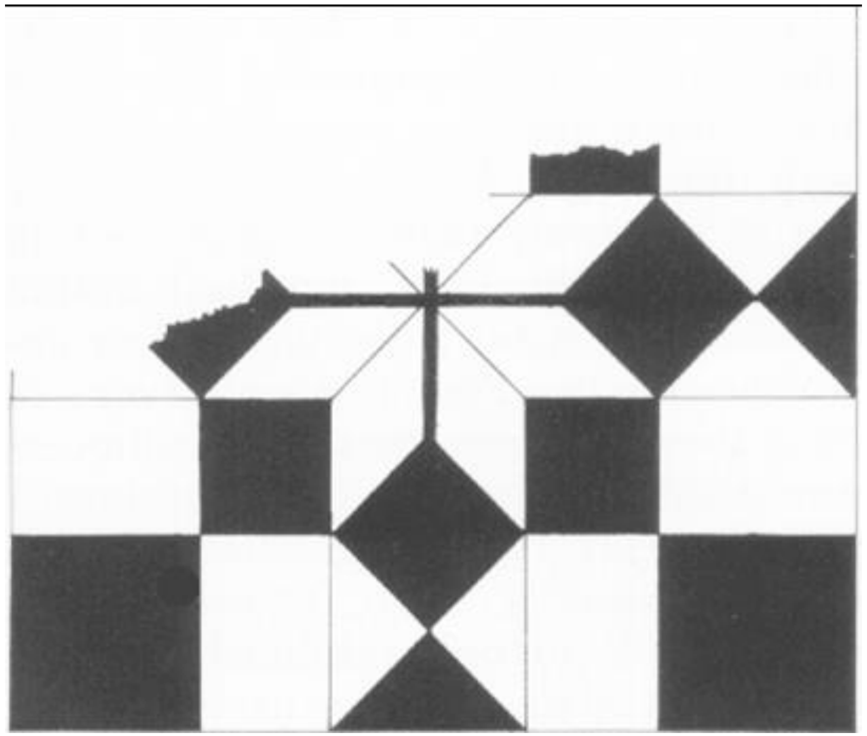
Understanding geometric patterns

What is the pattern on the floor?



magnified view of floor

rectified view

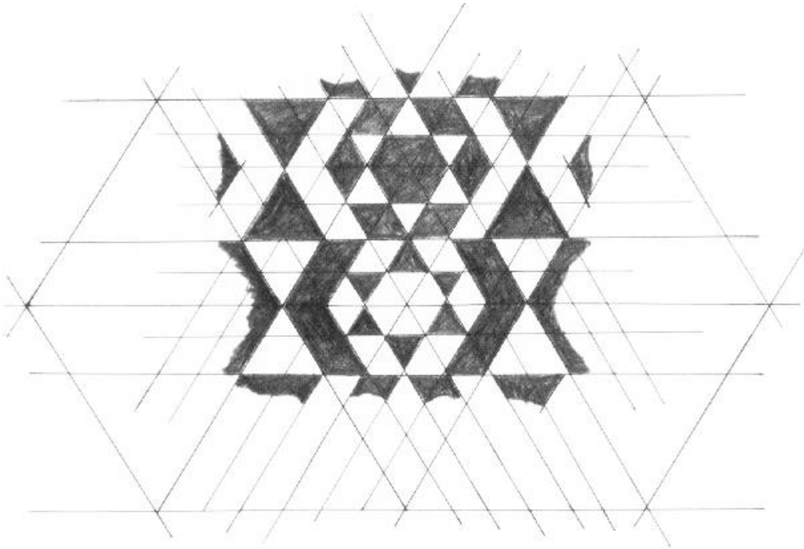


reconstruction from rectified view

Understanding geometric patterns

Very popular in renaissance drawings (when perspective was discovered)

rectified view of floor



reconstruction

A weird drawing

Holbein, "The Ambassadors"

A weird drawing

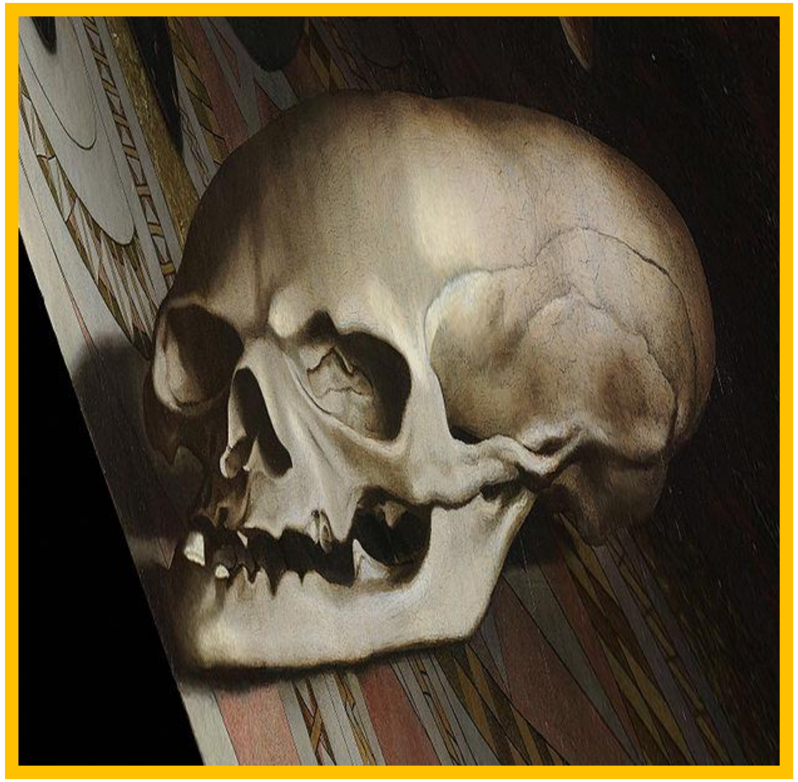
Holbein, “The Ambassadors”

What’s this???

A weird drawing

Holbein, “The Ambassadors”

skull under anamorphic perspective



rectified view

A weird drawing

Holbein, “The Ambassadors”

DIY: use a polished spoon to see the skull

Panoramas from image stitching

1. Capture multiple images from different viewpoints.

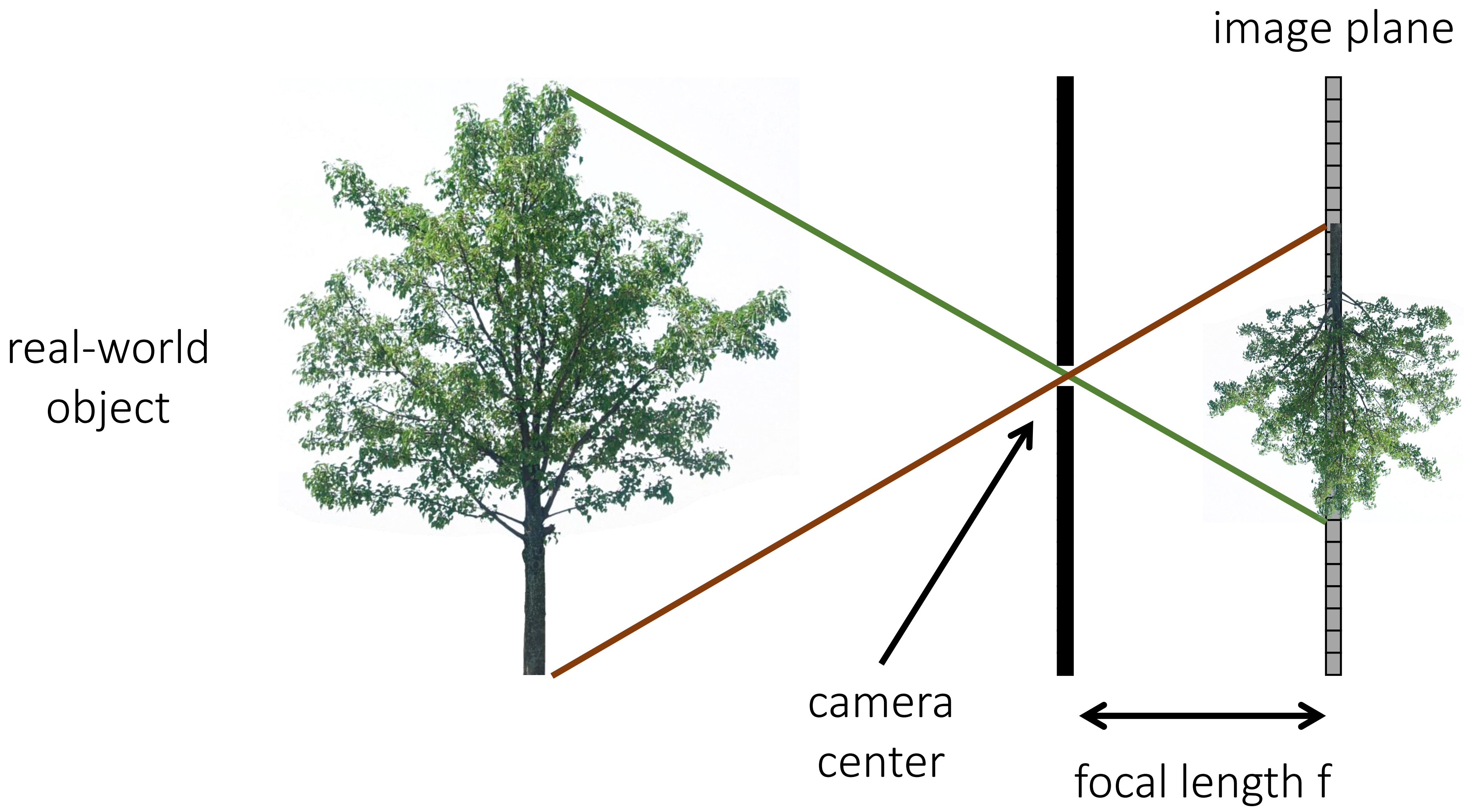
2. Stitch them together into a virtual wide-angle image.

When can we use homographies?

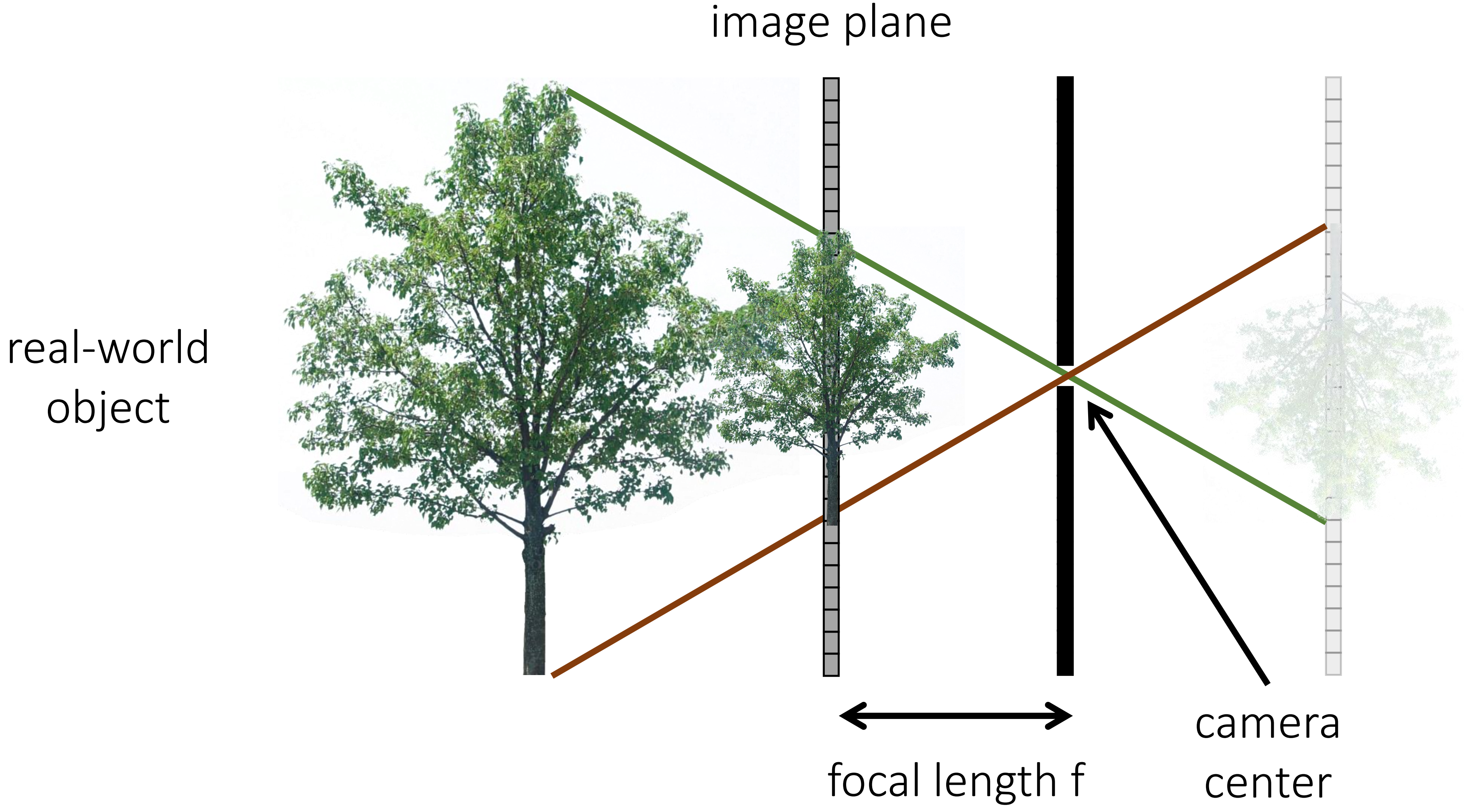
When does this work?

