Midterm Review

CS 180

Scope

- Everything discussed in lecture up to and including stereo (❌ epipolar)
- Key concepts and coding knowledge from Project 0, Project 1, Project 2, Project 3A, and Project 3B

How do I study for this?

- **Discussion** sheets have similar style questions
- Quizzes (answers released)
- Understand the **projects** well

You have a cheat sheet (handwritten + ipad print*, 1 page, letter-size, 2 sides)!
so don’t memorize formulas

*no screenshots / slides

Agenda

Structured with homeworks, each with concepts!

- **Project 0:** Pinhole camera model
- **Project 1:** Gaussian / laplacian pyramids
- **Project 2:** Filters + frequencies
- **Project 3A + stereo:** Homography + RANSAC

Intermediate checkpoint questions to discuss with your neighbors throughout

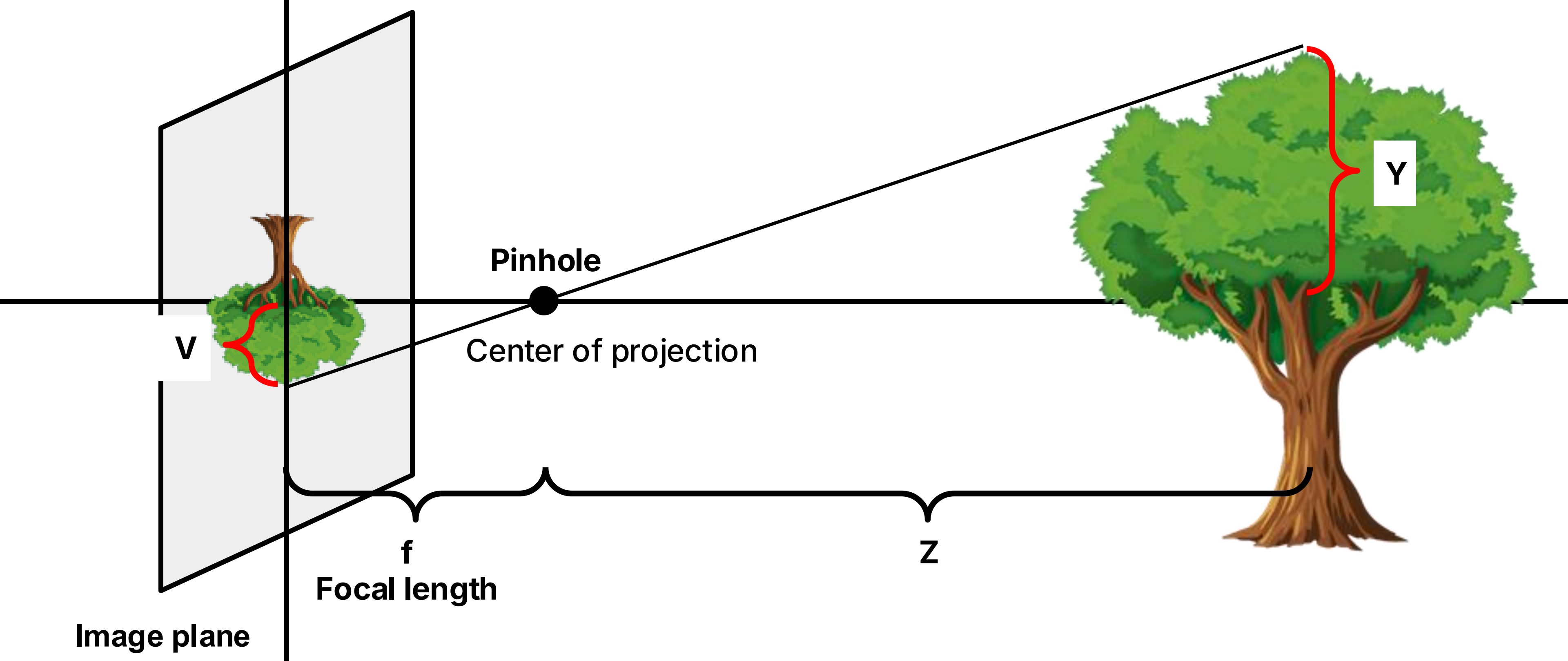
Which one you want to go over first? 🖐

Project 0

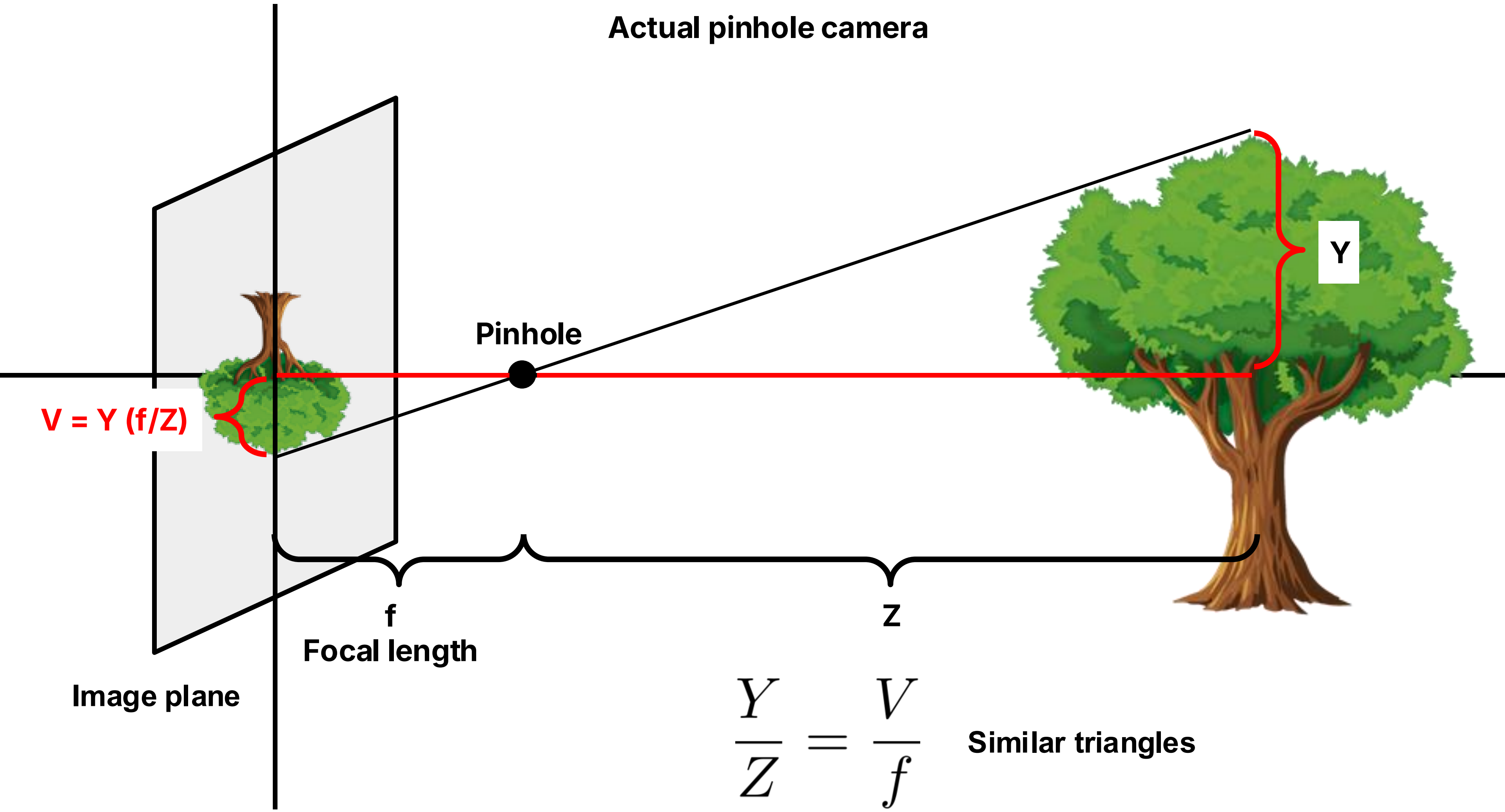
Key Topics:

