# Computer Vision
## Image Stitching

**Dr. Mrinmoy Ghorai**

**Indian Institute of Information Technology**
**Sri City, Chittoor**

# Perspective and 3D Geometry

- **Camera models and Projective geometry**
  - What's the mapping between image and world coordiantes?
- **Projection Matrix and Camera calibration**
  - What's the projection matrix between scene and image coordiantes?
  - How to calibrate the projection matrix?
- **Single view metrology and Camera properties**
  - How can we measure the size of 3D objects in an image?
  - What are the important camera properties?
- **Photo stitching**
  - What's the mapping from two images taken without camera translation?
- **Epipolar Geometry and Stereo Vision**
  - What's the mapping from two images taken with camera translation?
- **Structure from motion**
  - How can we recover 3D points from multiple images?

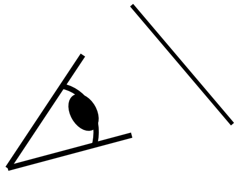
# This class: Image Stitching

- Combine two or more overlapping images to make one larger image

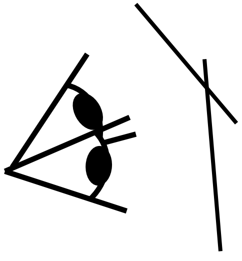
Slide credit: [Vaibhav Vaish]

# Idea: projecting images onto a common plane

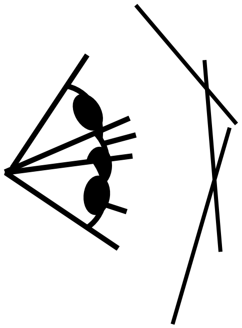
# Idea: projecting images onto a common plane



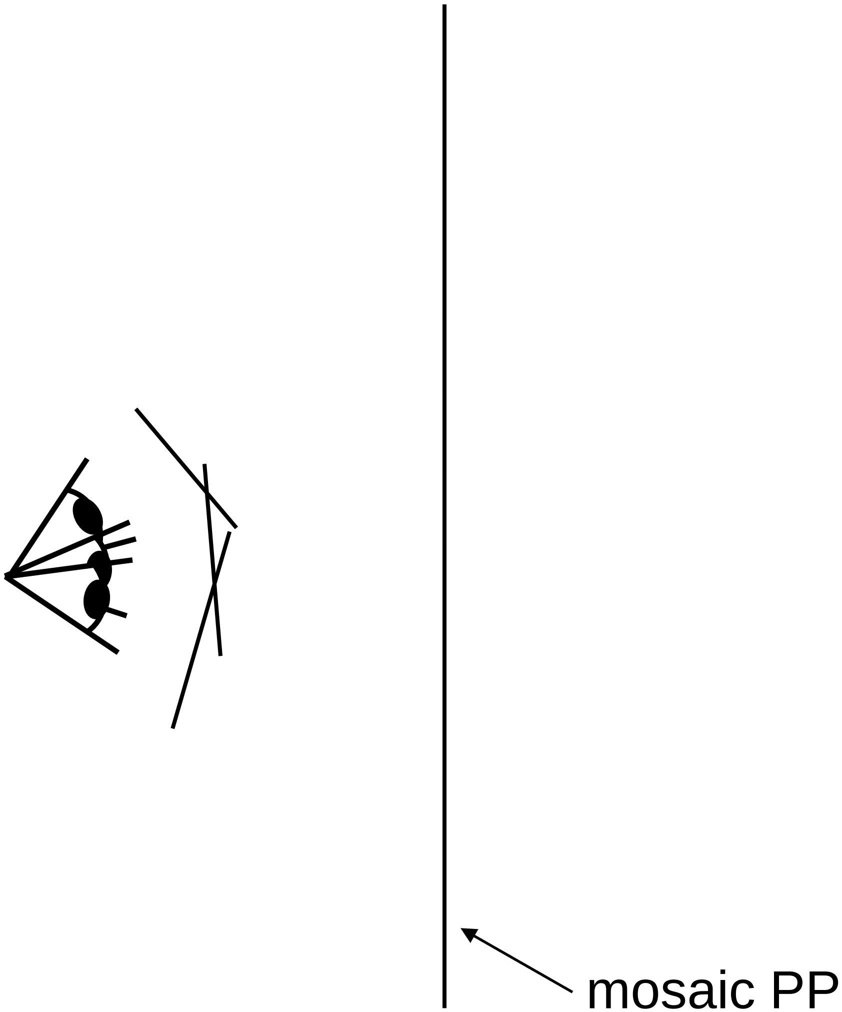
# Idea: projecting images onto a common plane



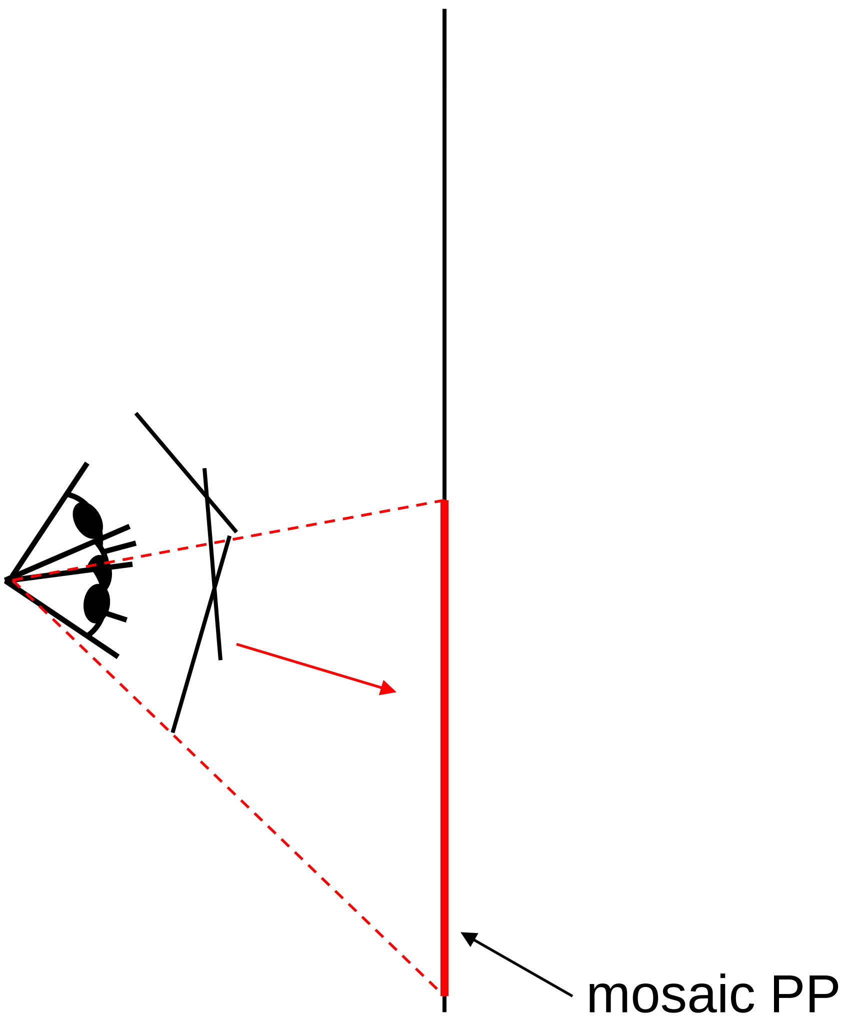
# Idea: projecting images onto a common plane



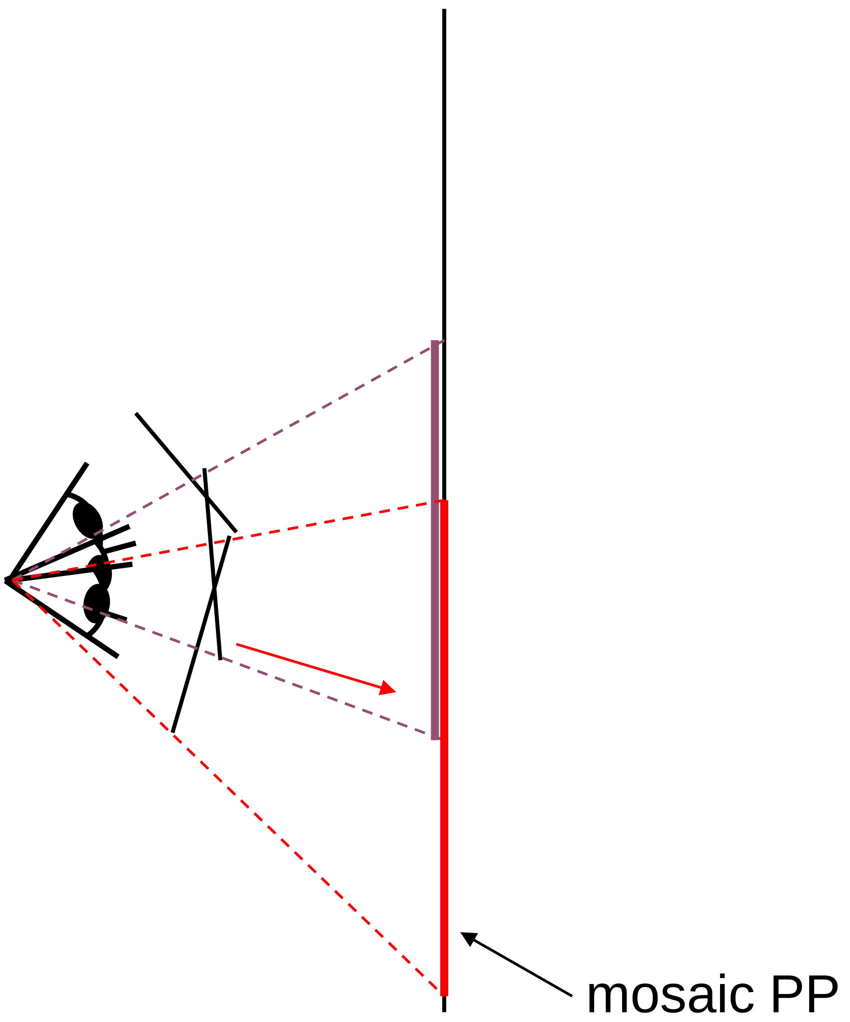
# Idea: projecting images onto a common plane