Use image warping.

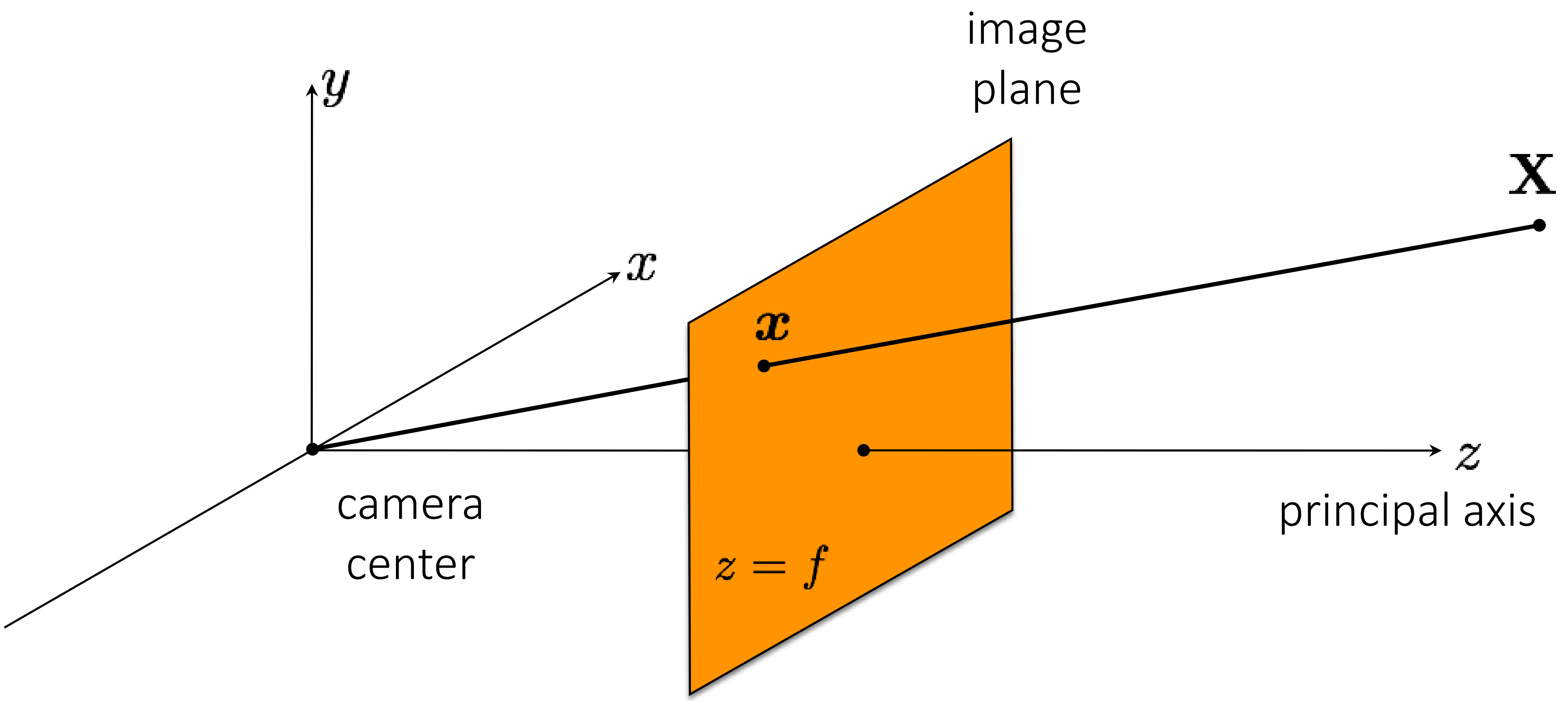
The pinhole camera



The (rearranged) pinhole camera



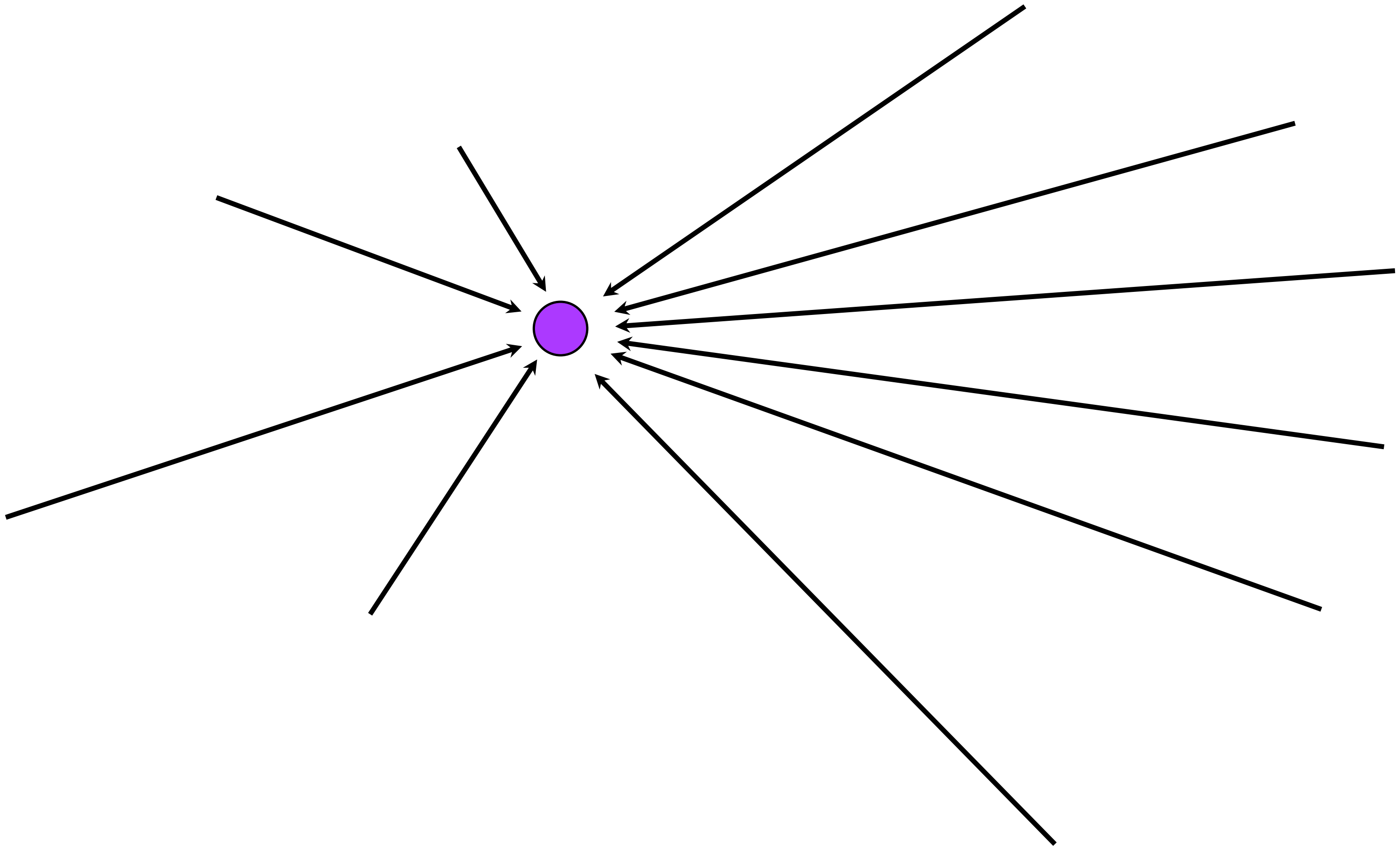
The (rearranged) pinhole camera



Pencil of rays

Consider all the rays passing through a point.

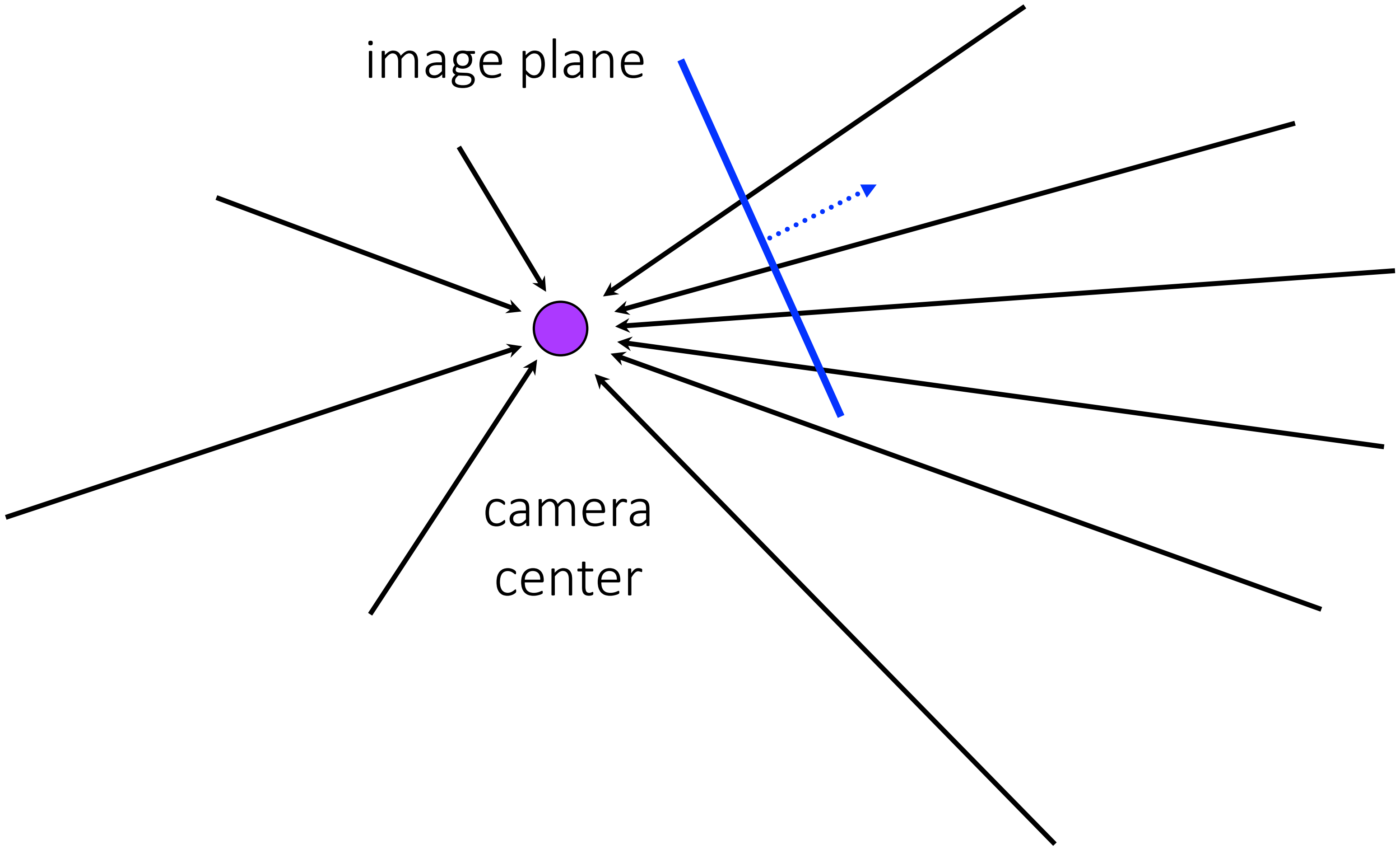
- What does it mean to take an image?



Pencil of rays

Consider all the rays passing through a point.

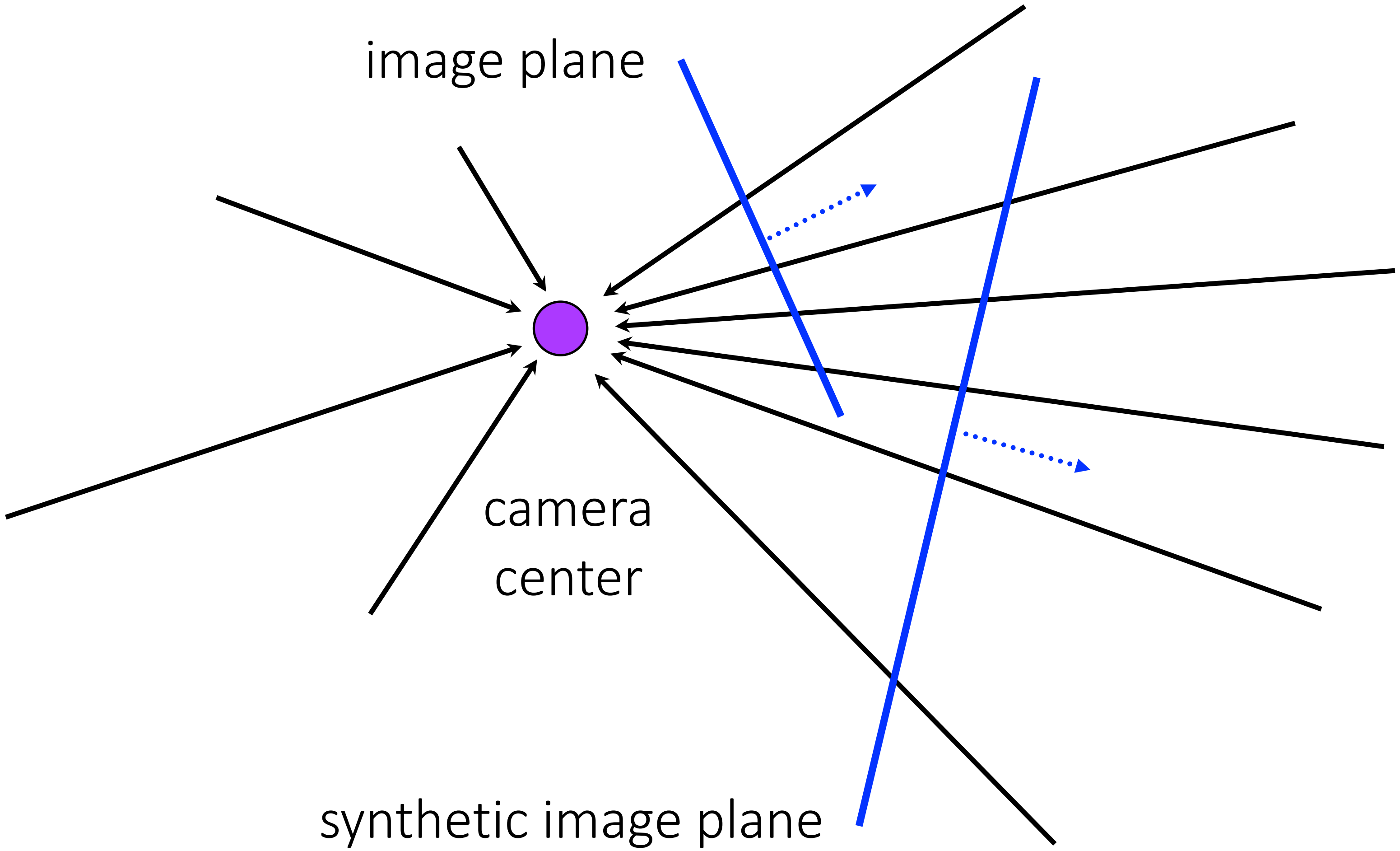
- Taking an image: slice through rays.
- What does it mean to change viewpoint?