- Pinhole Camera Model

Actual pinhole camera

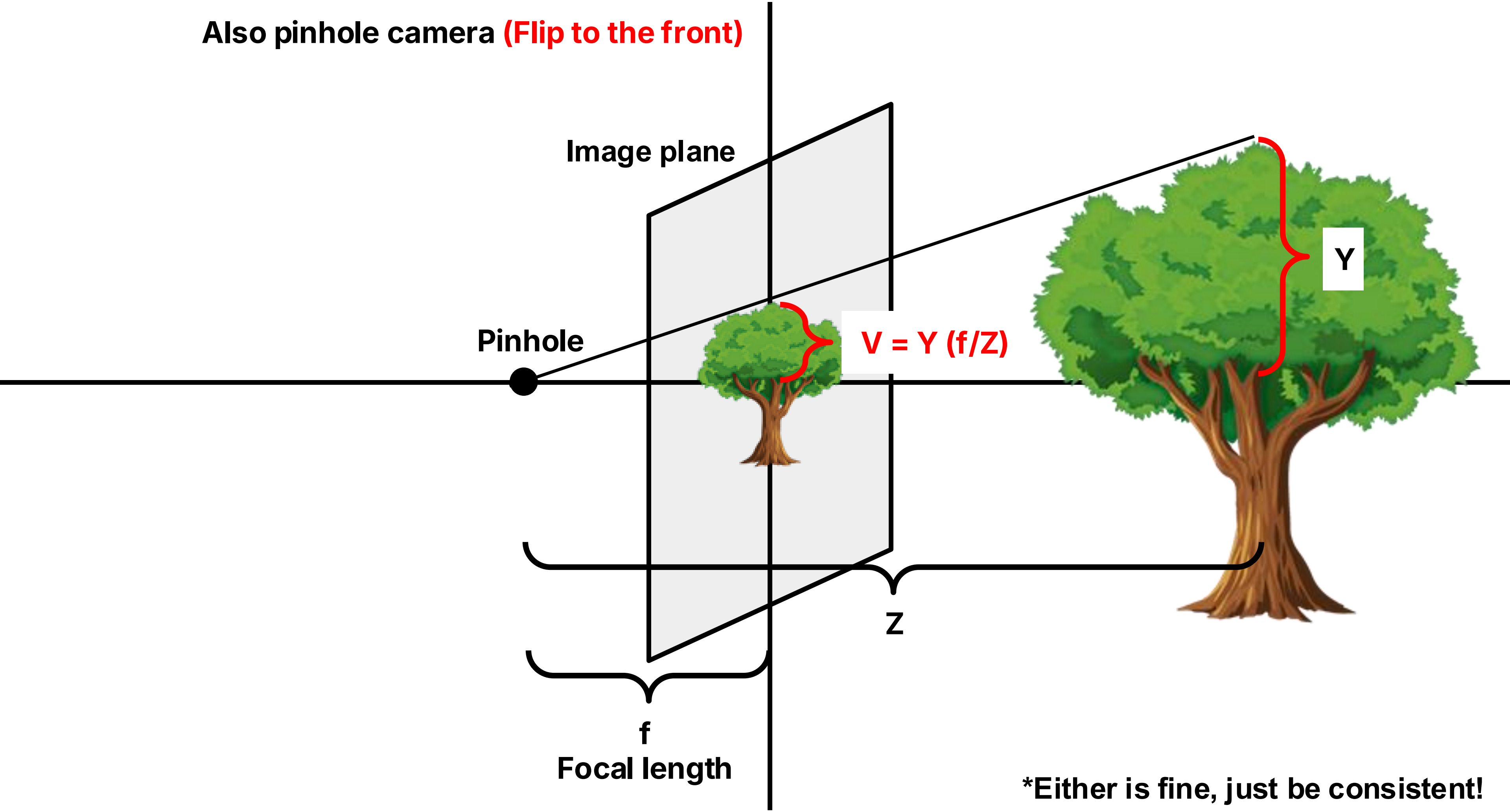


*How to relate Y (real world) to V (image)?



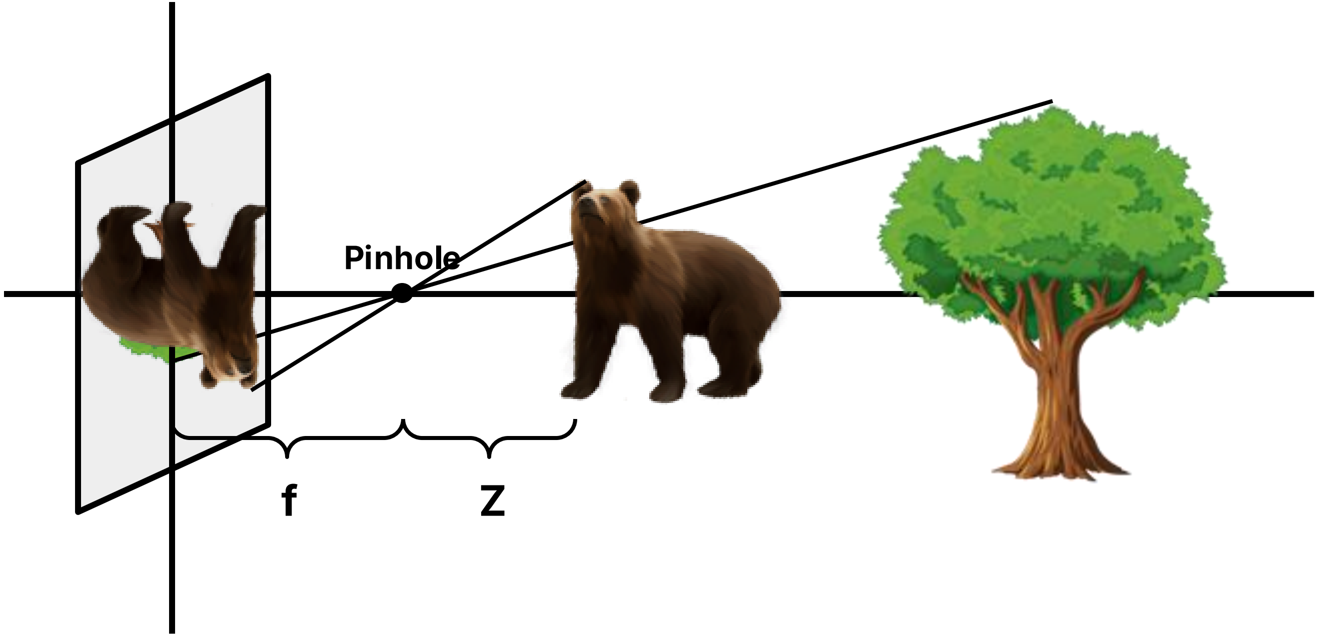
Actual pinhole camera
Pinhole
Y
V = Y (f/Z)
f
Focal length
Z
Image plane
$$\frac{Y}{Z} = \frac{V}{f}$$
Similar triangles

Also pinhole camera (Flip to the front)



*Either is fine, just be consistent!

Dolly Zoom (+ questions)



- What if we **zoom in** (larger f)?
- What if we **walk in** (smaller Z)?
- What if we **zoom out and walks in at the same time**?