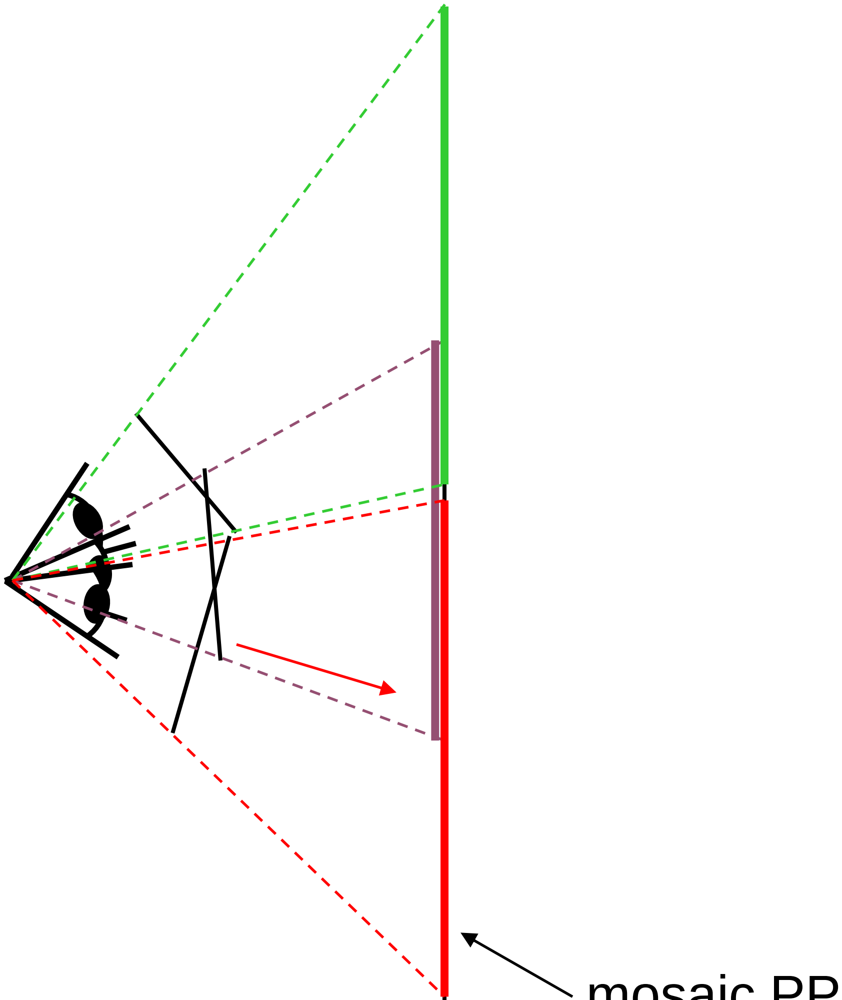
# Idea: projecting images onto a common plane



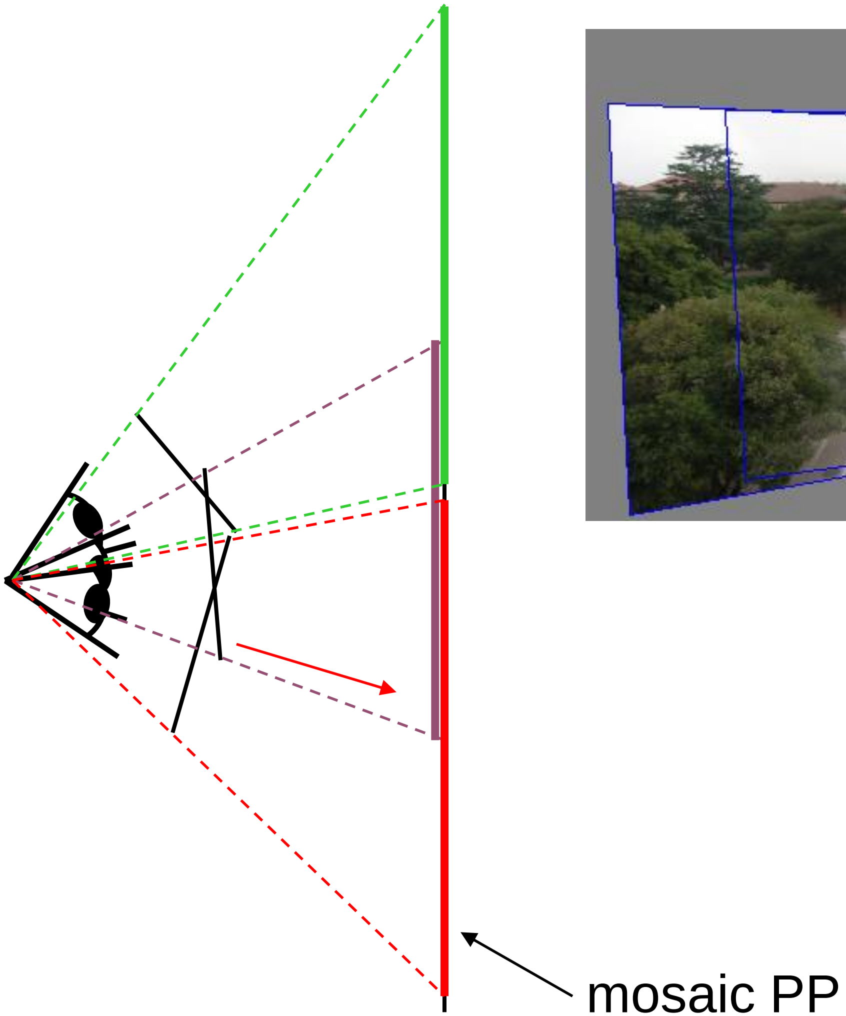
# Idea: projecting images onto a common plane