Pencil of rays

Consider all the rays passing through a point.

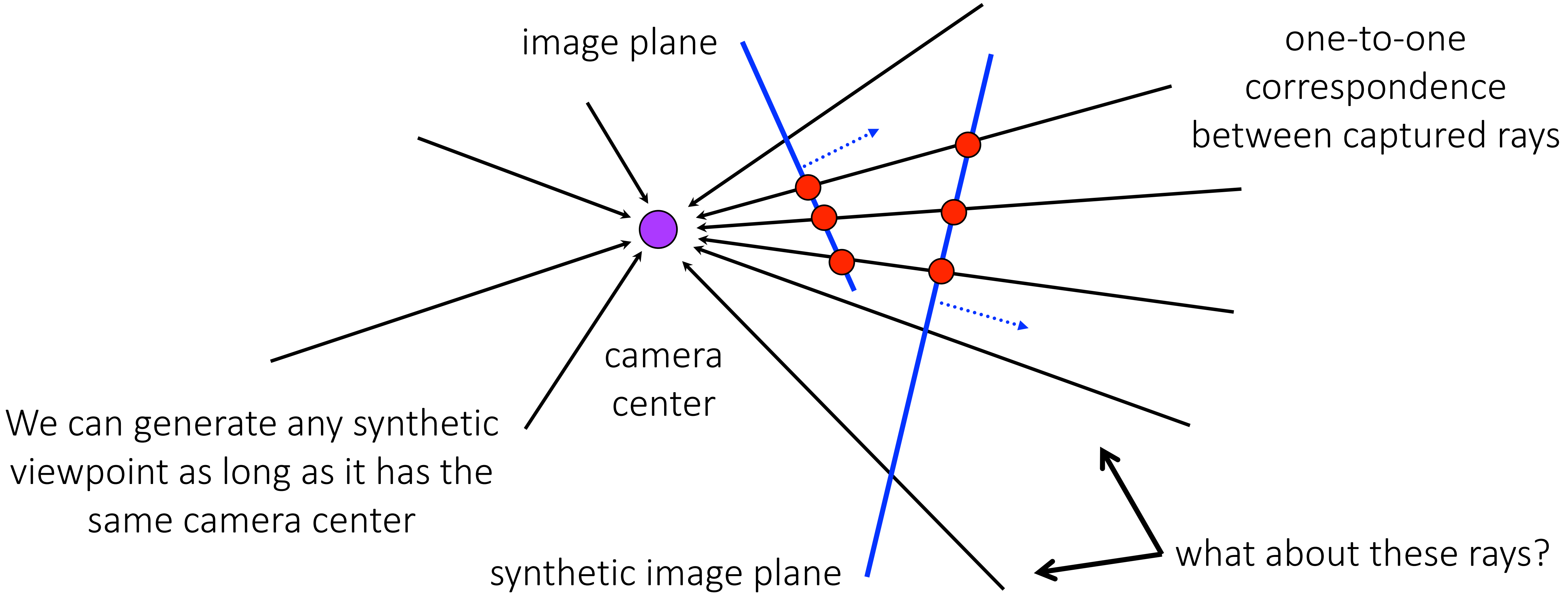
- Taking an image: slice through rays.
- Changing viewpoint: Rotate plane around center (optionally translate along viewpoint)



Pencil of rays

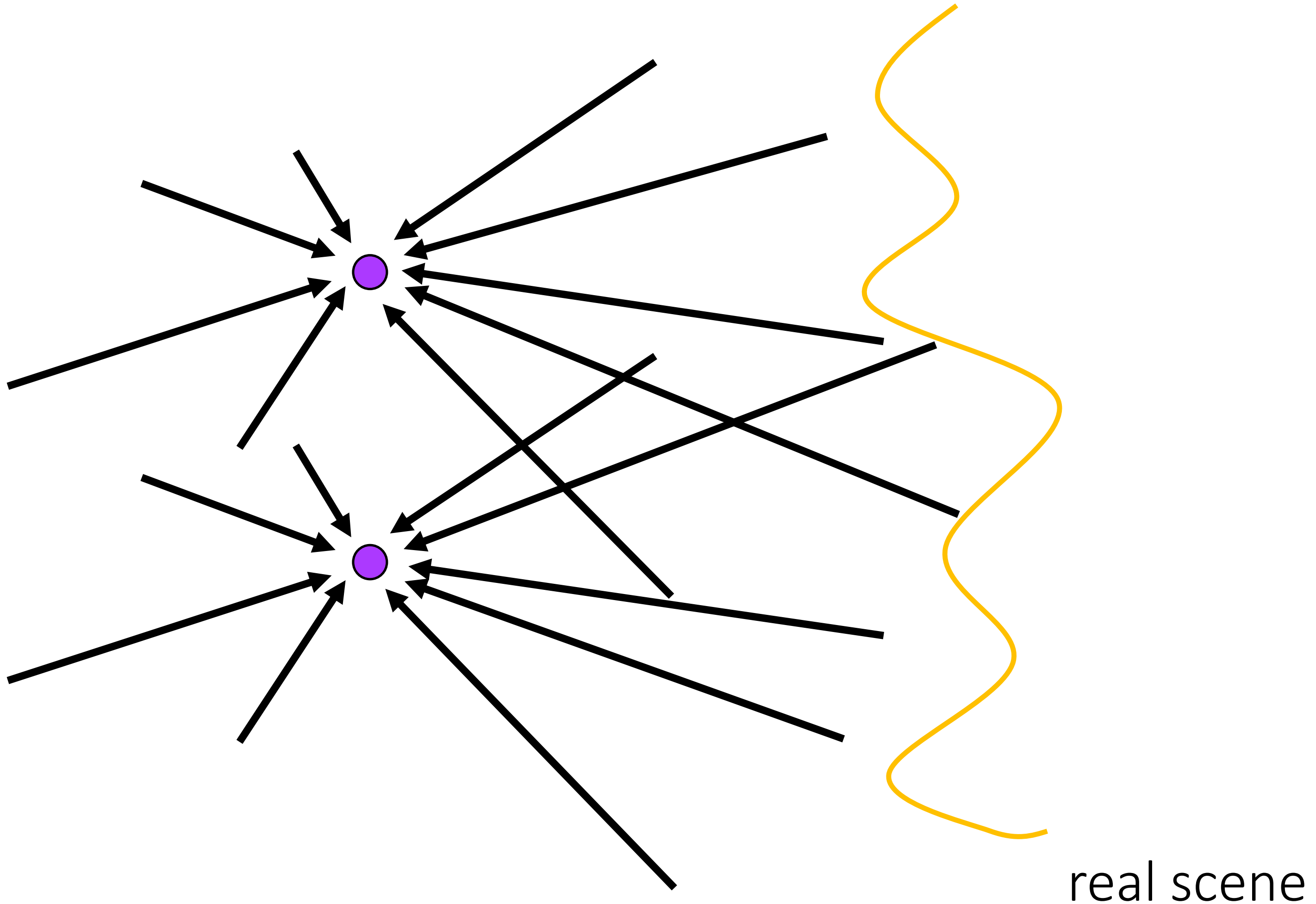
Consider all the rays passing through a point.

- Taking an image: slice through rays.
- Changing viewpoint: Rotate plane around center (optionally translate along viewpoint)



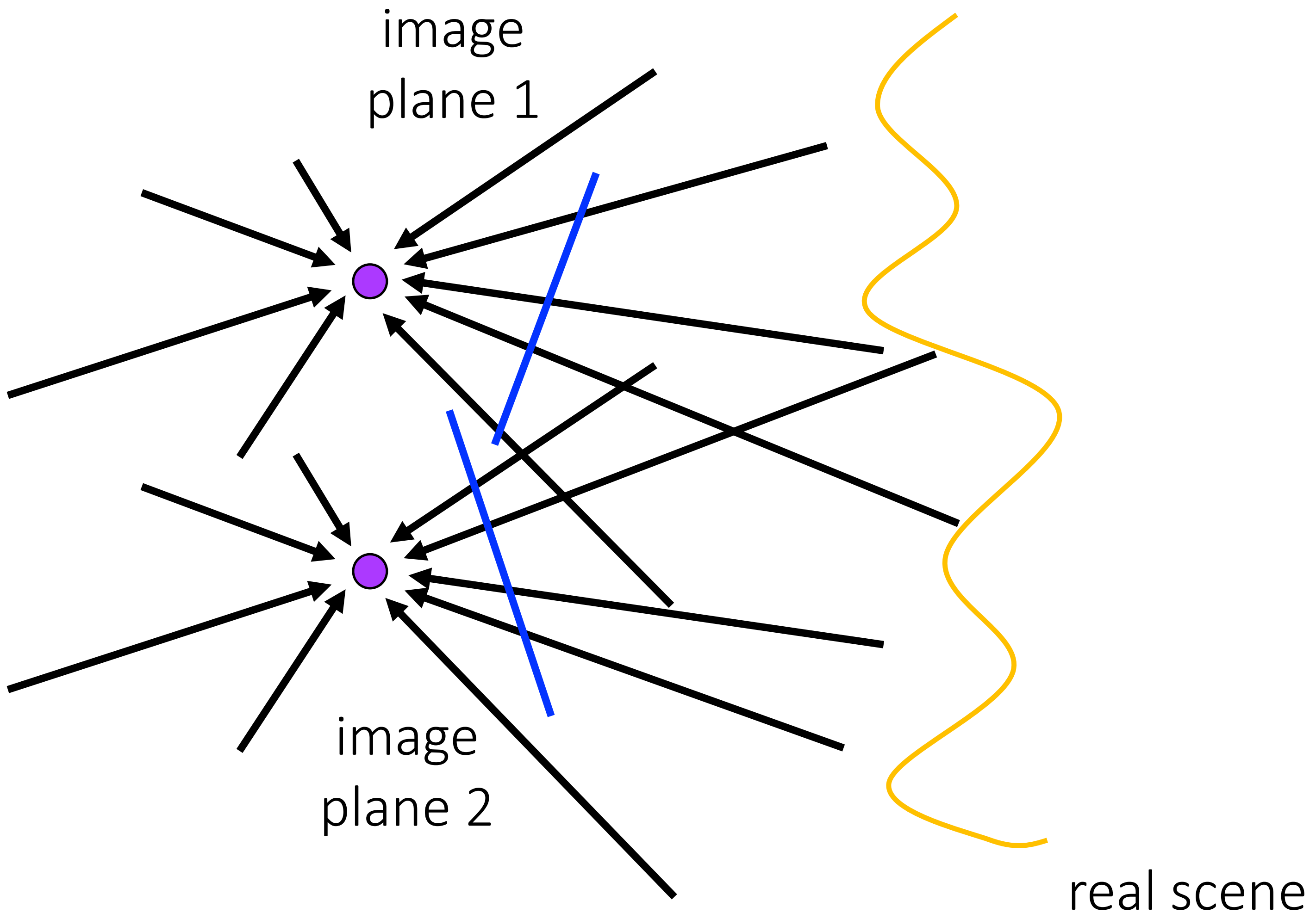
What happens if we change camera center?

Can we still use homographies to generate new views?