Project 1

Aliasing and Antialiasing

Gaussian Pyramids

Multi-Scale Image Alignment

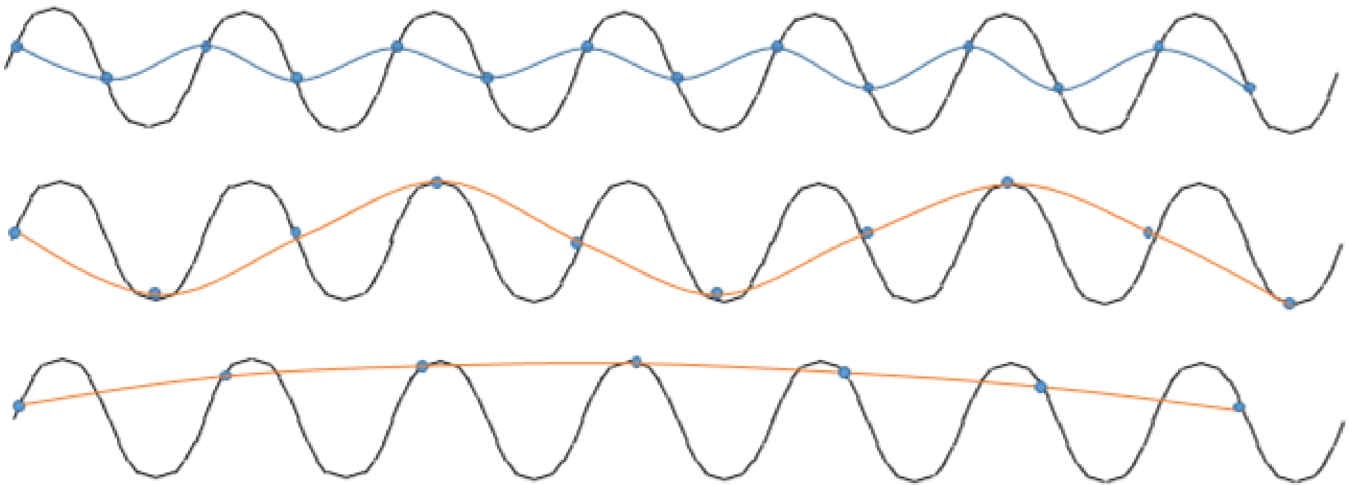
Sanity check!

T/F: An image in a computer always has shape (H, W, 3)

False!

- Can be channel-first (CHW) or channel-last (HWC)
- Can be single-channel (1) e.g., for depth

Aliasing



Aliasing Examples

Concept check!

Name a scenario where you could see aliasing

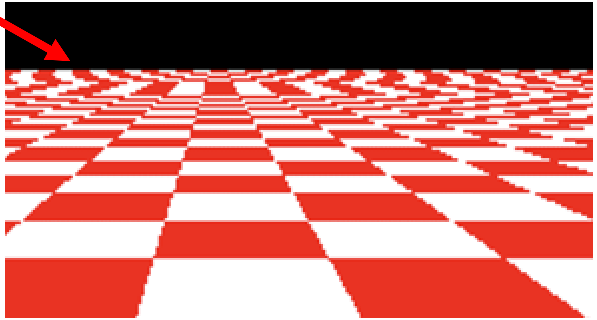
1) with your own eyes and

2) with a camera

Concept check!

1) Looking through a screen door

2) Taking a picture of a repeating pattern at low resolution



Antialiasing

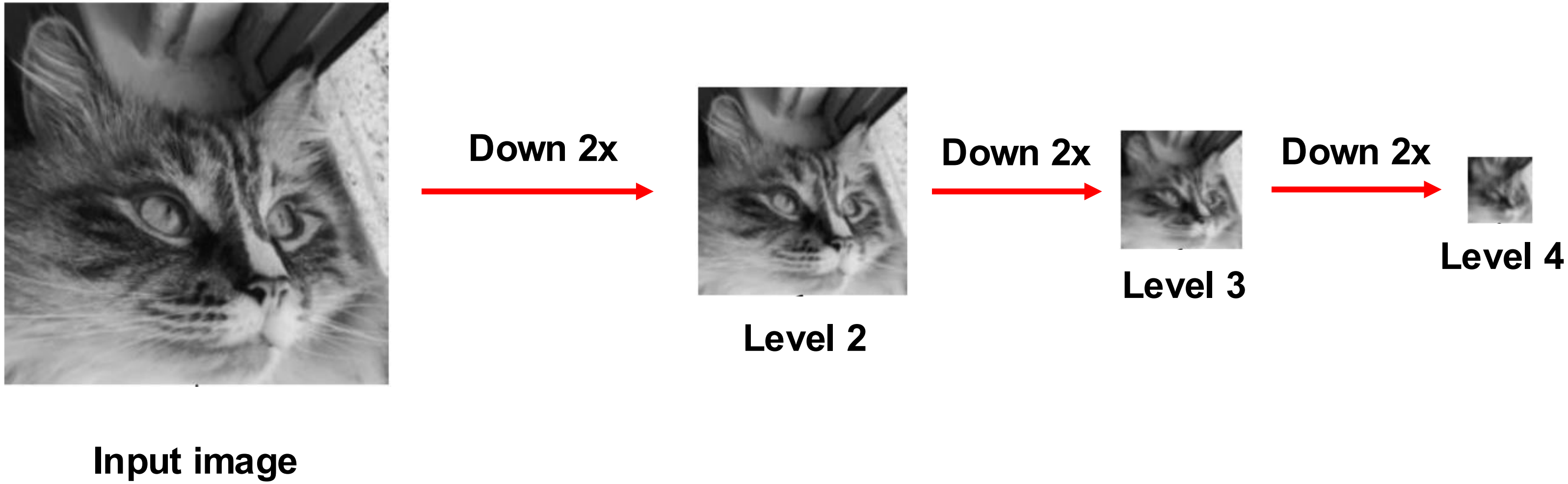
How can we prevent aliasing?

- Increase sampling rate (resolution)
- Filter out the high frequencies that can be aliased (blur / low pass)

Gaussian Pyramid

Computationally efficient for image processing

Useful for feature matching etc..



Chat with your neighbor

What does a Gaussian pyramid represent in the frequency space?

Multiple levels of low-pass filtering on an image.

Blurring an image after sampling is effective for antialiasing (T/F ?)

False, aliasing occurs during sampling (need to anti before).

What is the difference between a gaussian *pyramid* and *stack?*

A stack = same size, a pyramid = smaller

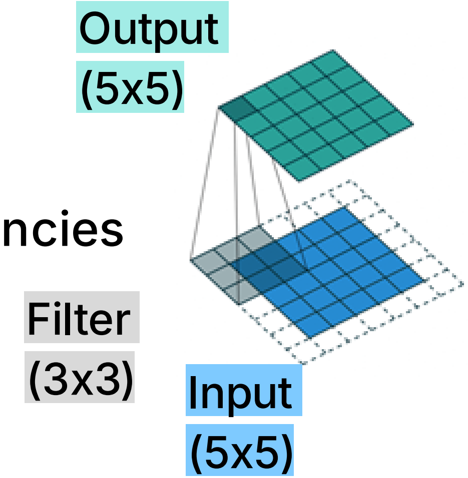
Project 2

Key Topics:

- Convolution
- Filters
    - Low Pass, High Pass, Hybrid Images
- Fourier Transform
- Gaussian and Laplacian Pyramids
    - Image Blending

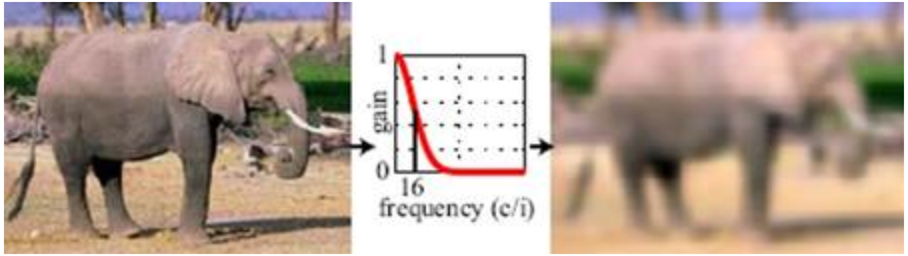
Convolution

- Multiplying with a **flipped (horz + vert)** kernel + shift across
  - **Cross-correlation:** no flip.
- Can **<u>enhance or suppress features</u>**
  - Lighten / Darken
  - Edge detection
  - Reduce (some kind) noise
- **Convolution** is commutative (can swap order)
- **Convolution theorem:** convolving = multiplying frequencies