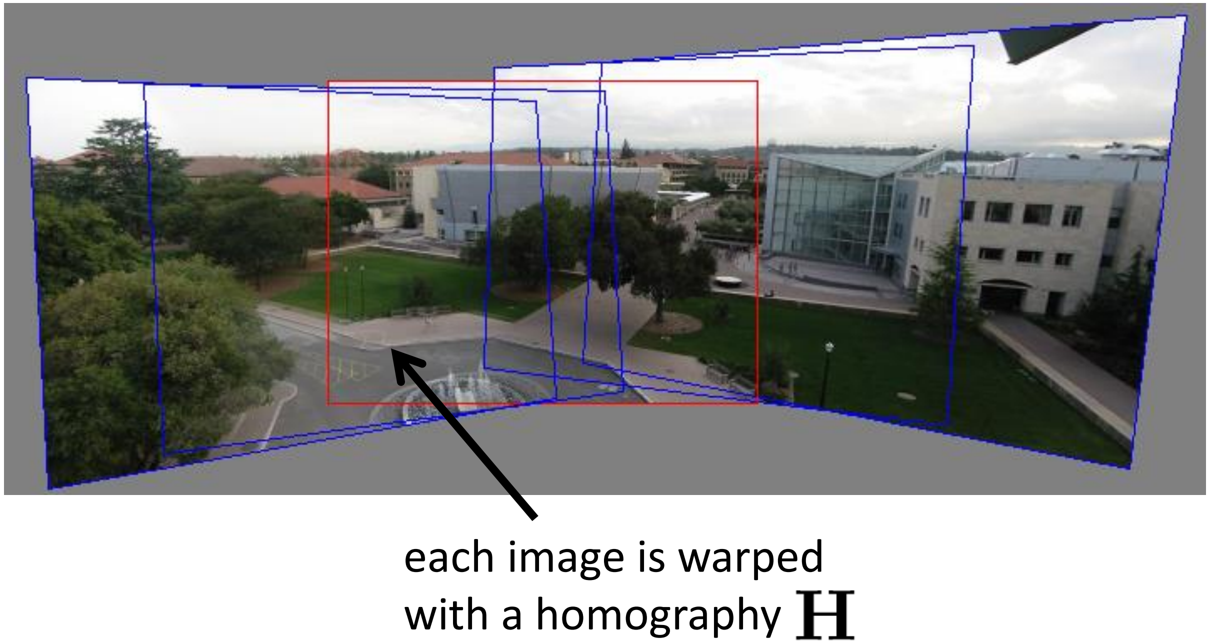


# Idea: projecting images onto a common plane



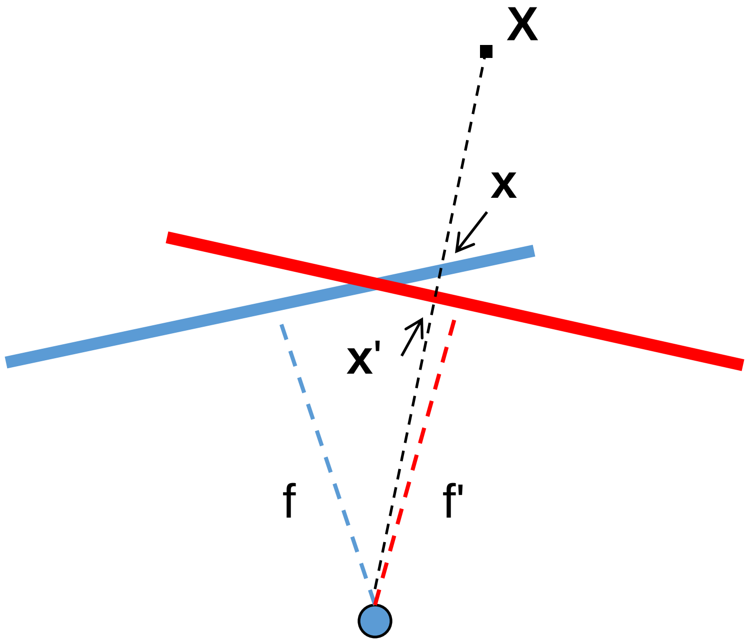
# Idea: projecting images onto a common plane



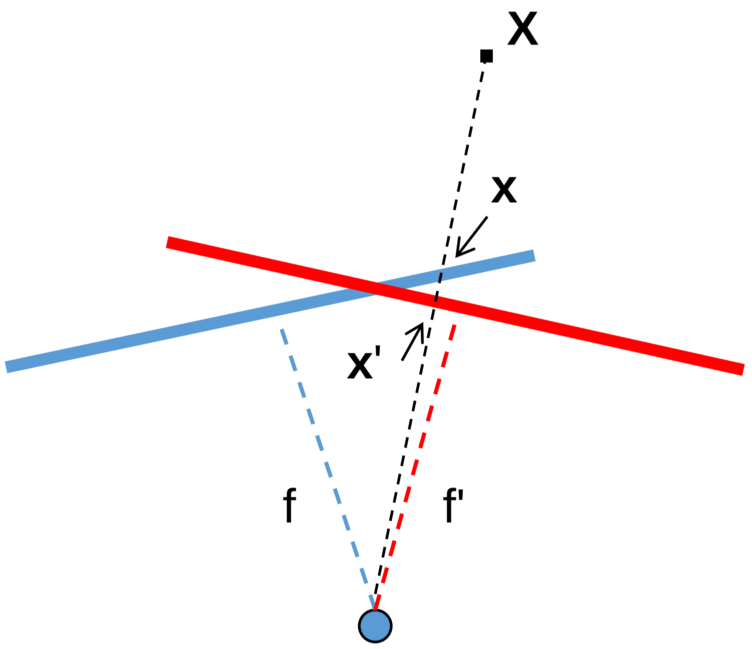
# Problem set-up

- x = K [R t] X
- x' = K' [R' t'] X
- t=t'=0



# Problem set-up

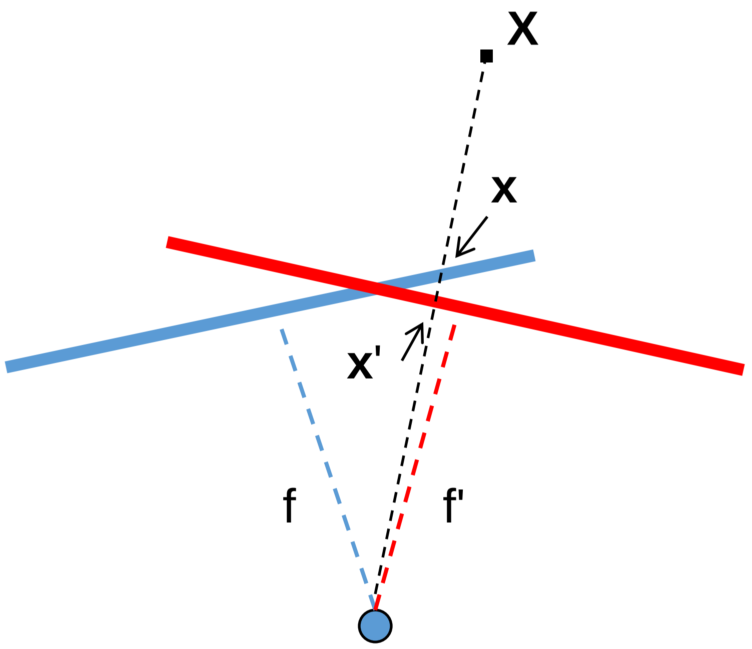
- $x = K\ [R\ t]\ X$
- $x' = K'\ [R'\ t']\ X$
- $t=t'=0$




- $x'=Hx$ where $H = K'\ R'\ R^{-1}\ K^{-1}$

# Problem set-up

- $x = K [R\ t] X$
- $x' = K' [R'\ t'] X$
- $t=t'=0$




- $x'=Hx$ where $H = K' R' R^{-1} K^{-1}$
- Typically only R and f will change (4 parameters), but, in general, H has 8 parameters

# Image Stitching Algorithm Overview

1. Detect keypoints (e.g., SIFT Detector)

2. Match keypoints (e.g., $1^{st}/2^{nd}$ NN < thresh)

3. Estimate homography with four matched keypoints (using RANSAC)

4. Combine images

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$w' = uh_7 + vh_8 + h_9$$

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$w' = uh_7 + vh_8 + h_9$$

$$w'u' = (uh_7 + vh_8 + h_9)u' = uh_1 + vh_2 + h_3$$

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

## Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$w' = uh_7 + vh_8 + h_9$$

$$w'u' = (uh_7 + vh_8 + h_9)u' = uh_1 + vh_2 + h_3 \quad \Rightarrow -uh_1 - vh_2 - h_3 + uu'h_7 + u'vh_8 + u'h_9 = 0$$

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

## Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$w' = uh_7 + vh_8 + h_9$$

$$w'u' = (uh_7 + vh_8 + h_9)u' = uh_1 + vh_2 + h_3 \quad \Rightarrow -uh_1 - vh_2 - h_3 + uu'h_7 + u'vh_8 + u'h_9 = 0$$

$$w'v' = (uh_7 + vh_8 + h_9)v' = uh_4 + vh_5 + h_6$$

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

$$w' = uh_7 + vh_8 + h_9$$

$$w'u' = (uh_7 + vh_8 + h_9)u' = uh_1 + vh_2 + h_3 \quad \Rightarrow -uh_1 - vh_2 - h_3 + uu'h_7 + u'vh_8 + u'h_9 = 0$$

$$w'v' = (uh_7 + vh_8 + h_9)v' = uh_4 + vh_5 + h_6 \quad \Rightarrow -uh_4 - vh_5 - h_6 + uv'h_7 + vv'h_8 + v'h_9 = 0$$

# Computing homography

Assume we have four matched points: How do we compute homography **H**?

Direct Linear Transformation (DLT)

$$\mathbf{x}' = \mathbf{H}\mathbf{x} \qquad \mathbf{x}' = \begin{bmatrix} w'u' \\ w'v' \\ w' \end{bmatrix} \qquad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}$$

$$\begin{bmatrix} -u & -v & -1 & 0 & 0 & 0 & uu' & vu' & u' \\ 0 & 0 & 0 & -u & -v & -1 & uv' & vv' & v' \end{bmatrix} \mathbf{h} = \mathbf{0} \qquad \mathbf{h} = \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix}$$

# Computing homography

Direct Linear Transform

$$\begin{bmatrix} -u_1 & -v_1 & -1 & 0 & 0 & 0 & u_1 u_1' & v_1 u_1' & u_1' \\ 0 & 0 & 0 & -u_1 & -v_1 & -1 & u_1 v_1' & v_1 v_1' & v_1' \\ & & & & \vdots & & & & \\ 0 & 0 & 0 & -u_n & -v_n & -1 & u_n v_n' & v_n v_n' & v_n' \end{bmatrix} \mathbf{h} = \mathbf{0} \Rightarrow \mathbf{A}\mathbf{h} = \mathbf{0}$$

# Computing homography

## Direct Linear Transform

$$\begin{bmatrix} -u_1 & -v_1 & -1 & 0 & 0 & 0 & u_1u_1' & v_1u_1' & u_1' \\ 0 & 0 & 0 & -u_1 & -v_1 & -1 & u_1v_1' & v_1v_1' & v_1' \\ & & & & \vdots & & & & \\ 0 & 0 & 0 & -u_n & -v_n & -1 & u_nv_n' & v_nv_n' & v_n' \end{bmatrix}\mathbf{h} = \mathbf{0} \Rightarrow \mathbf{Ah} = \mathbf{0}$$