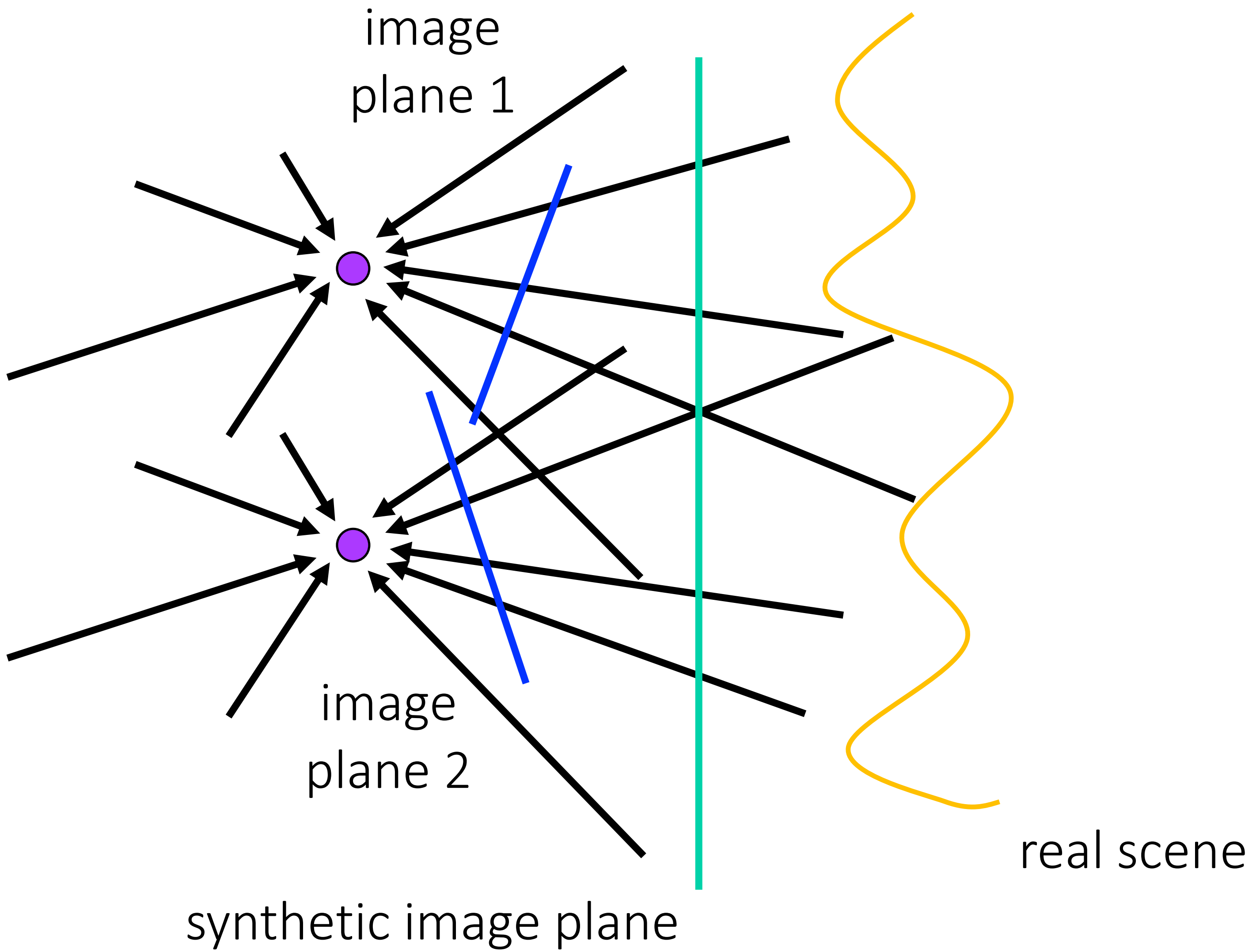
What happens if we change camera center?

Can we still use homographies to generate new views?



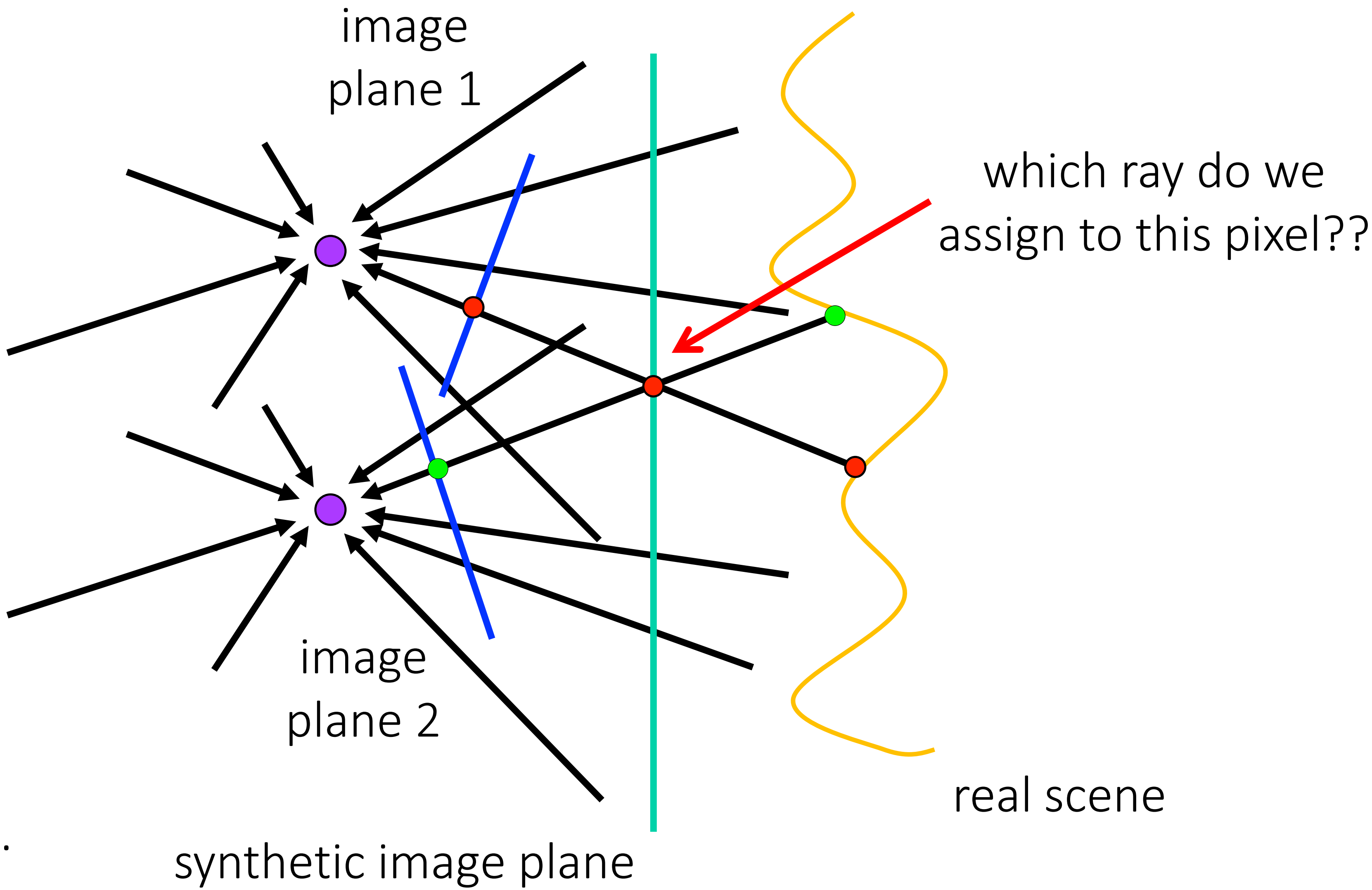
What happens if we change camera center?

Can we still use homographies to generate new views?



What happens if we change camera center?

Can we still use homographies to generate new views?



Projective transforms cannot *in general* deal with camera translation.

Homographies can handle camera translation when…

1. … the scene is planar; or

2. … the scene is very far or has small (relative) depth variation
   → scene is approximately planar

Projective transforms also work for …

translation with planar scenes or …

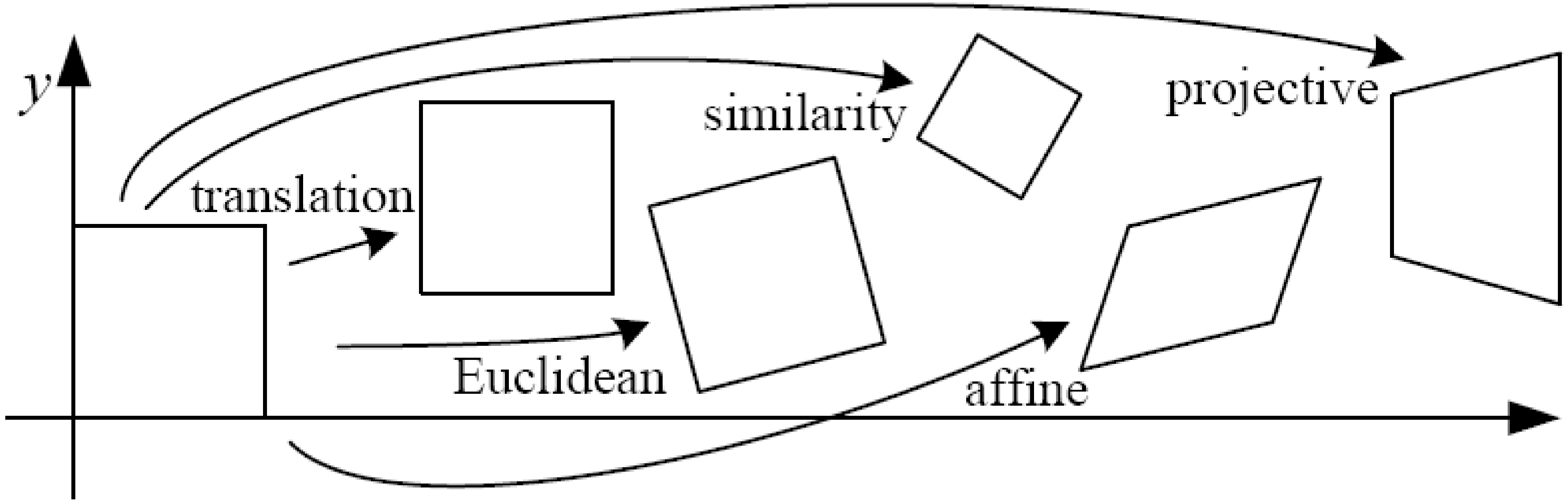
Projective transforms also work for …

planar scenes or …

scenes that are far away (with small depth variance).

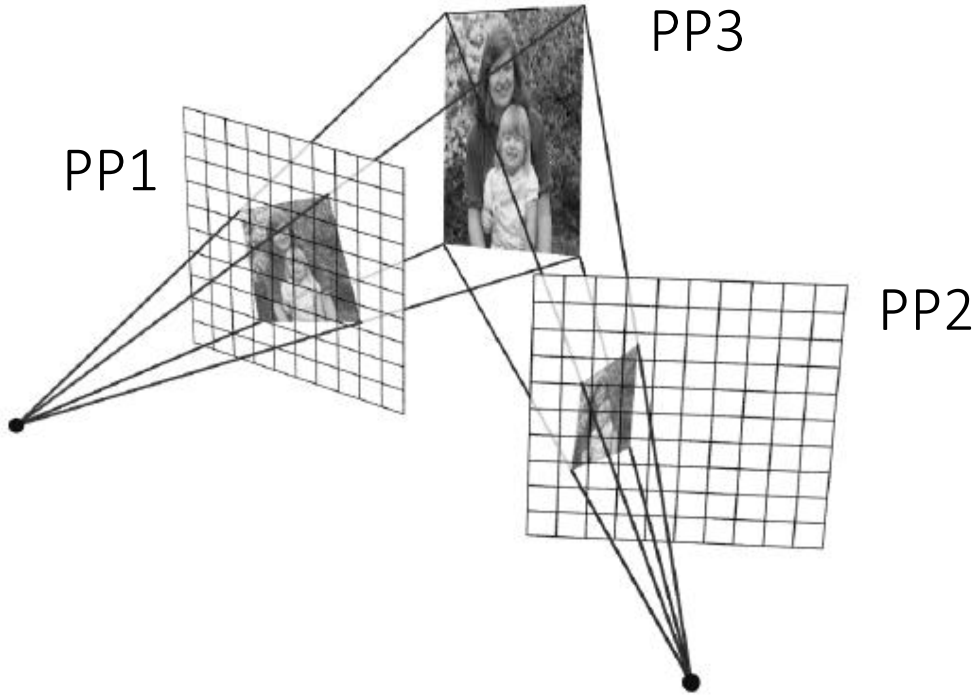
Computing with homographies

Classification of 2D transformations



Which kind transformation is needed to warp projective plane 1 into projective plane 2?

- A projective transformation (a.k.a. a homography).



Applying a homography

1. Convert to homogeneous coordinates: $p = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

What is the size of the homography matrix?