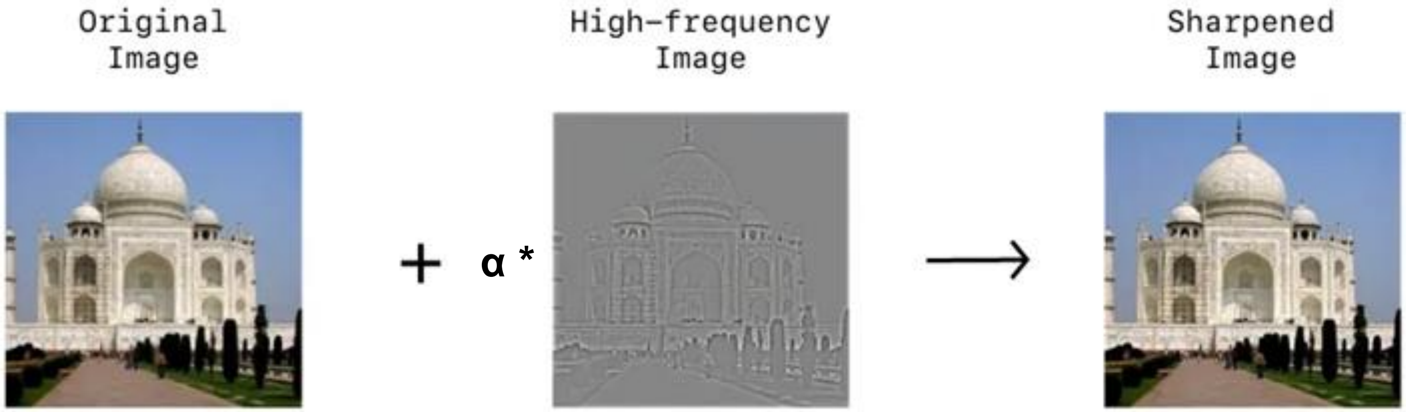
Low-Pass Filters

- Filter out high frequencies (Blur)
- Reduce aliasing
- **Examples:**
    - **Mean filter (box)**
    - **Gaussian filter**
    - (Cannot) Median filter



High-Pass Filters

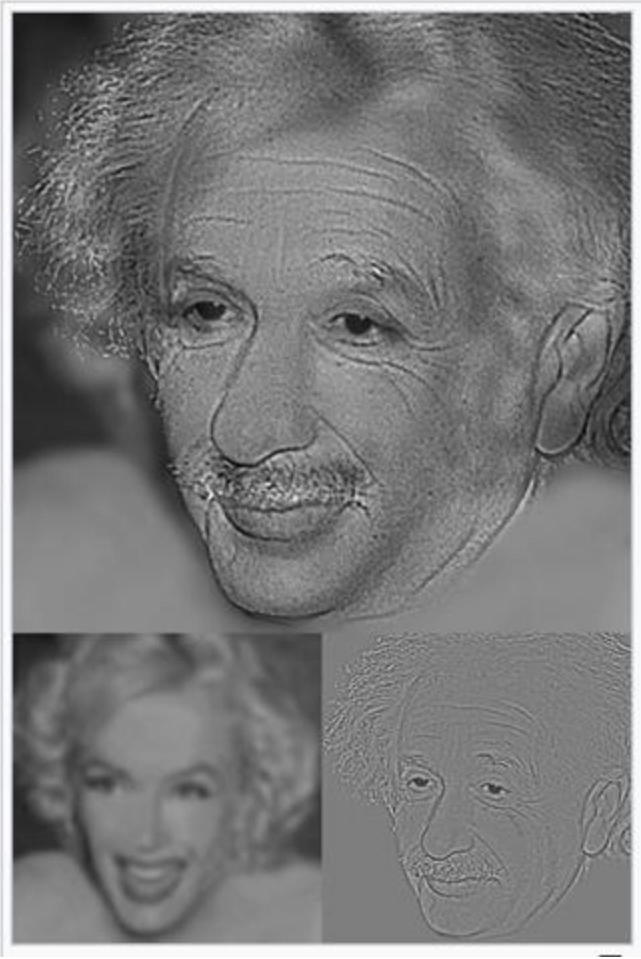
- Filter out low-frequency, like flat regions
- **Examples:**
    - **Unsharp masking =** I + alpha * (high pass)



Hybrid Images

- **Low pass** of one image
- **High pass** of another image

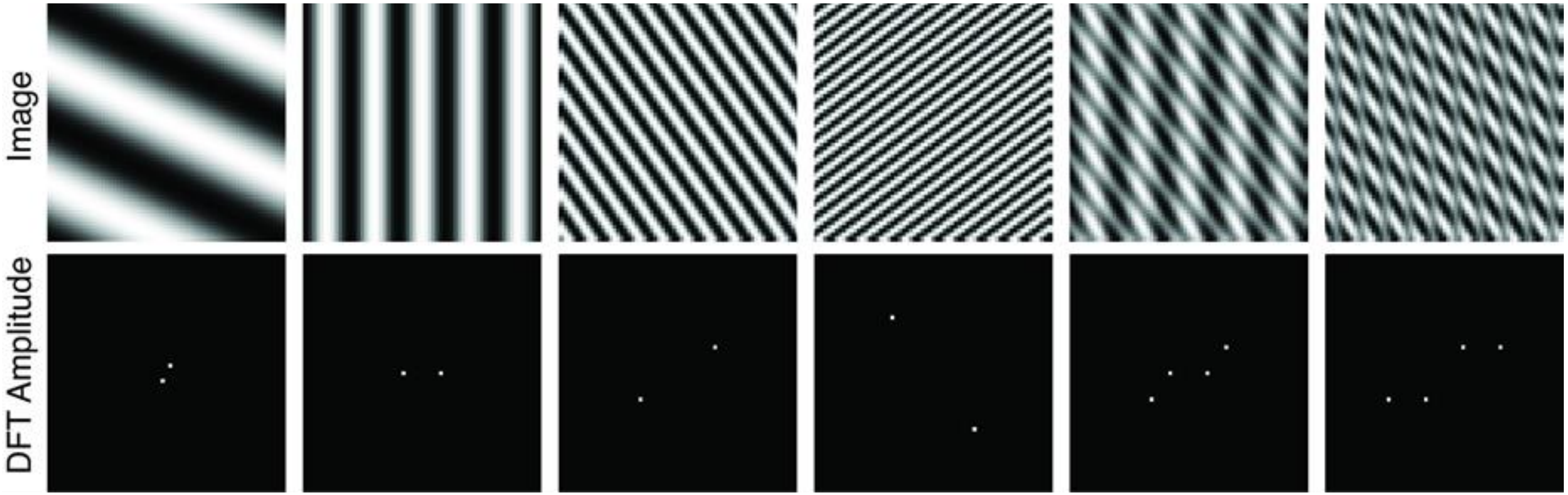
Hybrid = LowPass(im1) + HighPass(im2)