- Apply SVD: $\boldsymbol{UDV}^T = \boldsymbol{A}$
- $\boldsymbol{h} = \boldsymbol{V}_{\text{smallest}}$ (column of $\boldsymbol{V}$ corr. to smallest singular value)

Explanations of [SVD]() and [solving homogeneous linear systems]()

# Computing homography

## Direct Linear Transform

$$\begin{bmatrix} -u_1 & -v_1 & -1 & 0 & 0 & 0 & u_1u_1' & v_1u_1' & u_1' \\ 0 & 0 & 0 & -u_1 & -v_1 & -1 & u_1v_1' & v_1v_1' & v_1' \\ & & & & \vdots & & & & \\ 0 & 0 & 0 & -u_n & -v_n & -1 & u_nv_n' & v_nv_n' & v_n' \end{bmatrix} \mathbf{h} = \mathbf{0} \Rightarrow \mathbf{A}\mathbf{h} = \mathbf{0}$$

- Apply SVD: $\boldsymbol{UDV}^T = \boldsymbol{A}$
- $\boldsymbol{h} = \boldsymbol{V}_{\text{smallest}}$ (column of $\boldsymbol{V}$ corr. to smallest singular value)

$$\mathbf{h} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_9 \end{bmatrix} \quad \mathbf{H} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}$$

Explanations of [SVD] and [solving homogeneous linear systems]

# Computing homography

Assume we have matched points with outliers:

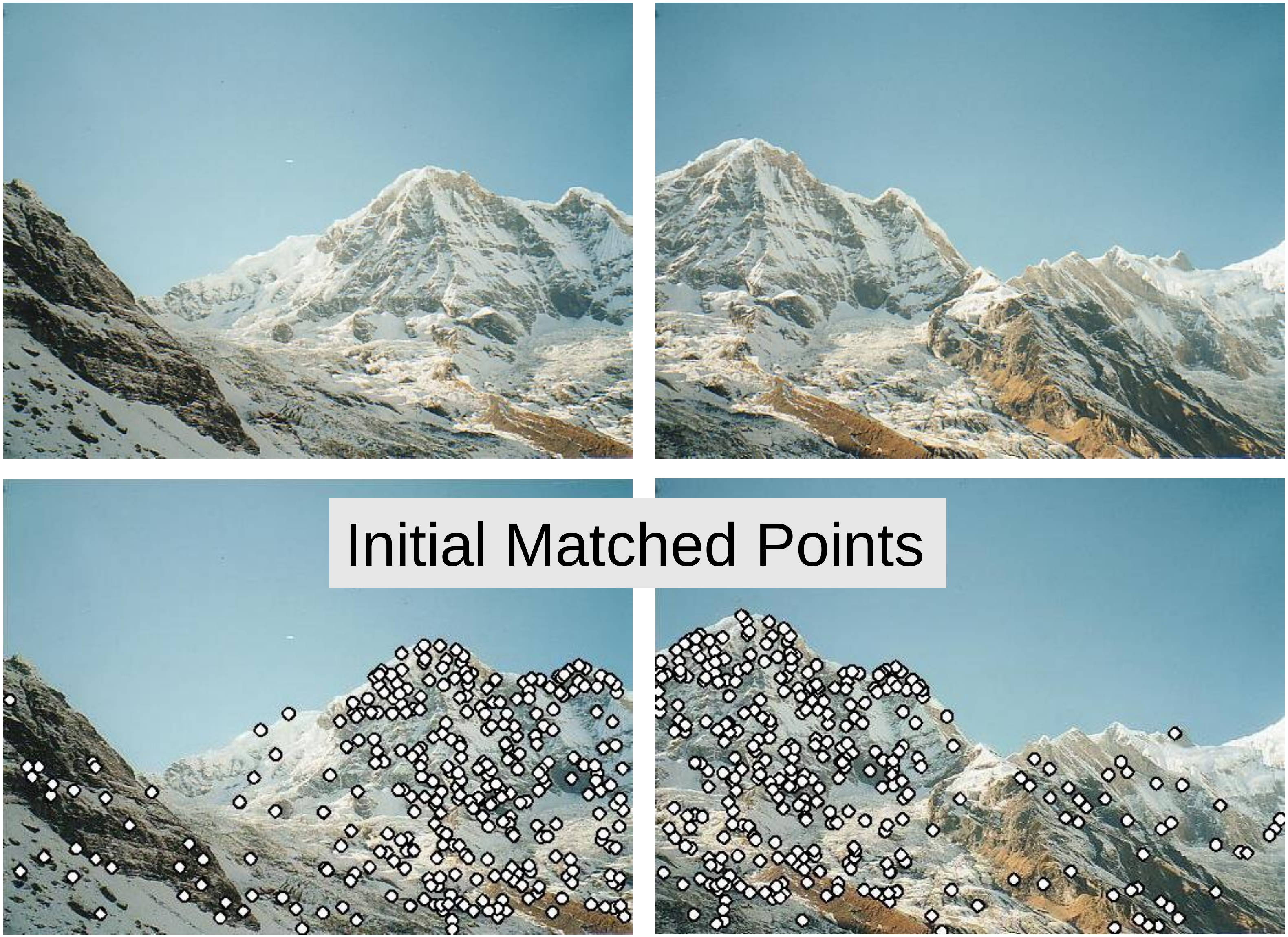
How do we compute homography **H**?

Automatic Homography Estimation with RANSAC

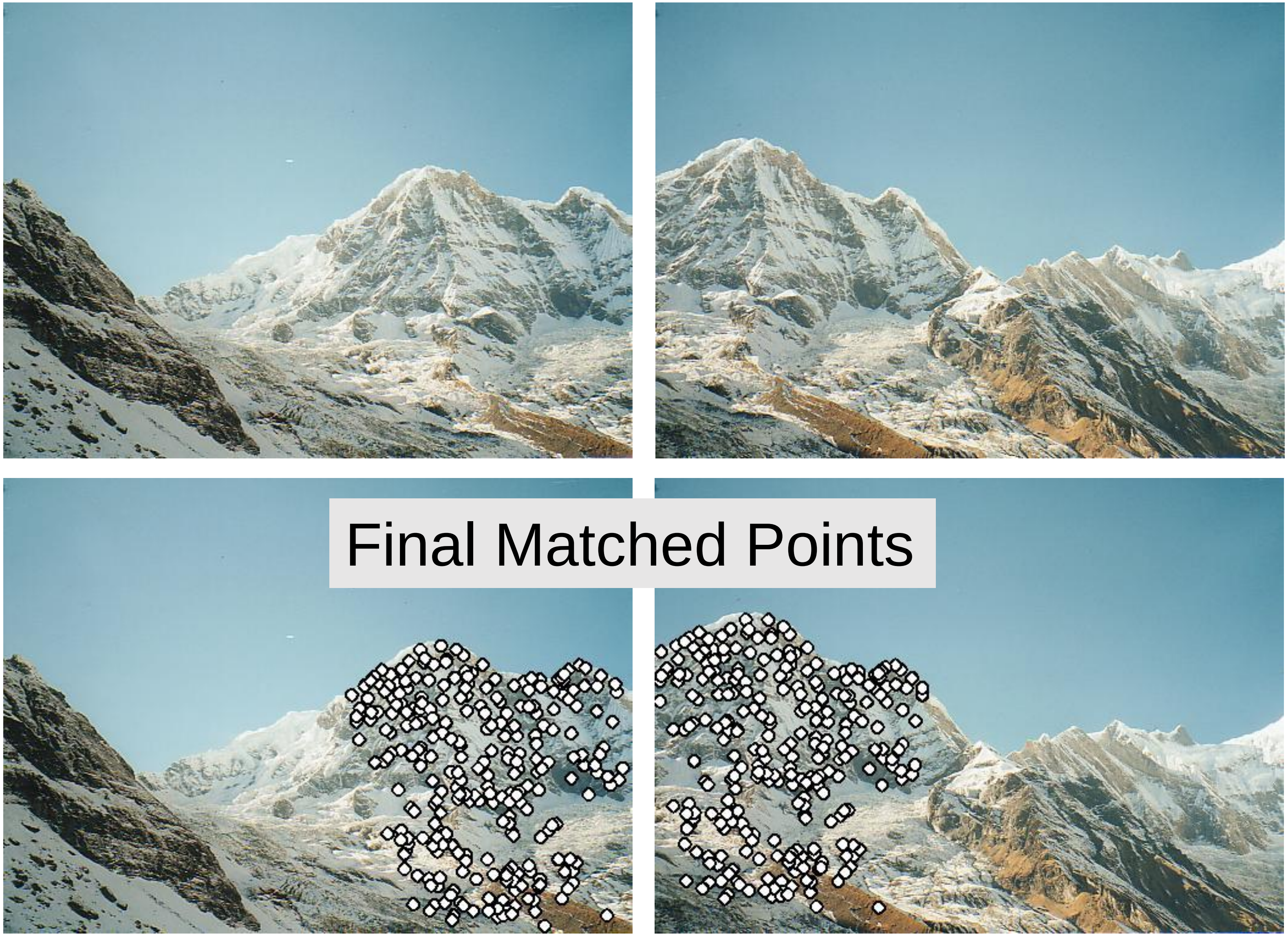
1. Choose 4 random potential matches
2. Compute **H** using Direct Linear Transformation
3. Project points from **x** to **x**' for each potentially matching pair: $\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i$
4. Count points with projected distance < t
   - E.g., t = 3 pixels
5. Repeat steps 1-4 *N* times
   - Choose **H** with most inliers