2. Multiply by the homography matrix: $P' = H \cdot P$

3. Convert back to heterogeneous coordinates: $P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \Rightarrow p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$

Applying a homography

1. Convert to homogeneous coordinates: $p = \begin{bmatrix} x \\ y \end{bmatrix} \;\Rightarrow\; P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$

What is the size of the homography matrix? ↘ Answer: 3 x 3

2. Multiply by the homography matrix: $P' = H \cdot P$

How many degrees of freedom does the homography matrix have? ↗

3. Convert back to heterogeneous coordinates: $P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \;\Rightarrow\; p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$

Applying a homography

1. Convert to homogeneous coordinates: $$p = \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

What is the size of the homography matrix? ↘ Answer: 3 x 3

2. Multiply by the homography matrix: $$P' = H \cdot P$$

How many degrees of freedom does the homography matrix have? ↗ Answer: 8

3. Convert back to heterogeneous coordinates: $$P' = \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} \Rightarrow p' = \begin{bmatrix} x'/w' \\ y'/w' \end{bmatrix}$$

Applying a homography

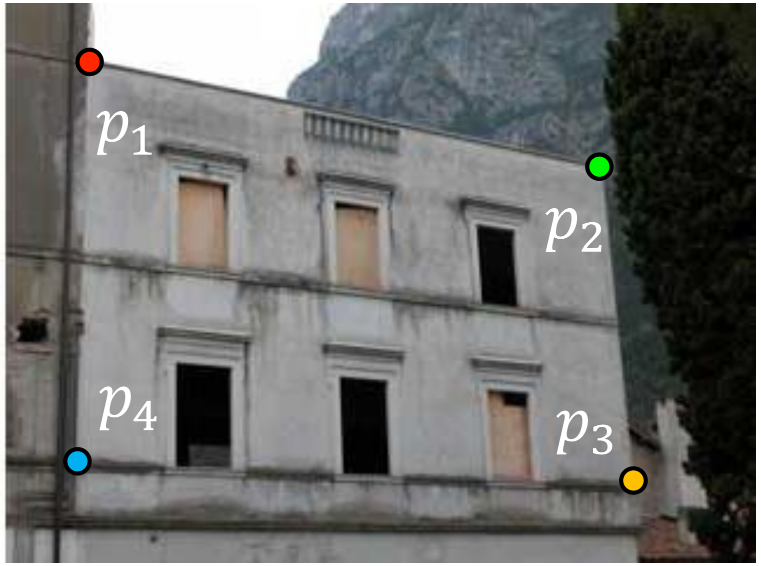
What is the size of the homography matrix? ↘ Answer: 3 x 3

$$P' = H \cdot P$$

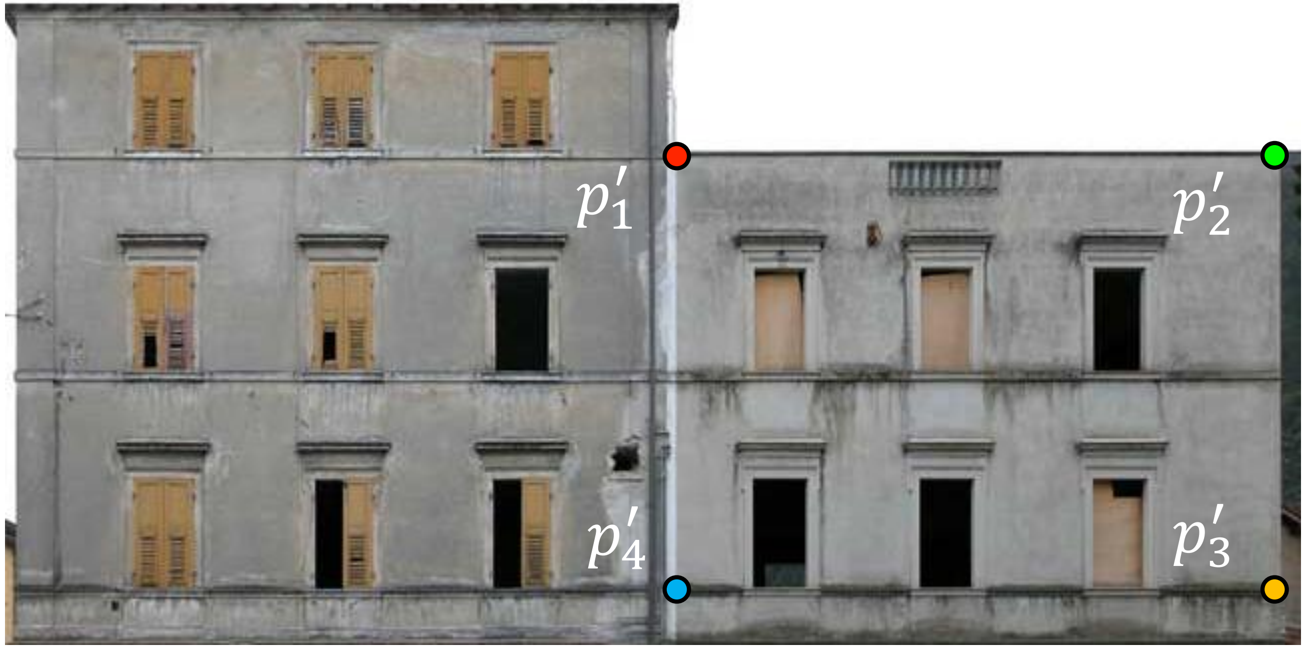
How many degrees of freedom does the homography matrix have? ↗ Answer: 8

How do we compute the homography matrix?

Create point correspondences



original image



target image

How many correspondences do we need?

Determining the homography matrix

Write out linear equation for each correspondence:

$$P' = H \cdot P \quad \text{or} \quad \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Expand matrix multiplication:

$$x' = \alpha(h_1 x + h_2 y + h_3)$$
$$y' = \alpha(h_4 x + h_5 y + h_6)$$
$$1 = \alpha(h_7 x + h_8 y + h_9)$$

Divide out unknown scale factor:

$$x'(h_7 x + h_8 y + h_9) = (h_1 x + h_2 y + h_3)$$
$$y'(h_7 x + h_8 y + h_9) = (h_4 x + h_5 y + h_6)$$

Determining the homography matrix

Re-arrange terms:

$$h_7xx' + h_8yx' + h_9x' - h_1x - h_2y - h_3 = 0$$
$$h_7xy' + h_8yy' + h_9y' - h_4x - h_5y - h_6 = 0$$

Re-write in matrix form:

$$\mathbf{A}_i \boldsymbol{h} = \mathbf{0}$$

$$\mathbf{A}_i = \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix}$$

$$\boldsymbol{h} = \begin{bmatrix} h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 & h_8 & h_9 \end{bmatrix}^\top$$

Determining the homography matrix

Stack together constraints from multiple point correspondences:

$$\mathbf{A}\boldsymbol{h} = \mathbf{0}$$

$$\begin{bmatrix} \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \\ \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \\ \vdots \\ \begin{bmatrix} -x & -y & -1 & 0 & 0 & 0 & xx' & yx' & x' \\ 0 & 0 & 0 & -x & -y & -1 & xy' & yy' & y' \end{bmatrix} \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Homogeneous linear least squares problem

- Solve with SVD (or `lmdivide` in MATLAB)

Reminder: Determining unknown transformations

Affine transformation:

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

Why can we drop the last line?

Vectorize transformation parameters:

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

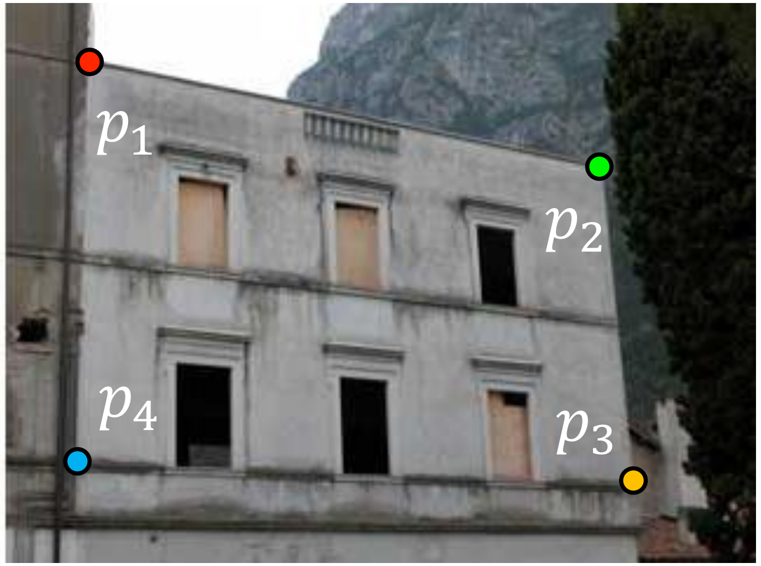
Stack equations from point correspondences:

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

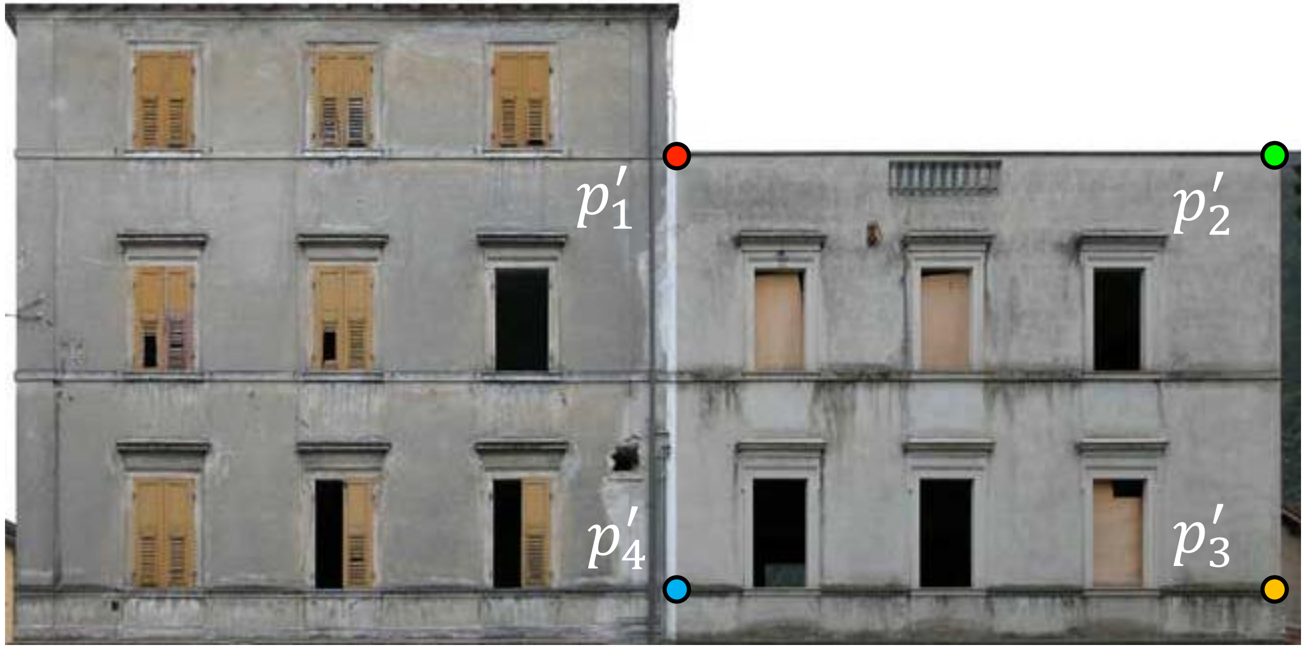
Notation in system form:

$$Ax = b$$

Create point correspondences



original image



target image

Can we automate this step?

The image correspondence pipeline

The image correspondence pipeline

1. Feature point detection

2. Feature point description

3. Feature matching and homography estimation

Feature points

What points should we try to match across the two images?

Feature points

What points should we try to match across the two images?

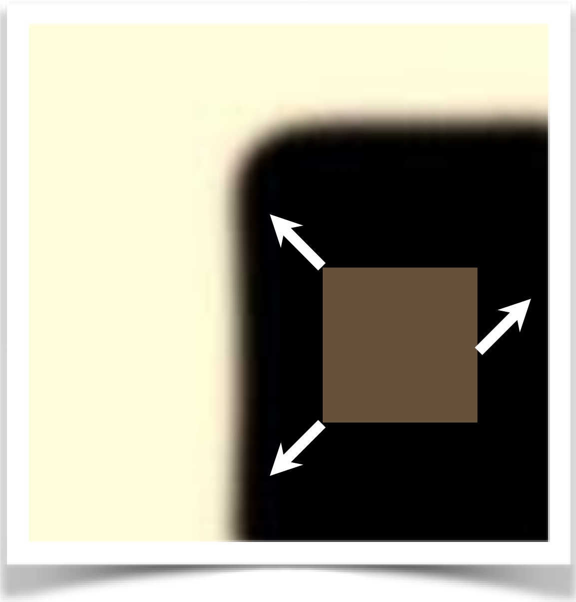
- Points that are prominent in both images.
- Points that are easy to detect.
- Points that are hard to confuse.

Example: corners.

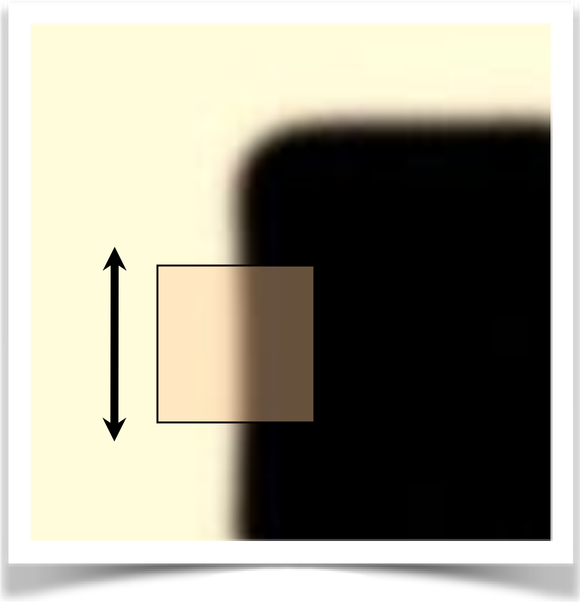
How do we detect a corner?