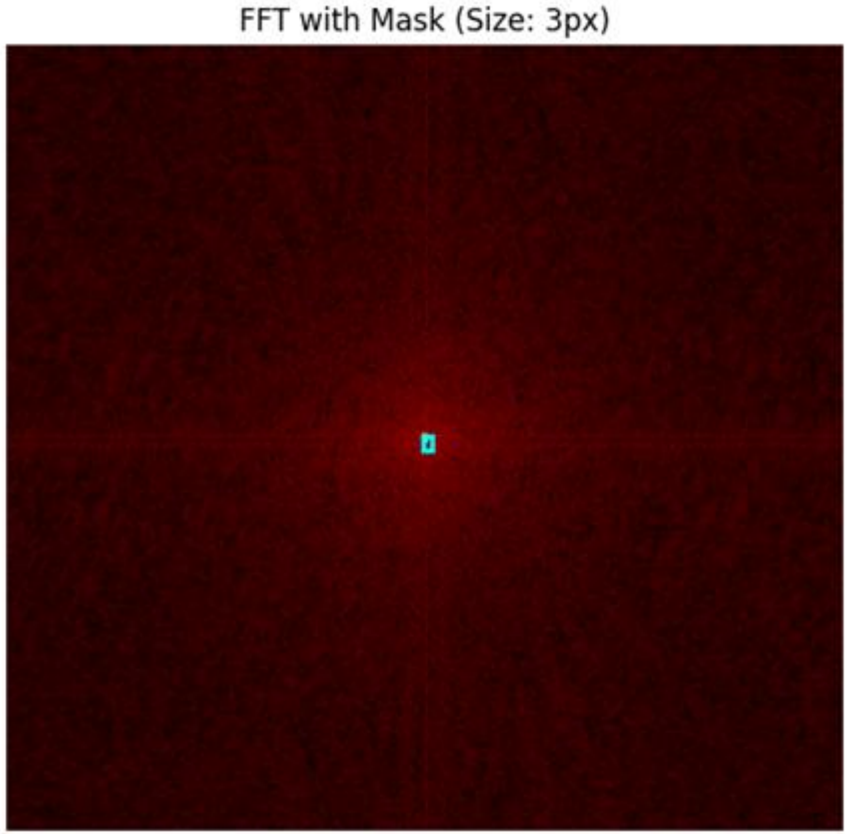
Fourier Transform

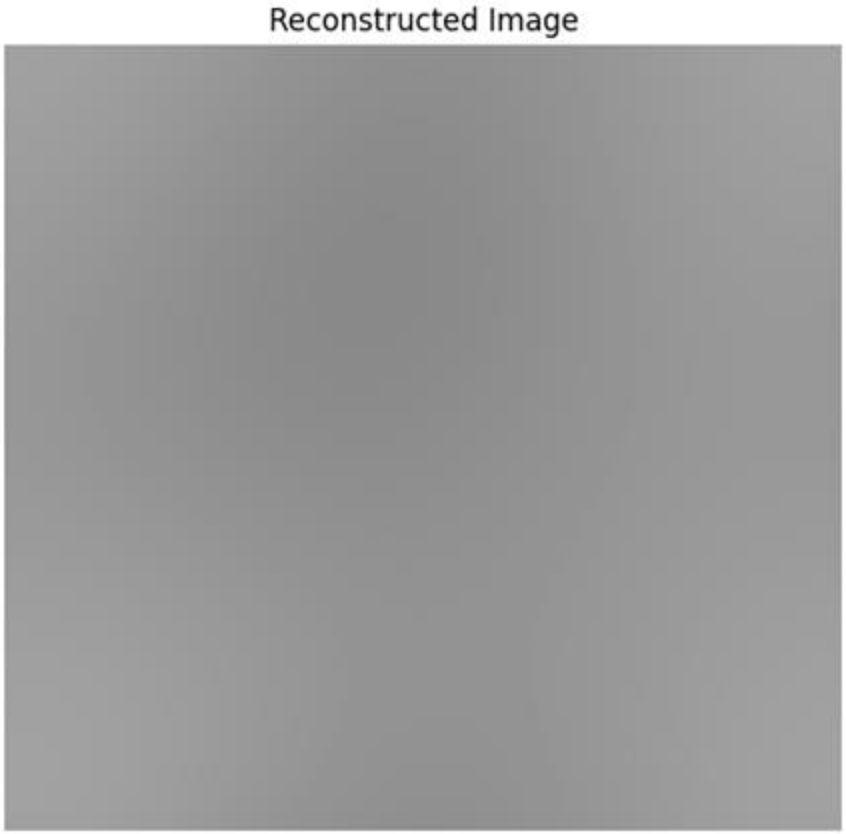
- Convert **spatial ⇒ frequency**
- "How much is each (directional) frequency?"
- Has **amplitude** and **phase**

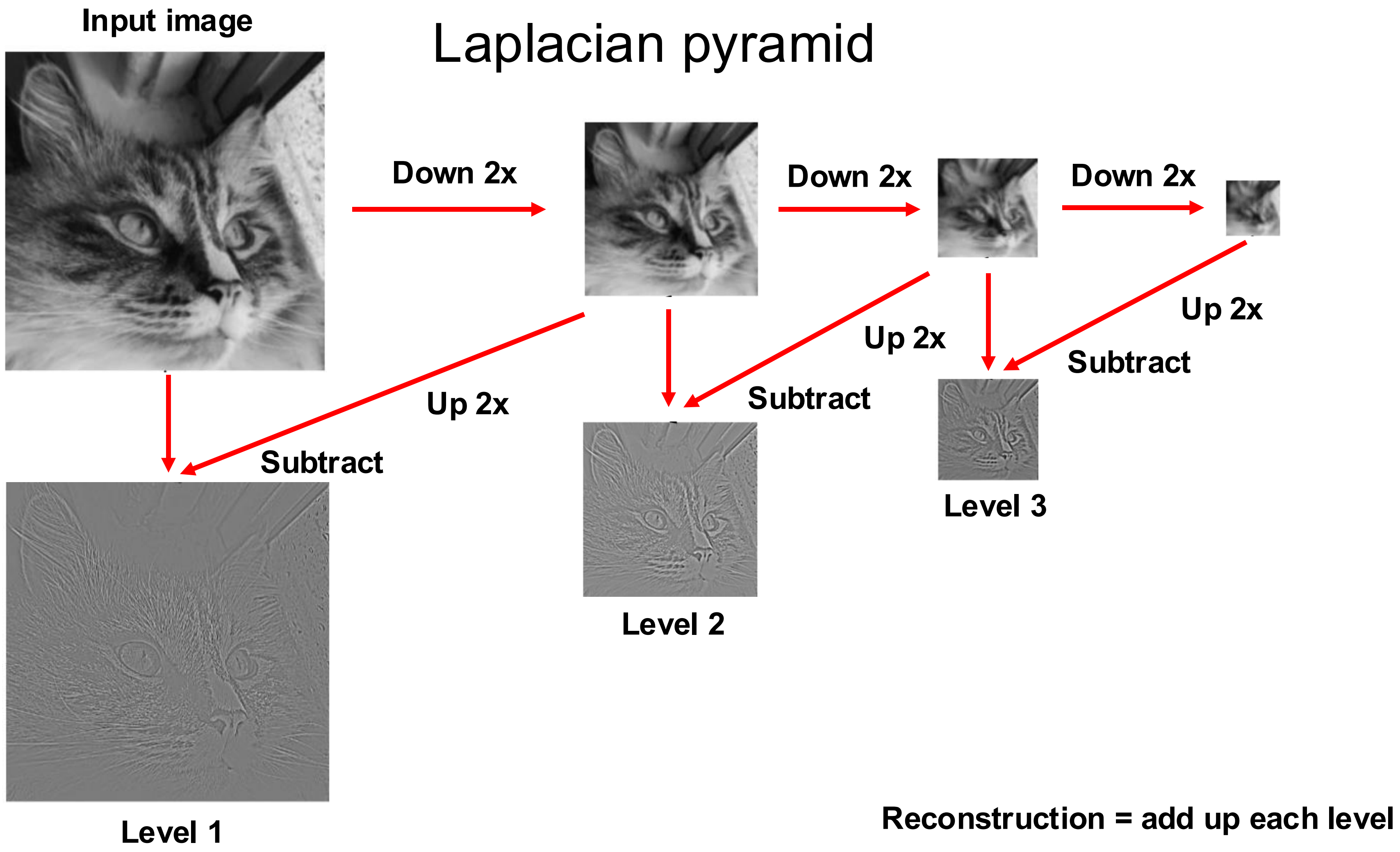
Amplitude



FFT Masking





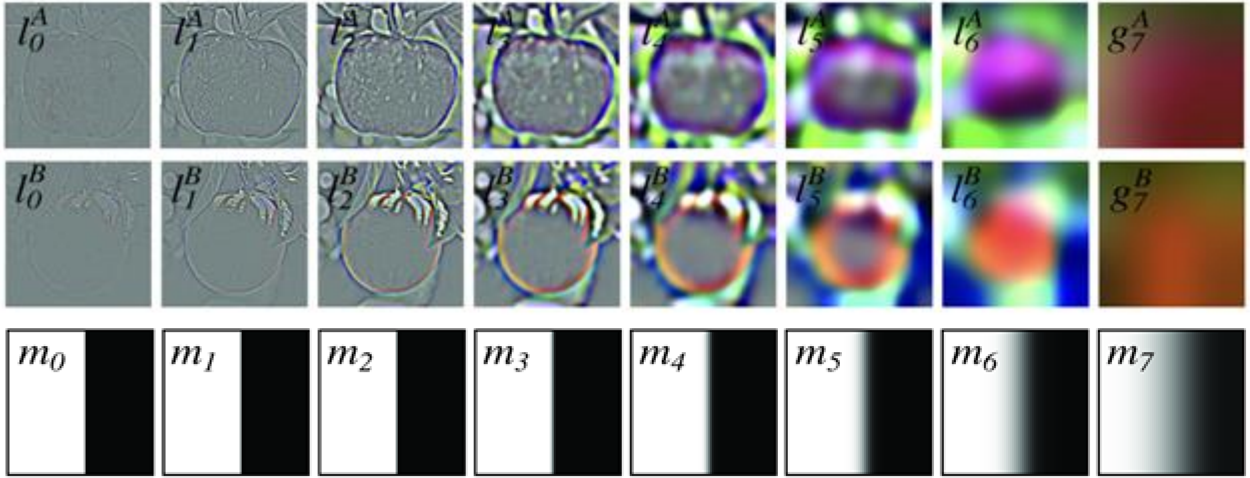


Input image
Laplacian pyramid
Down 2x
Down 2x
Down 2x
Up 2x
Subtract
Up 2x
Subtract
Up 2x
Subtract
Level 1
Level 2
Level 3
Reconstruction = add up each level