HZ Tutorial 'CVPR99

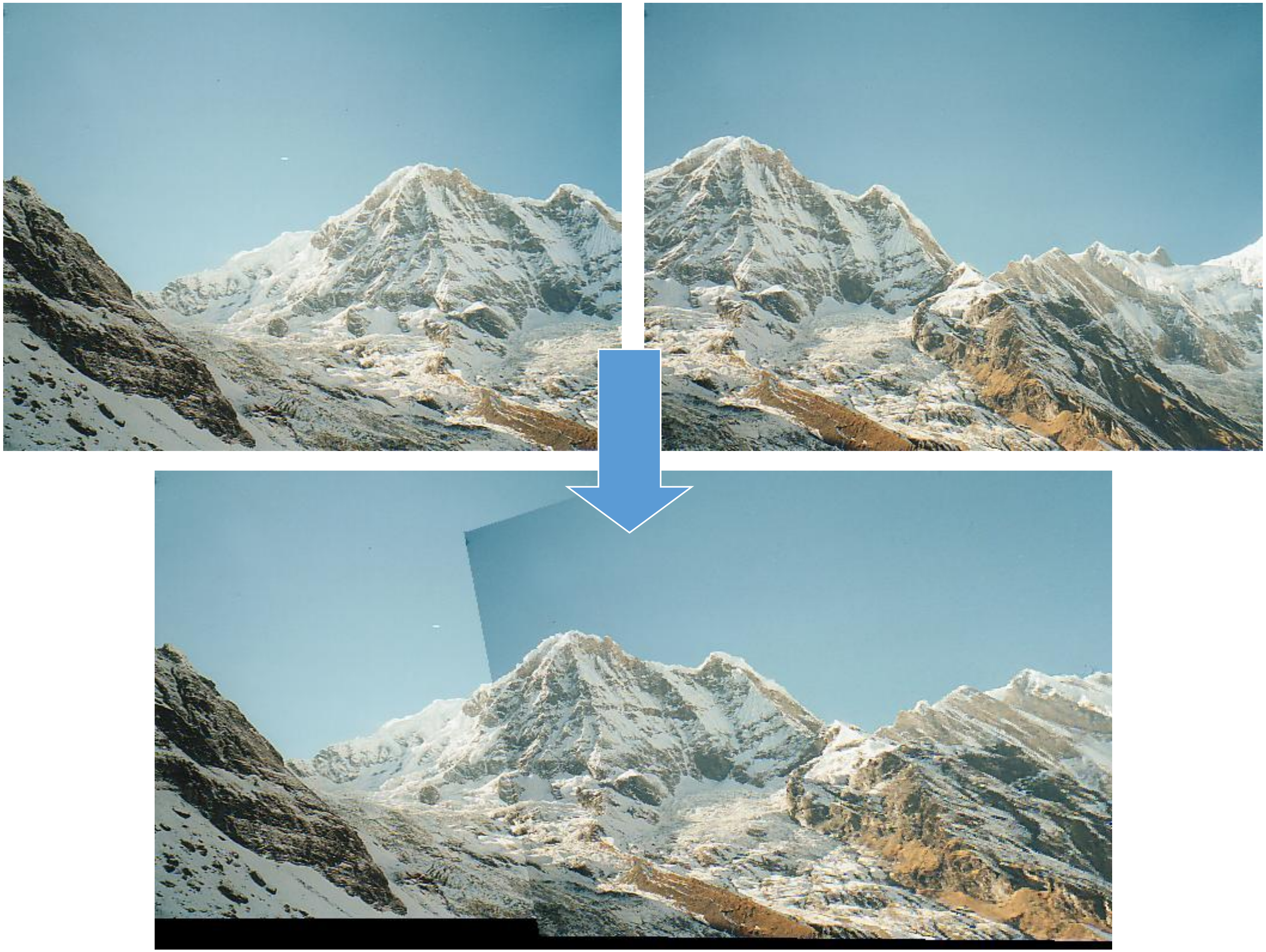
# RANSAC for Homography



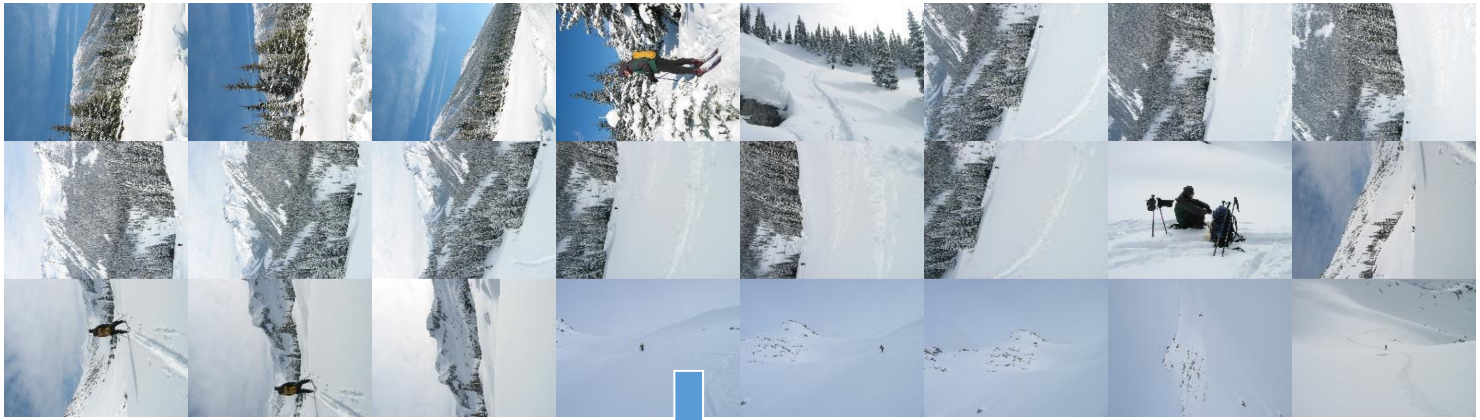
# RANSAC for Homography

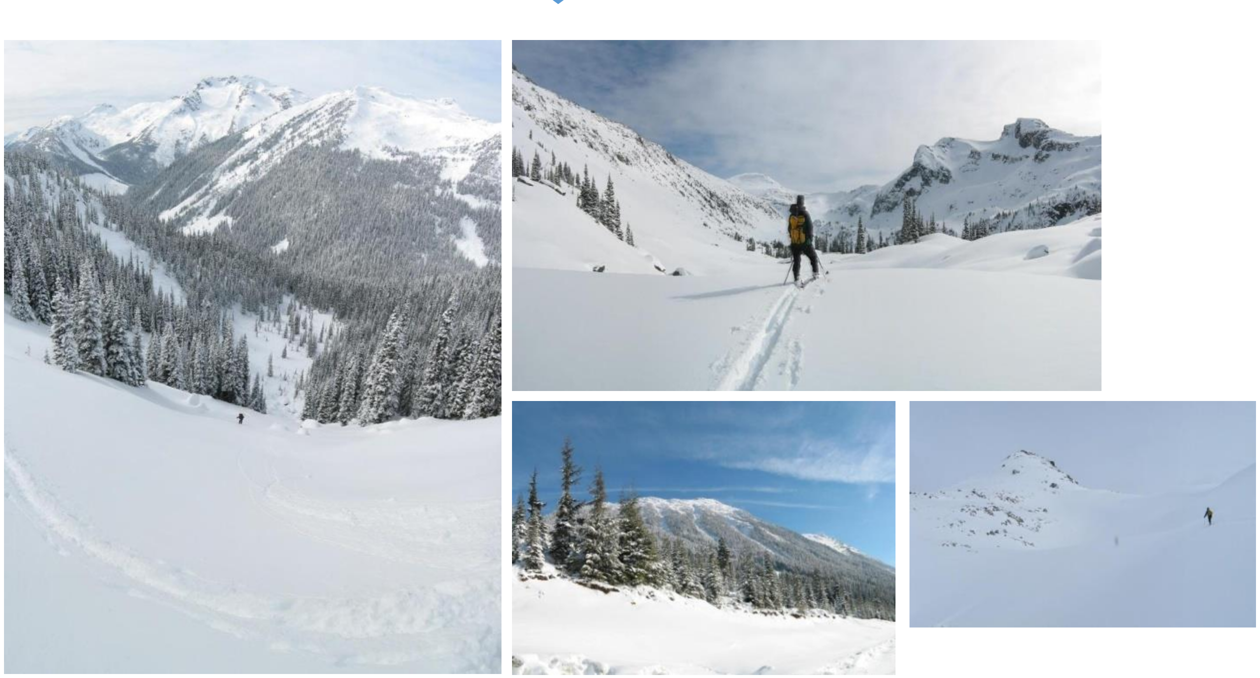


# RANSAC for Homography



# Application: Recognizing Panoramas



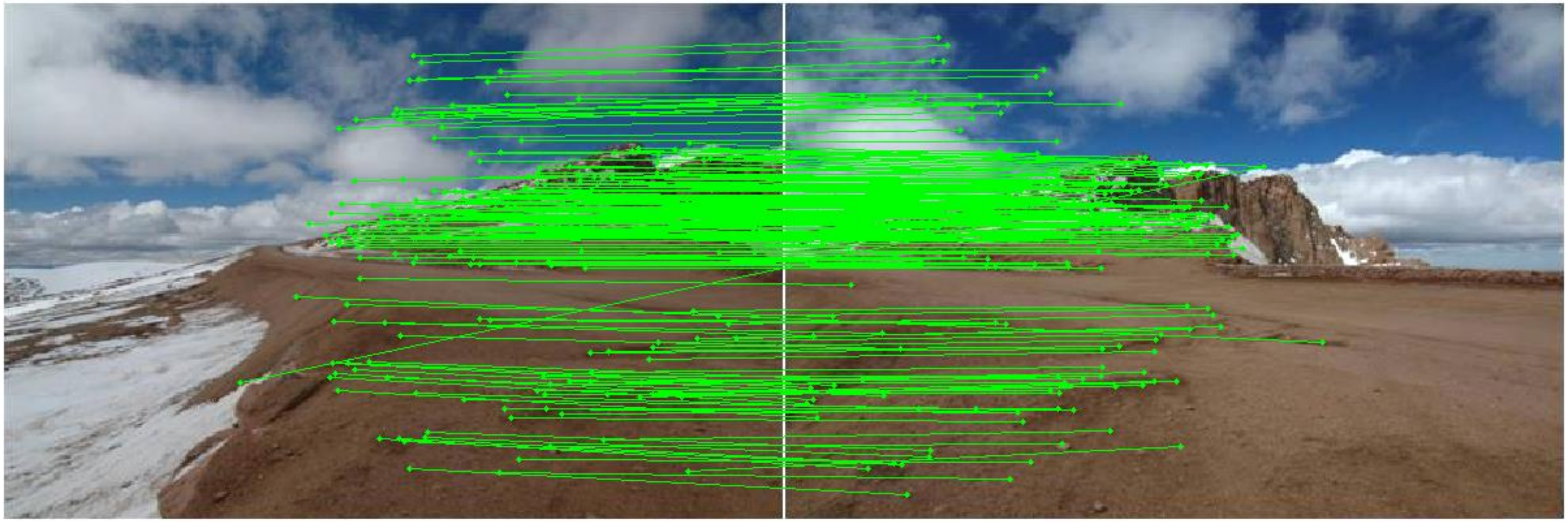