Easily recognized by looking through a small window

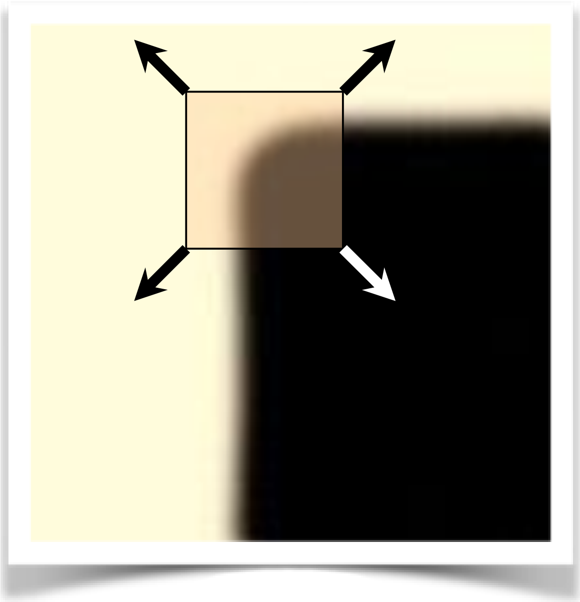
- Shifting the window should give large change in intensity



"flat" region:
no change in all directions



"edge":
no change along the edge direction

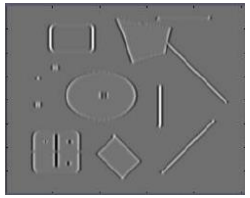


"corner":
significant change in all directions

Harris corner detector

1. Compute image gradients over small region

$I_x = \frac{\partial I}{\partial x}$ $\quad$ $I_y = \frac{\partial I}{\partial y}$



2. Subtract mean from each image gradient

3. Compute the covariance matrix

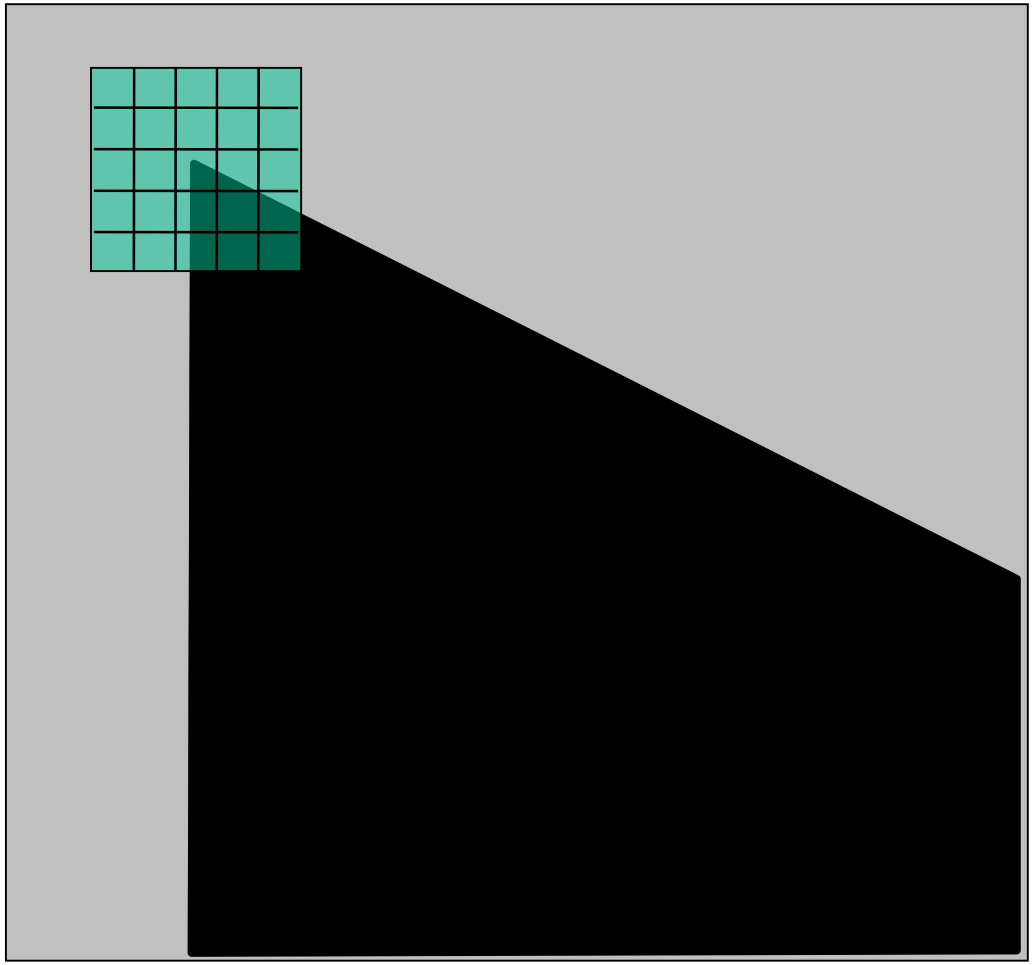
$$\begin{bmatrix} \sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\ \sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y \end{bmatrix}$$

4. Compute eigenvectors and eigenvalues

5. Use threshold on eigenvalues to detect corners

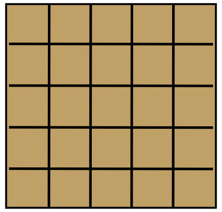
Compute image gradients

Must be computed over a small region



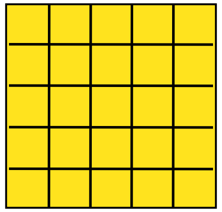
array of x gradients

$$I_x = \frac{\partial I}{\partial x}$$

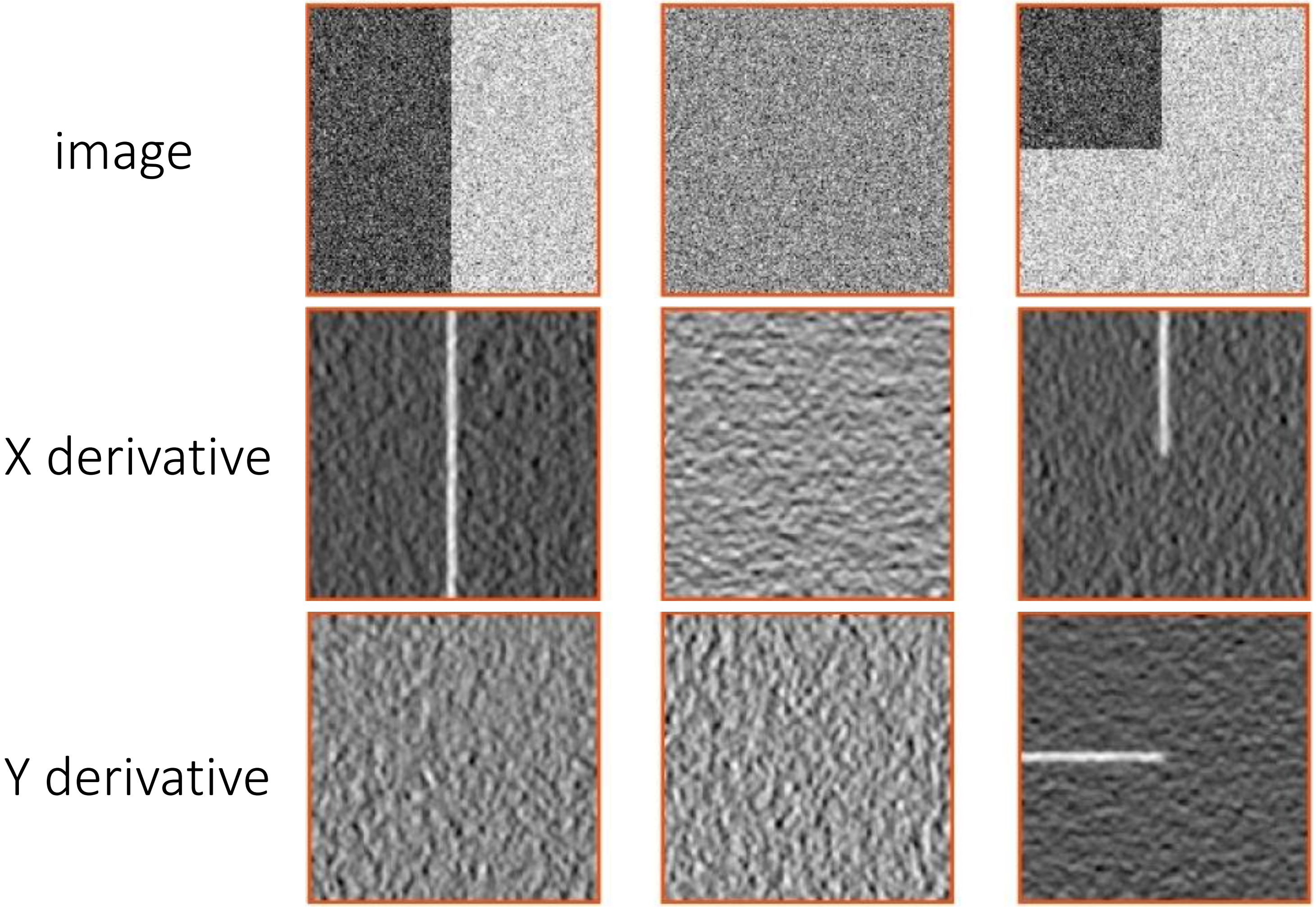


array of y gradients

$$I_y = \frac{\partial I}{\partial y}$$

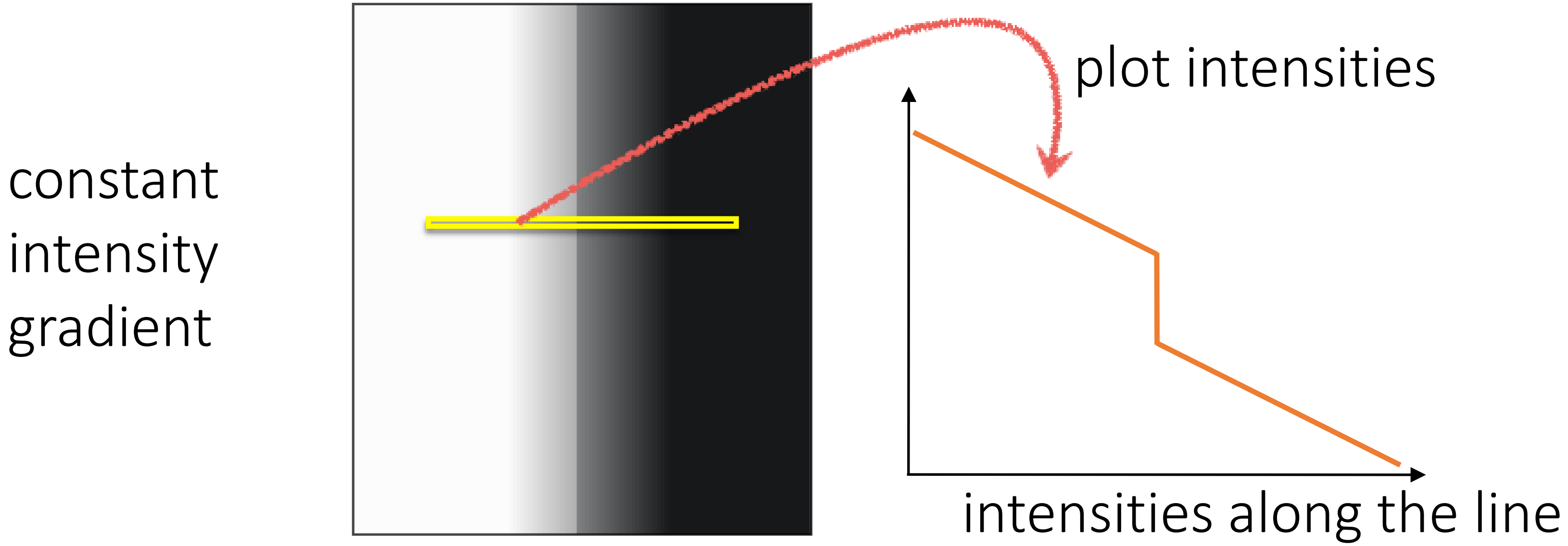


Compute image gradients



Subtract mean

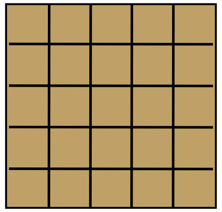
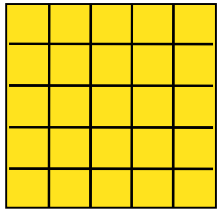
Data is centered ('DC' offset is removed)



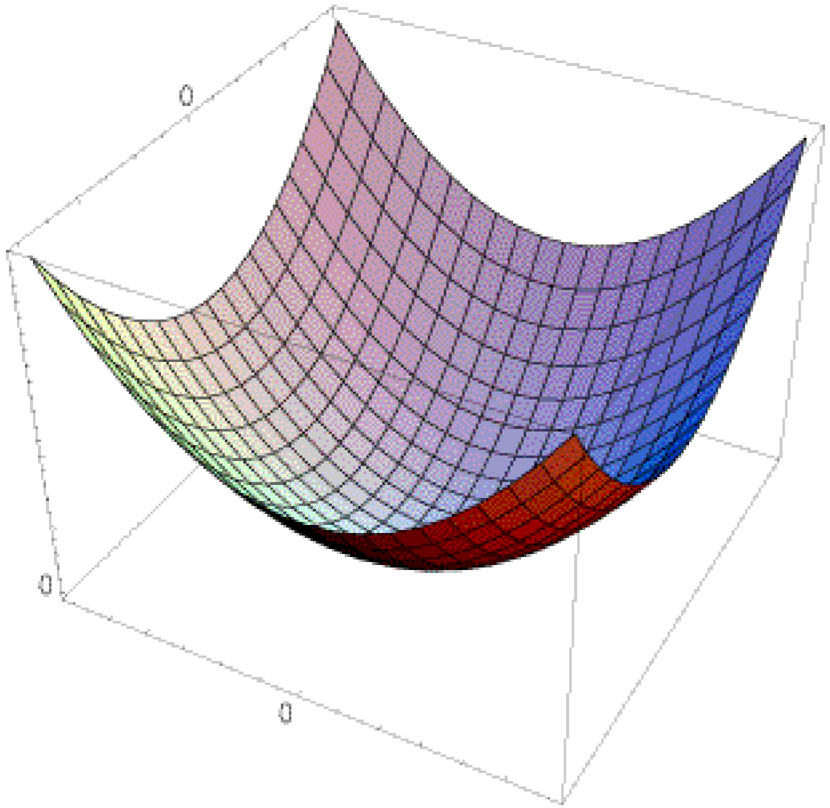
Compute the covariance matrix

$$\begin{bmatrix} \sum_{p \in P} I_x I_x & \sum_{p \in P} I_x I_y \\ \sum_{p \in P} I_y I_x & \sum_{p \in P} I_y I_y \end{bmatrix}$$