Image Blending

Application for Laplacian stack / pyramid

Blending at each “band” of the frequencies



Project 3A

Homogeneous Coordinates

Image Transformations

Homography

Interpolation

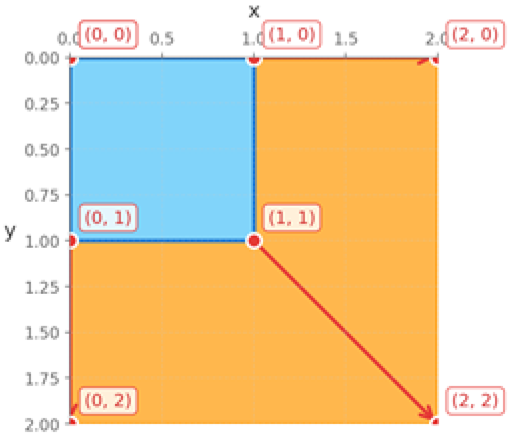
Image Transformations

Scale

Scale X

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \boxed{2} & 0 \\ 0 & \boxed{2} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

Scale Y



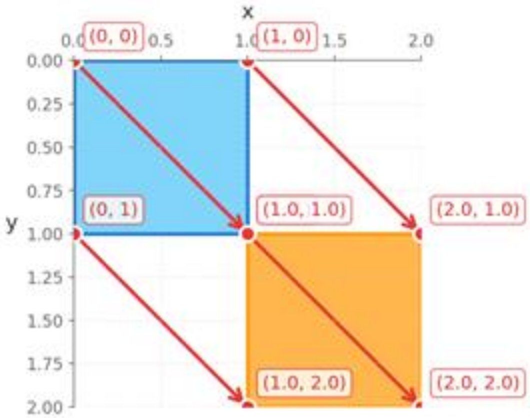
Translation (homogeneous)

Translate X

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

Delete

Translate Y



Rotation 30 deg + Translation

Rotate not at (0, 0)

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos 30° & -\sin 30° & \boxed{t_x} \\ \sin 30° & \cos 30° & \boxed{t_y} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

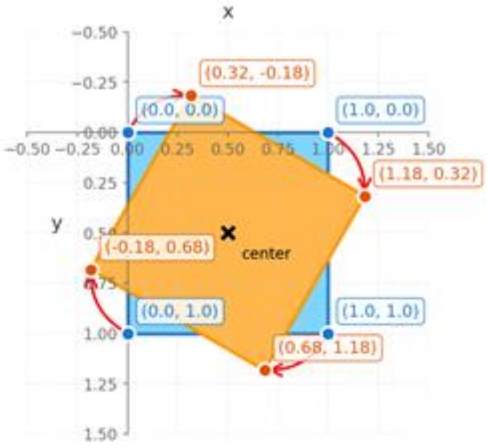
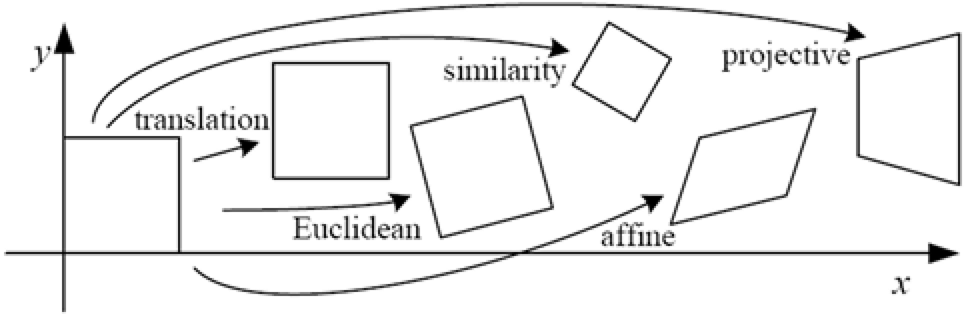


Image Transformations



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

Chat with your neighbor!

T/F: An affine transformation can map parallel lines to intersecting lines.

False, affine transformations preserve parallel lines

T/F: A homography/projective transformation can bend straight lines into curves.

False

T/F: The order of transformations matters – applying rotation then translation generally gives a different result than applying translation then rotation.

True

Homography

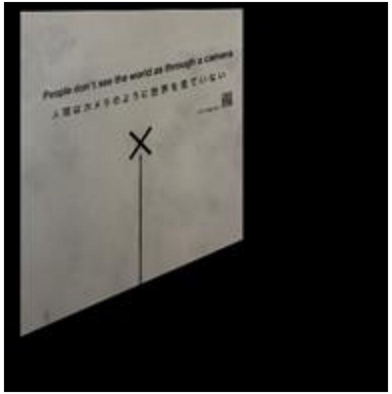
- All previous + **perspective distortion**
- Describes planar transformations (one plane to another)
- **Cannot represent nonlinear warps** (e.g. lens distortion, fisheye effect or independent object motion).

Homography

$$\begin{pmatrix} a & b & c \\ d & e & f \\ \boxed{g \quad h} & & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \\ \boxed{1} \end{pmatrix}$$

How to "lift" each point

Project on z = 1 plane



Chat with your neighbour!

T/F: Homography relates two images from a camera that moves.

False! If you move the "content" changes! It's not an image transformation.

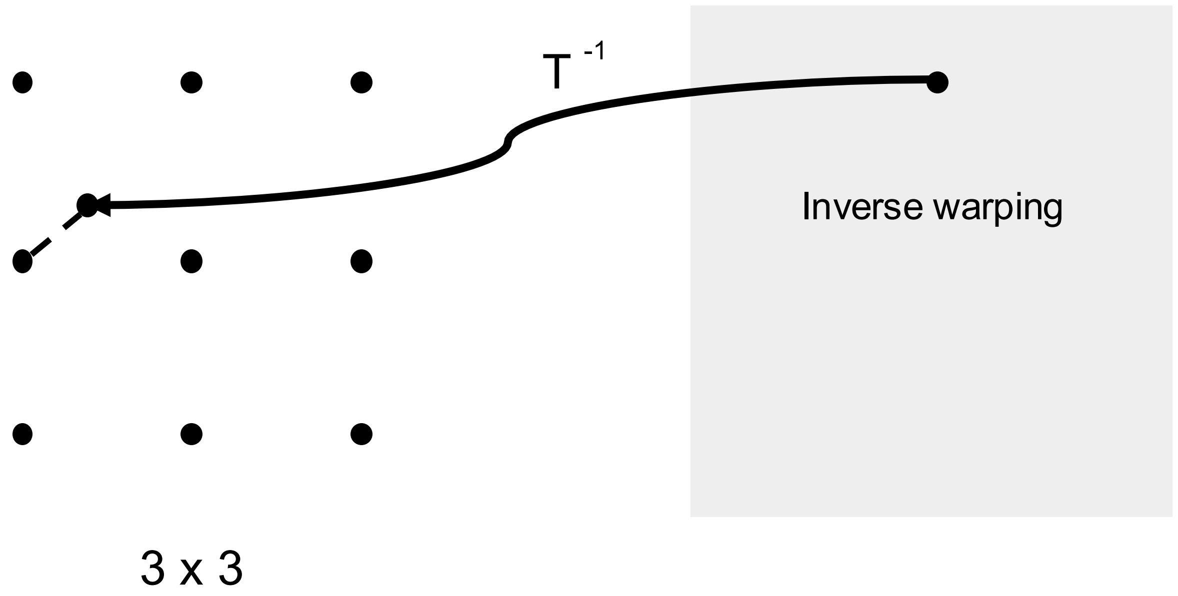
T/F: Keep the number of point correspondences at the **minimum** to avoid over-determining the homography.