Some of following material from Brown and Lowe 2003 talk

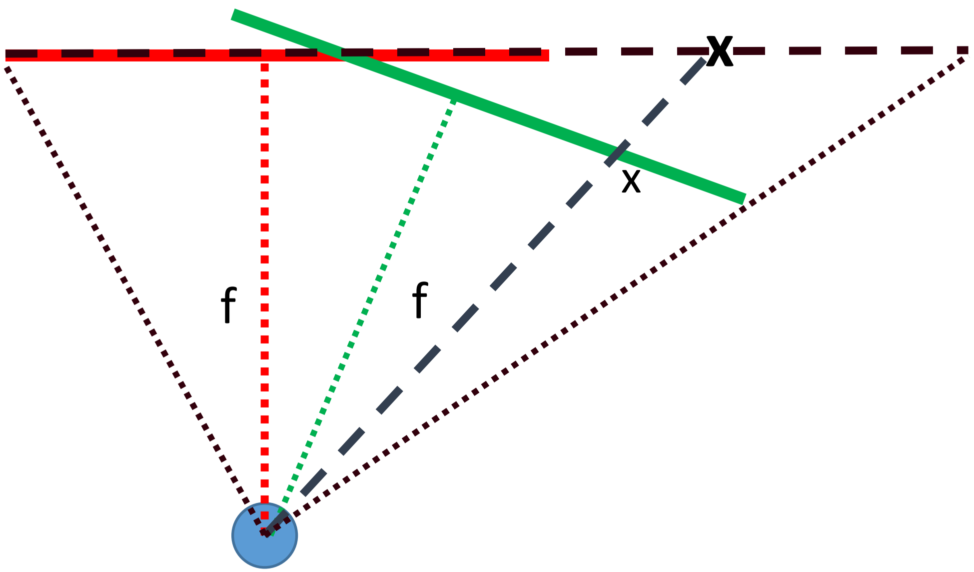
Brown and Lowe 2003, 2007

# Choosing a Projection Surface

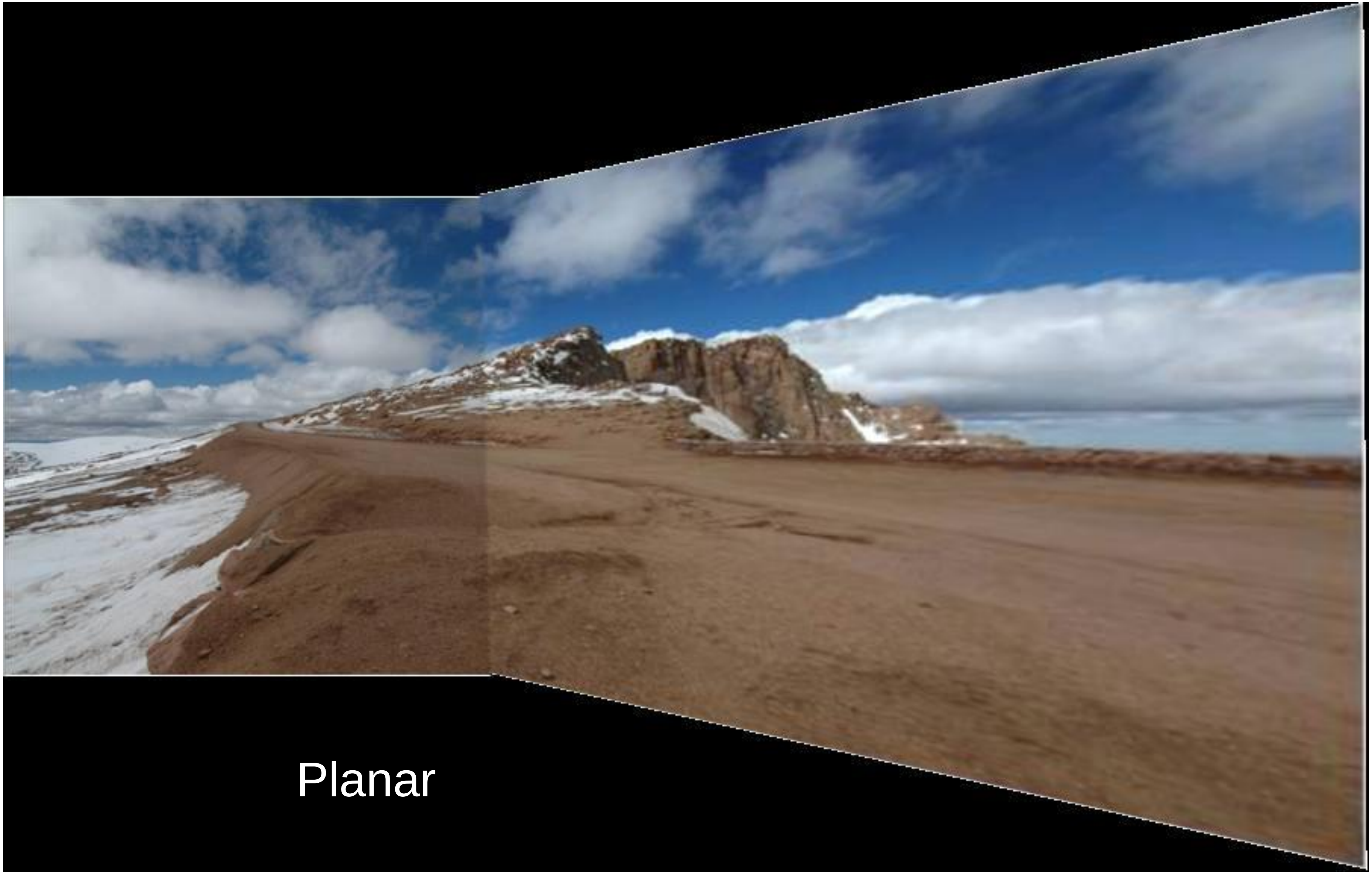
Many to choose: planar, cylindrical, spherical, cubic, etc.