$I_x = \frac{\partial I}{\partial x}$ $\qquad$ $I_y = \frac{\partial I}{\partial y}$

$\sum_{p \in P} I_x I_y$ =sum( .* )

array of x gradients $\qquad$ array of y gradients



By computing the covariance matrix, we fit a *quadratic* to the image patch

Computing eigenvalues

Since M is symmetric, we have

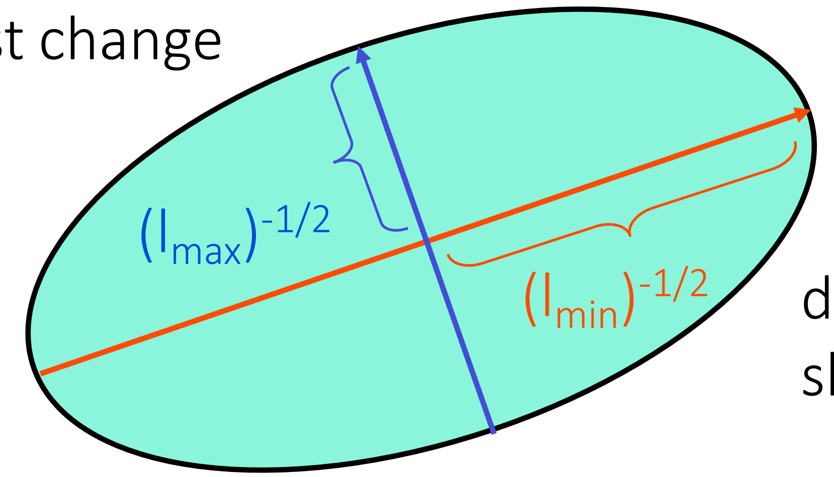
$$M = R^{-1}\begin{bmatrix}\lambda_1 & 0 \\ 0 & \lambda_2\end{bmatrix}R$$

We can visualize *M* as an ellipse with axis lengths determined by the eigenvalues and orientation determined by *R*

Ellipse equation:

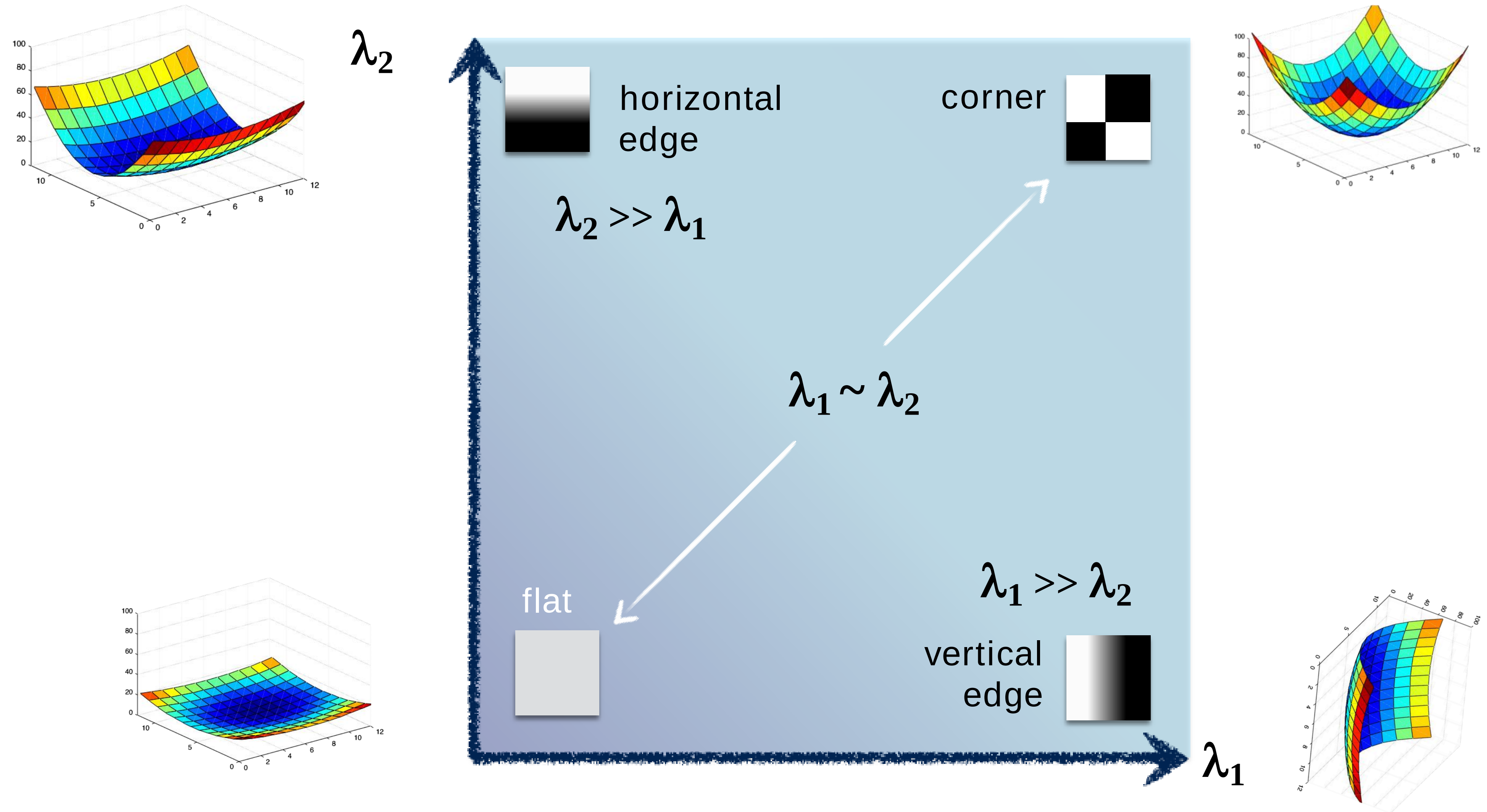
$$[u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$

direction of the fastest change



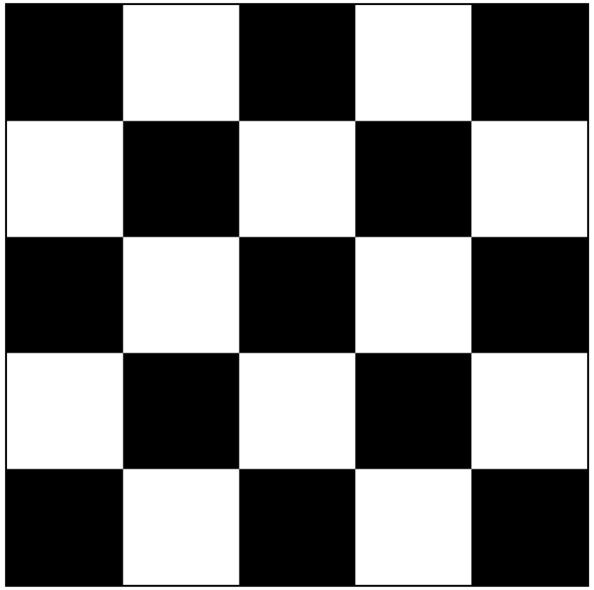
direction of the slowest change

Interpreting eigenvalues

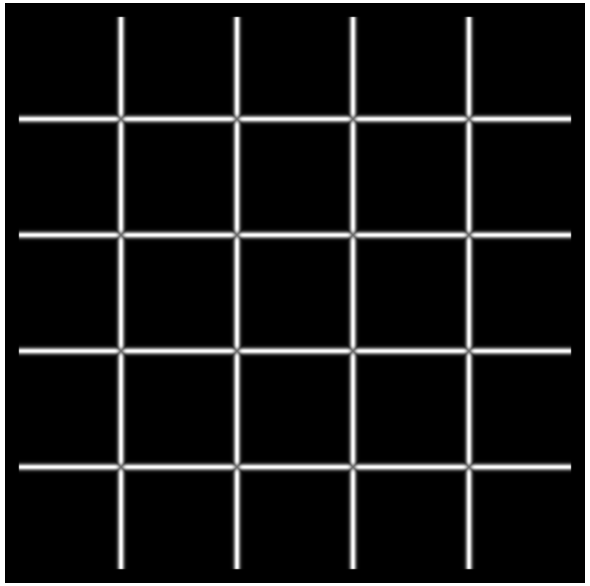


Corner detection summary

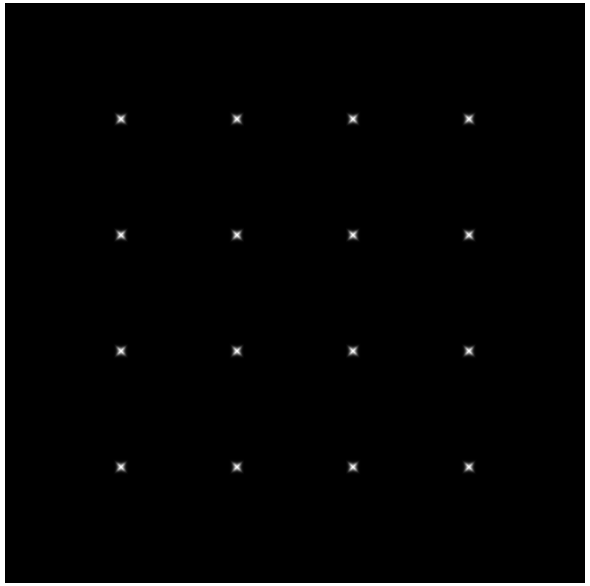
- Compute the gradient at each point in the image
- Create the *H* matrix from the entries in the gradient
- Compute the eigenvalues.
- Find points with large response (λ_{min} > threshold)
- Choose those points where λ_{min} is a local maximum as features



I



$\lambda_{\max}$



$\lambda_{\min}$

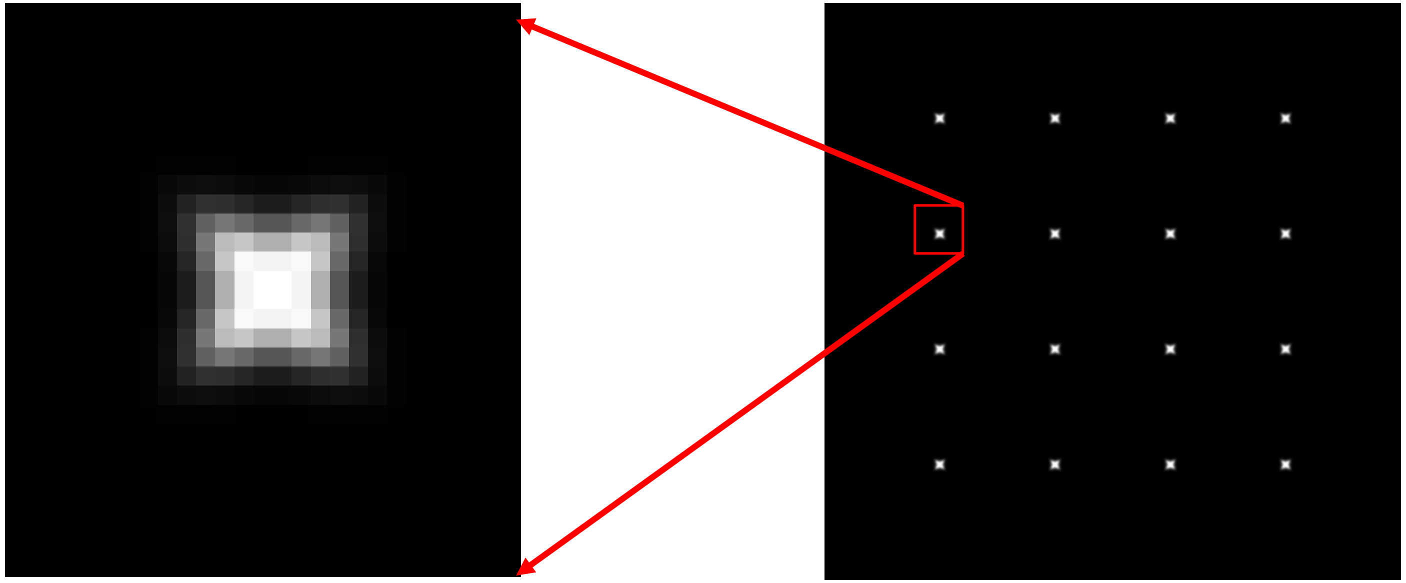
The Harris corner criterion

λ_{min} is a variant of the "Harris operator" for feature detection

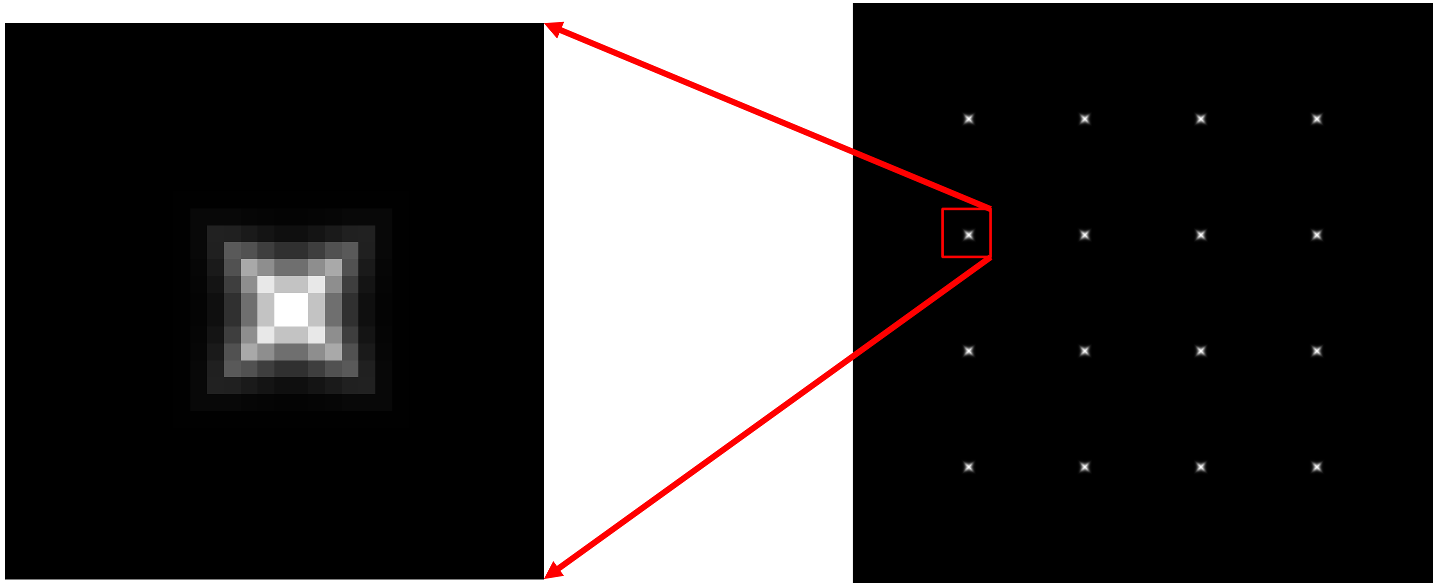
$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$
$$= \frac{determinant(H)}{trace(H)}$$