False!

T/F: Applying Homography to a straight line never produces a curved line.

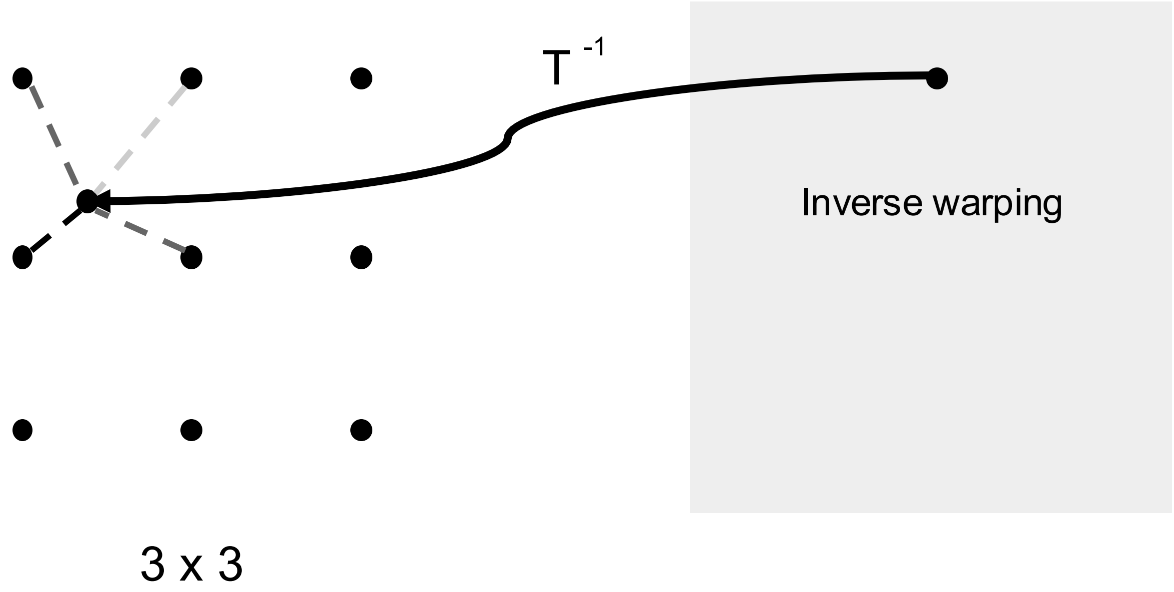
True

Nearest Neighbor Interpolation



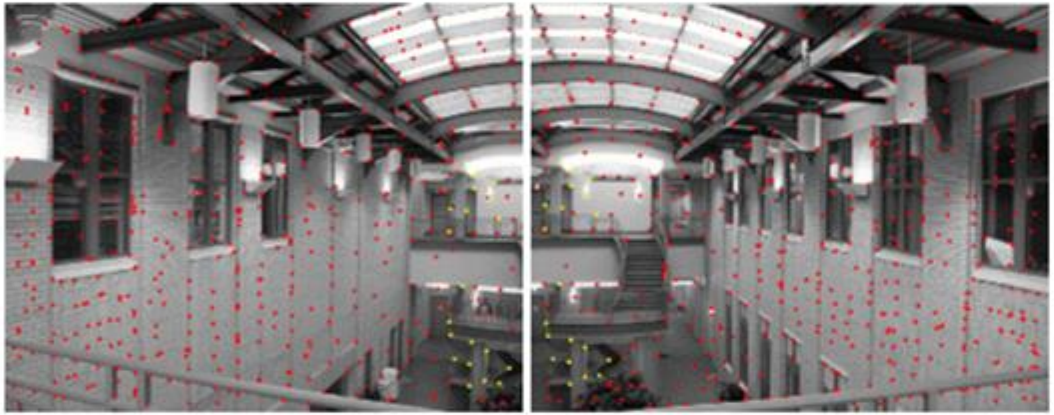
Pick closest (aka round)

Bilinear Interpolation



Weighted average of closest known points
(Closer weighs more)

Project 3B



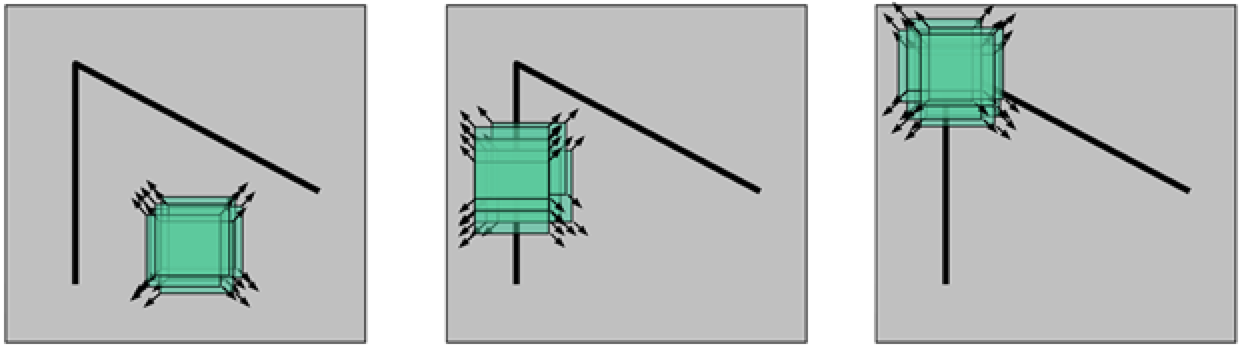
Feature Detection

RANSAC

Feature Detection (Harris)

- Interesting points = corners
- Corners = a lot of changes on both x and y
- Look at a location where both x & y edges are strong

$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$



No change

Change **except** in edge direction

Change in **all** directions

Feature Detection (Harris)

Find the direction with largest changes

$$M = \sum_{(x,y)\in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \longrightarrow M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

Some properties

$$\boxed{\begin{aligned} \det M &= \lambda_1 \lambda_2 \\ \operatorname{trace} M &= \lambda_1 + \lambda_2 \end{aligned}}$$

$$R = \frac{\det M}{\operatorname{Trace} M}$$

Larger is more “conery”

Chat with your neighbor

(T / F) Harris corner detection is perfectly invariant to 90-degree rotation

True

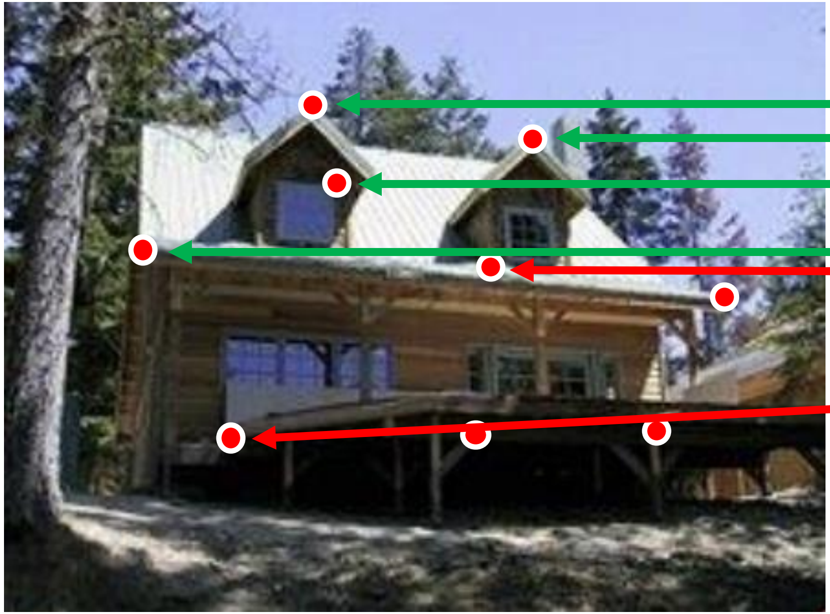
(T / F) Corner detection matches the points between images

False; it just finds interesting points.