# Planar Mapping



# Planar Projection

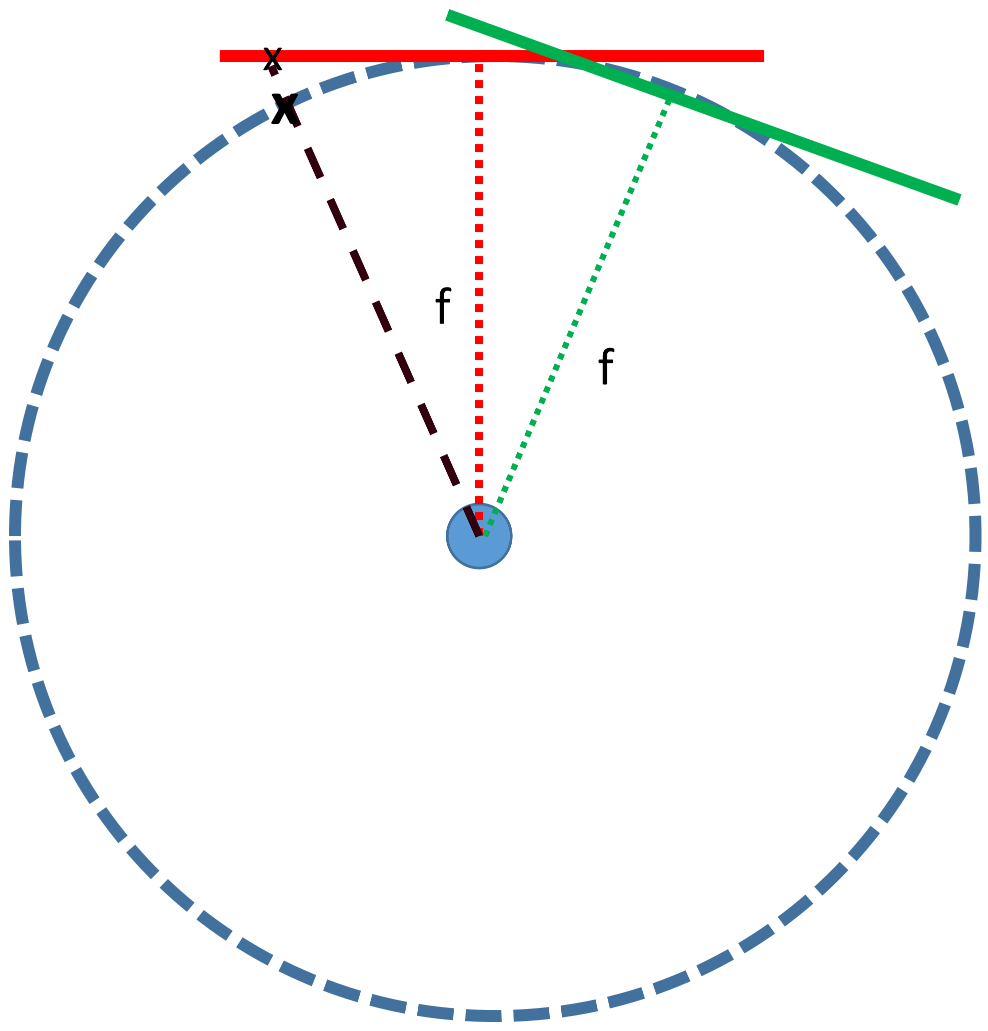



Photos by Russ Hewett

# Planar Projection

Planar

# Cylindrical Mapping



# Cylindrical Projection

Cylindrical

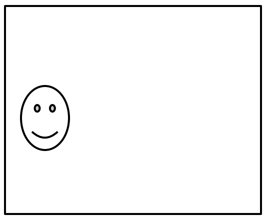
# Cylindrical Projection

Cylindrical


Planar

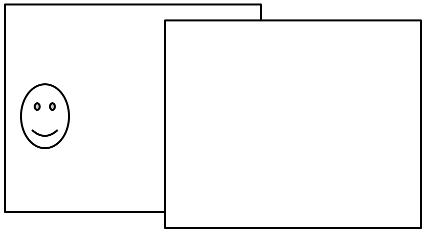

Cylindrical

# Assembling the panorama



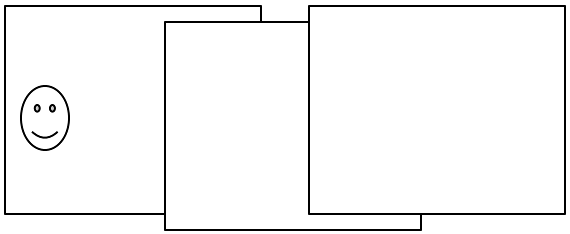
- Stitch pairs together, blend, then crop

# Assembling the panorama



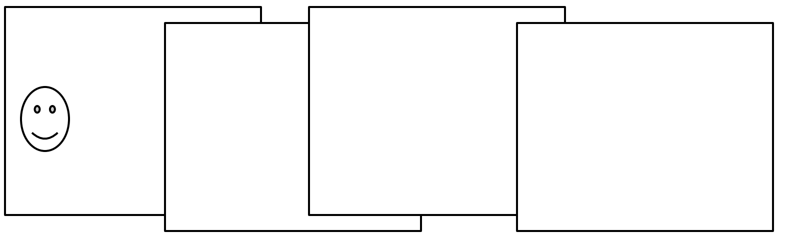
- Stitch pairs together, blend, then crop

# Assembling the panorama



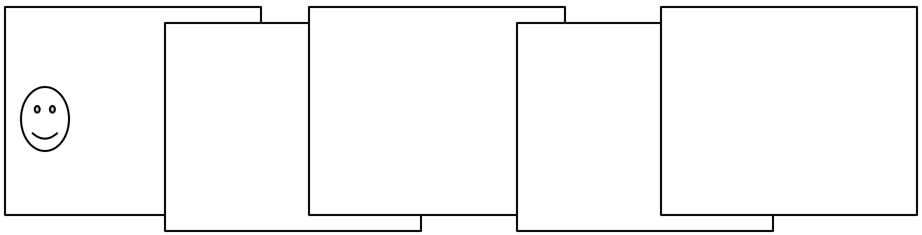
- Stitch pairs together, blend, then crop

# Assembling the panorama



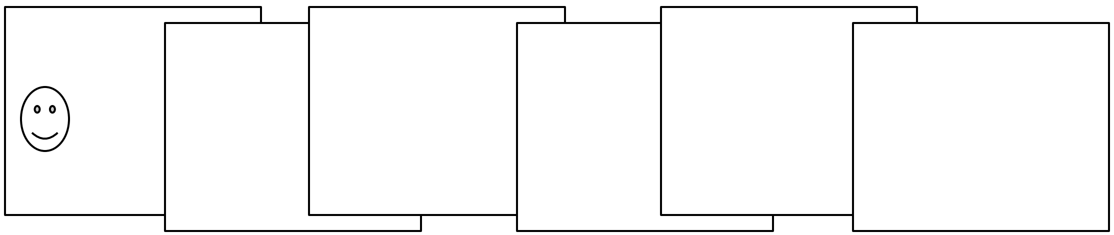
- Stitch pairs together, blend, then crop

# Assembling the panorama



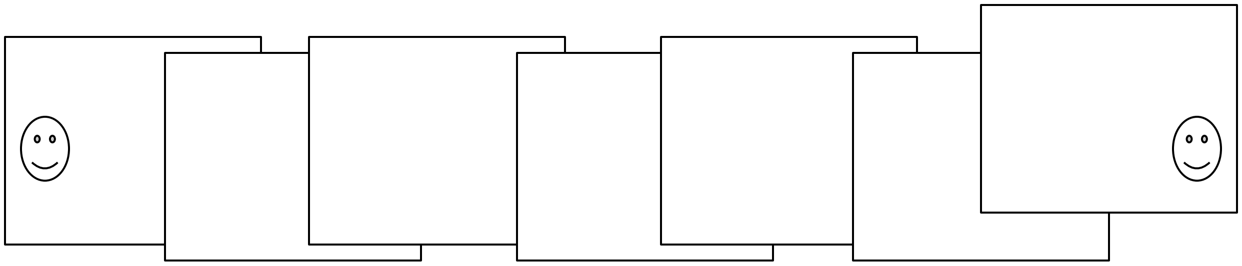
- Stitch pairs together, blend, then crop

# Assembling the panorama



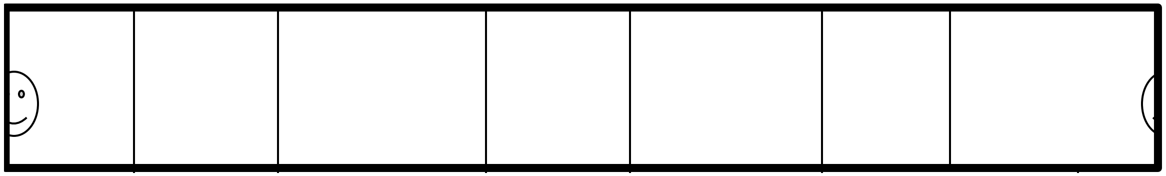
- Stitch pairs together, blend, then crop

# Assembling the panorama



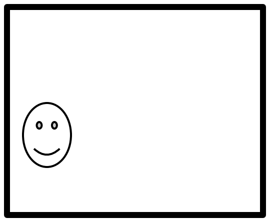
- Stitch pairs together, blend, then crop