- The *trace* is the sum of the diagonals, i.e., $trace(H) = h_{11} + h_{22}$
- Very similar to λ_{min} but less expensive (no square root)
- Called the "Harris Corner Detector" or "Harris Operator"
- Lots of other detectors, this is one of the most popular

The Harris corner criterion

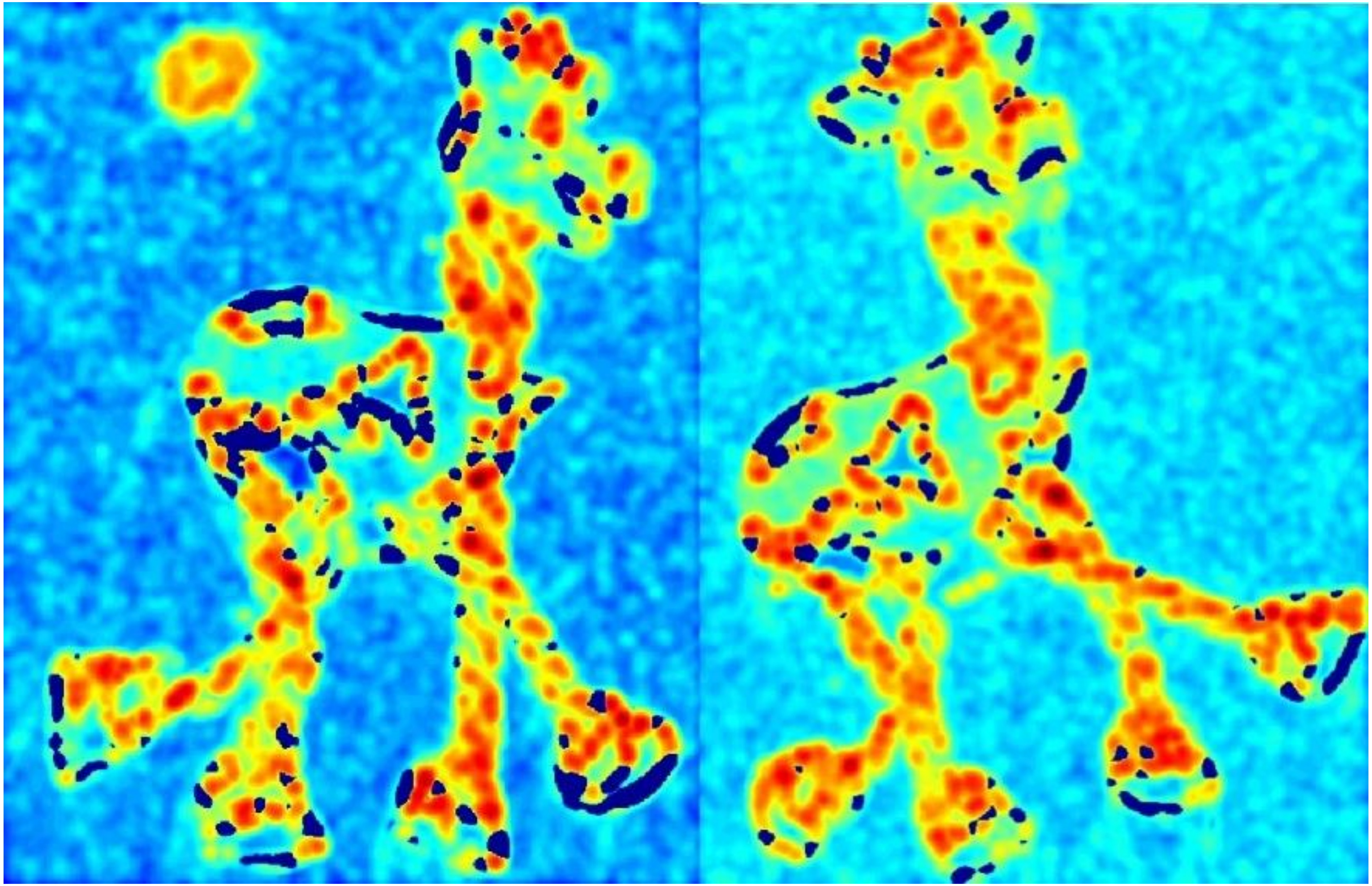


Harris criterion

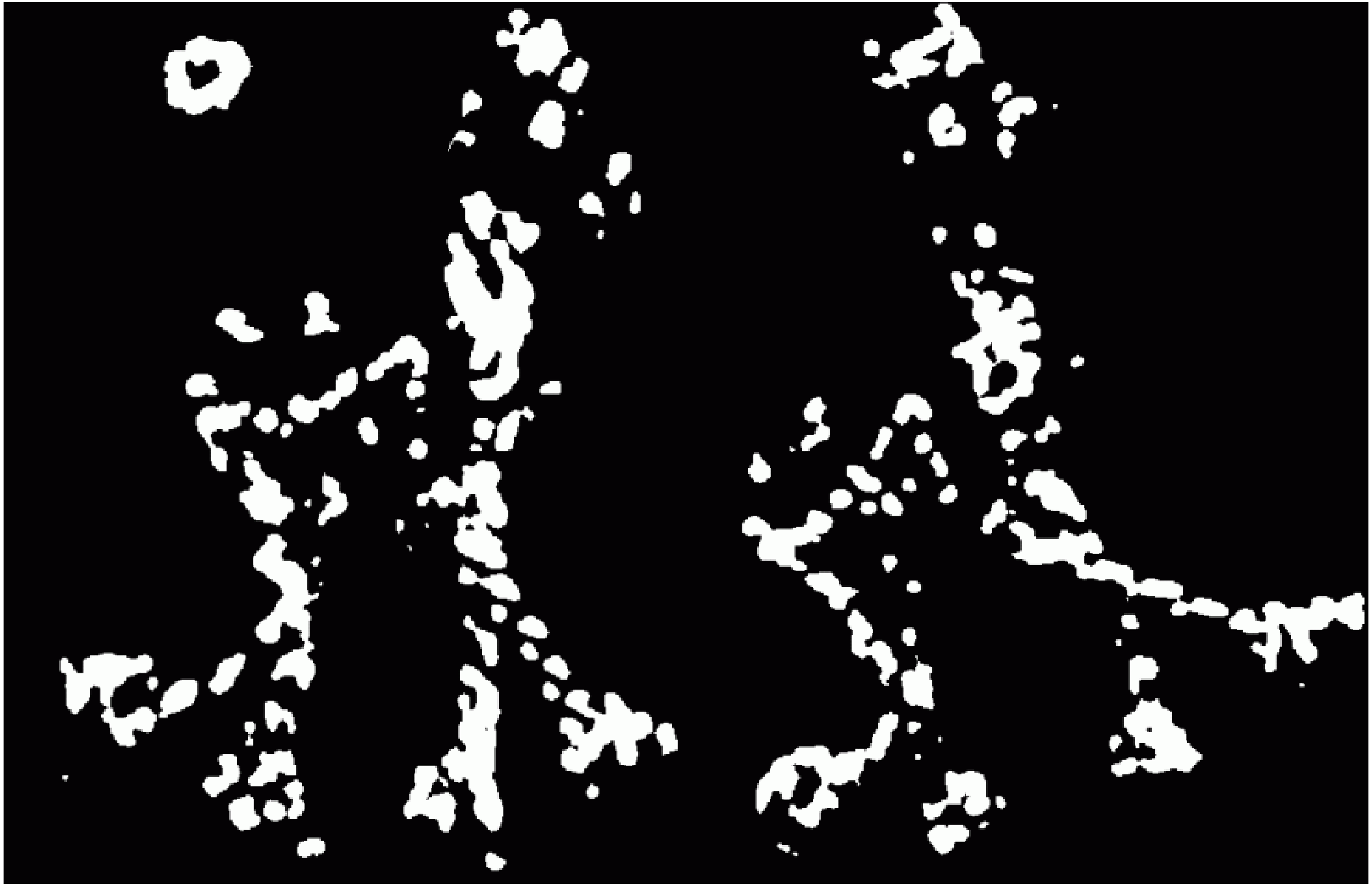


$\lambda_{\min}$

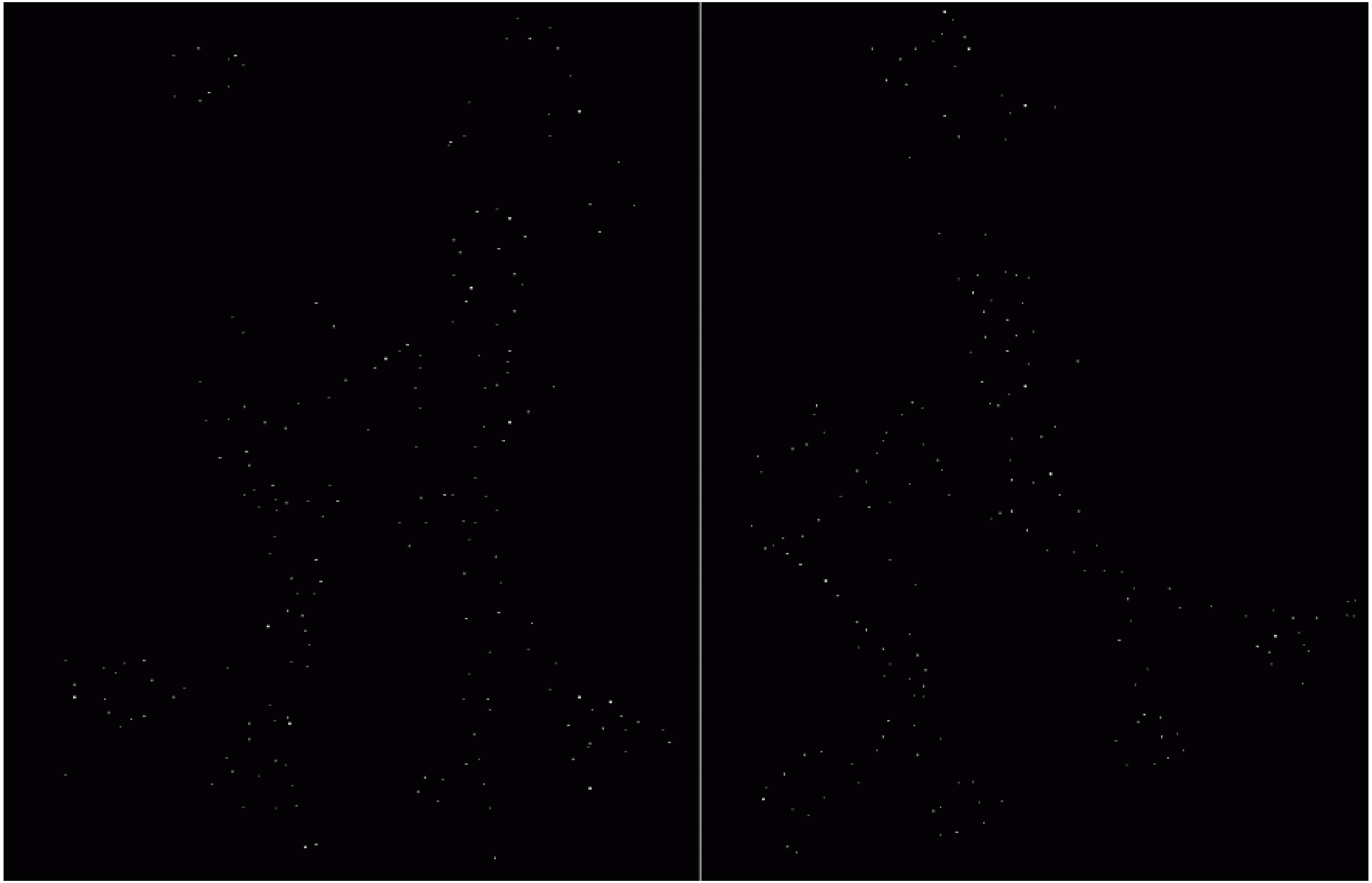

Corner response




Thresholded corner response



Non-maximal suppression



References

Basic reading:

- Szeliski textbook, Sections 2.1.2, 9.1.

Additional reading:

- Hartley and Zisserman, “Multiple View Geometry,” Cambridge University Press 2003.
  as usual when it comes to geometry and vision, this book is the best reference; Sections 2 and 4 in particular discuss everything about homography estimation