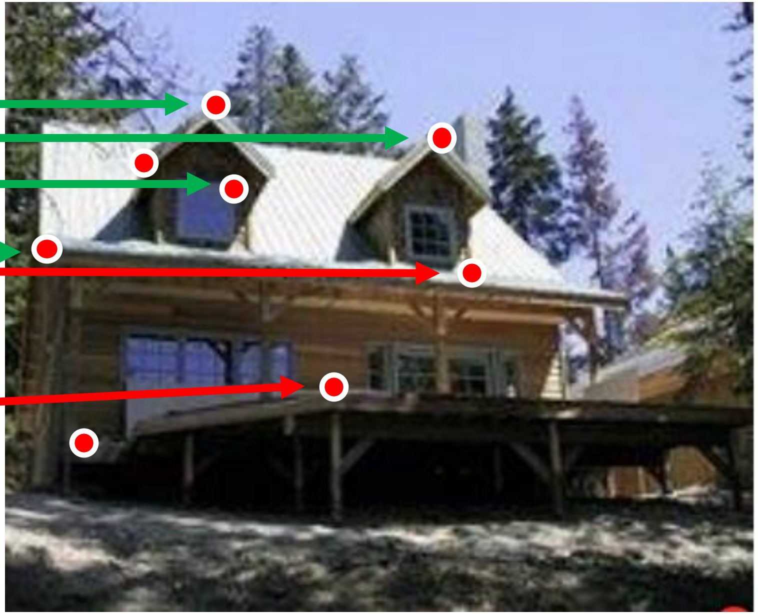
(T / F) Corners are pixels with a lot of changes in either x or y directions.

No; need both x and y!

Solve for homography



H



Solve homography!

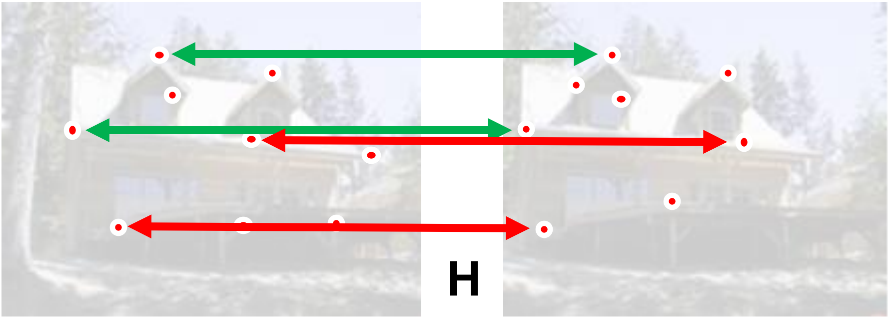
How to deal with some of them being really wrong?

RANSAC

Random **Sa**mple **C**onsensus

1. Select (minimum) pairs (at random)

2. Compute homography H



*why minimum pairs? Not maximum?

RANSAC

Random **Sa**mple **C**onsensus

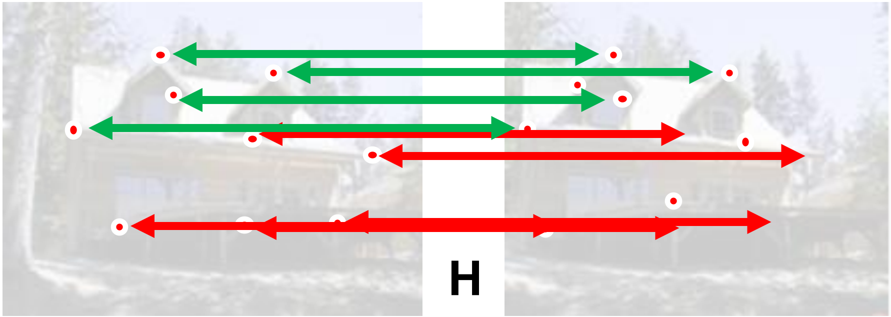
1. Select 4 pairs (at random)

2. Compute homography H

3. Check how well homography H explains
   (Count inliers, good matches)

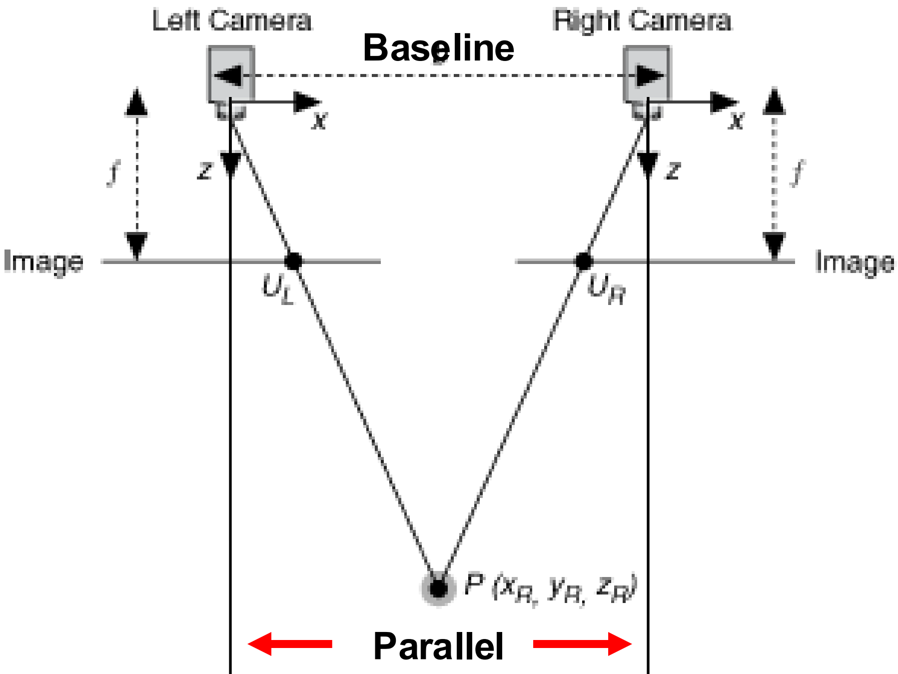
4. Keep the best H (most inliers)

5. Recompute H (with all inliers)



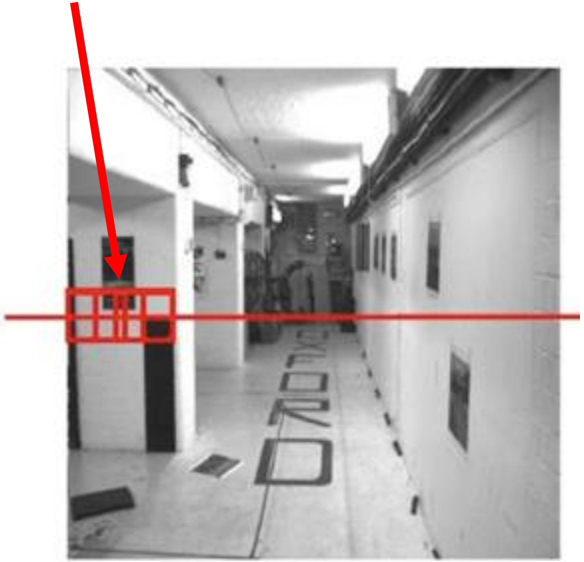
Stereo

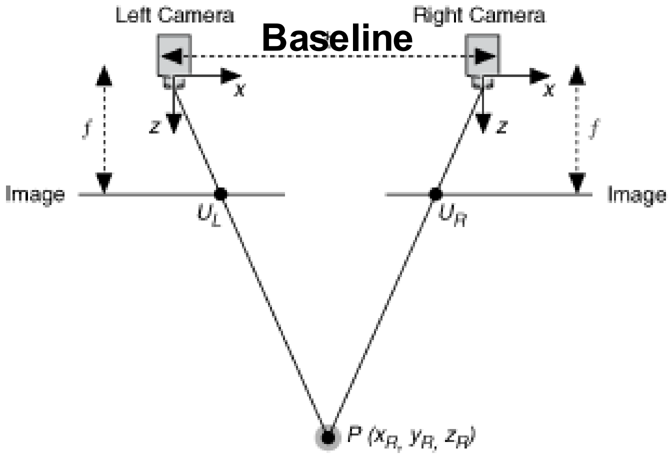
- **Parallel** pinhole cameras (same focal length)
- If know the baseline, can estimate depth!



Getting depth from stereo images

- Say we know: **f and baseline**
- Find correspondences (same points on images)
- Lie on **horizontal lines**





Good luck 🤞