# Assembling the panorama



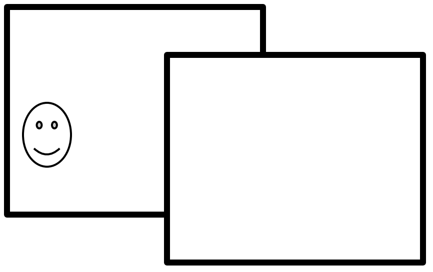
- Stitch pairs together, blend, then crop

# Problem: Drift



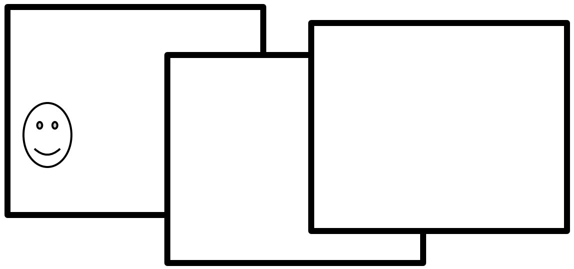
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



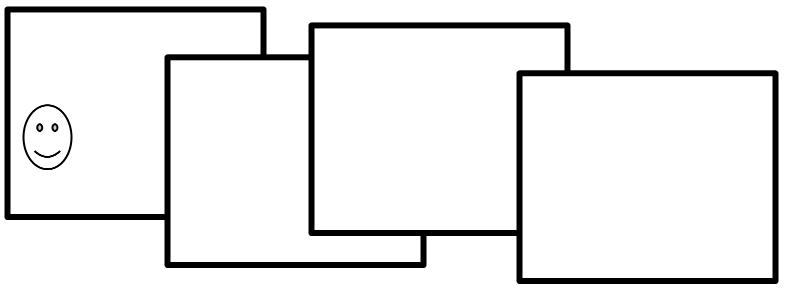
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



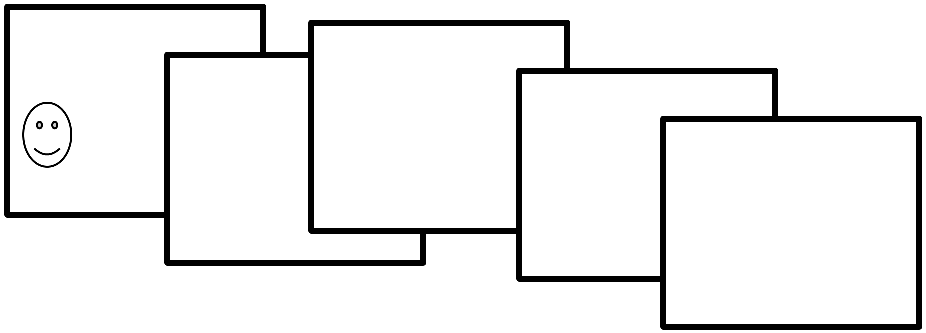
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



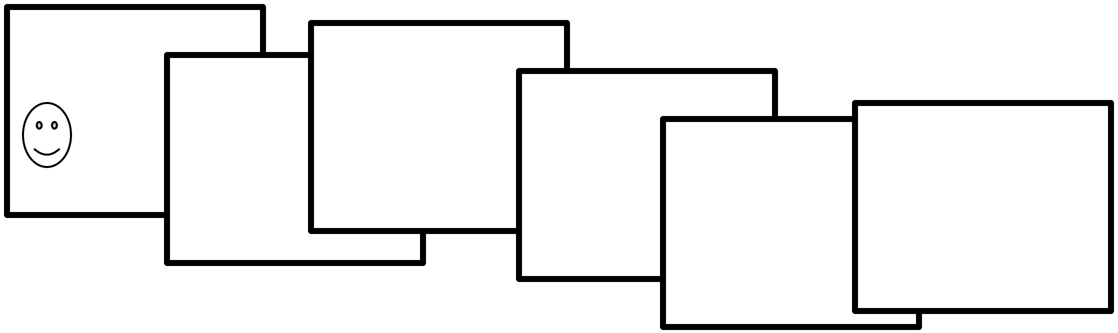
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



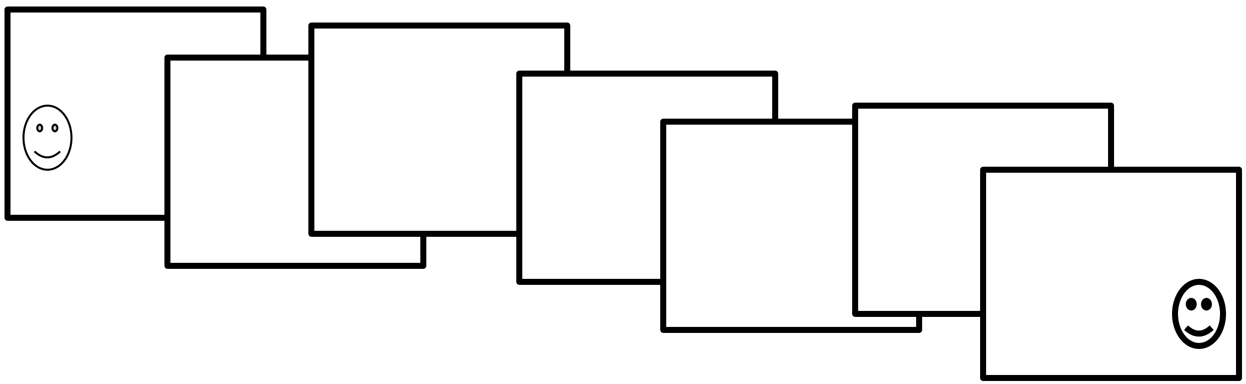
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



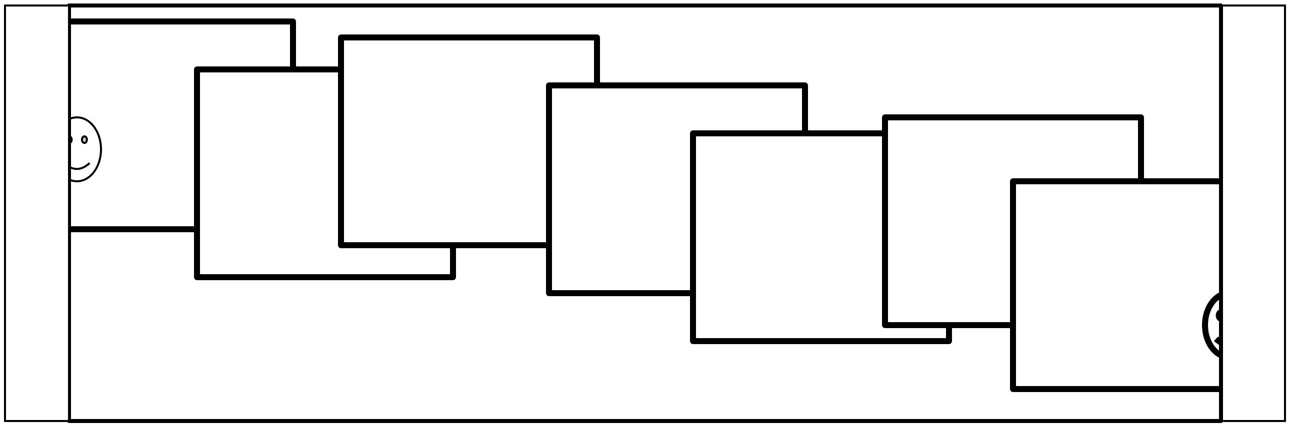
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



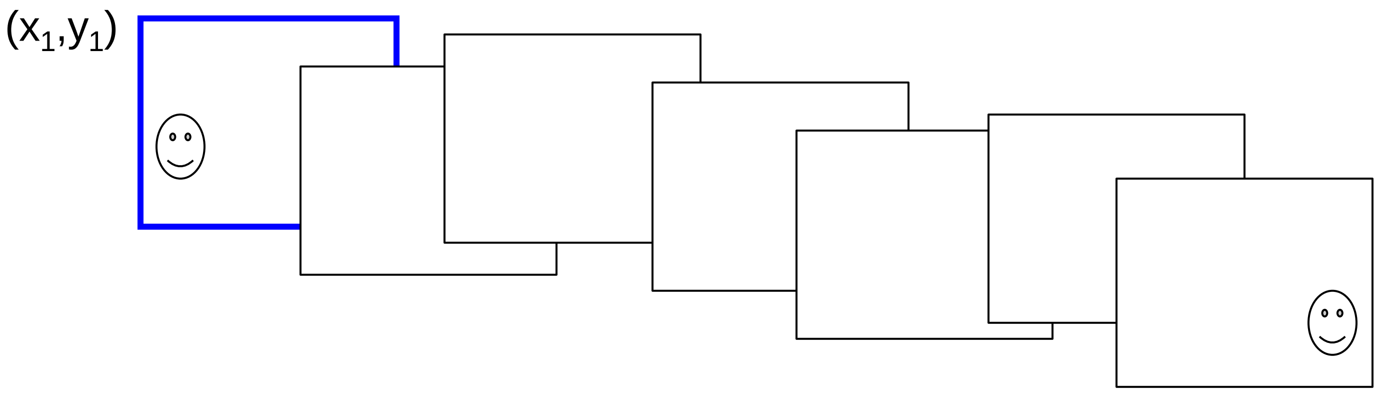
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



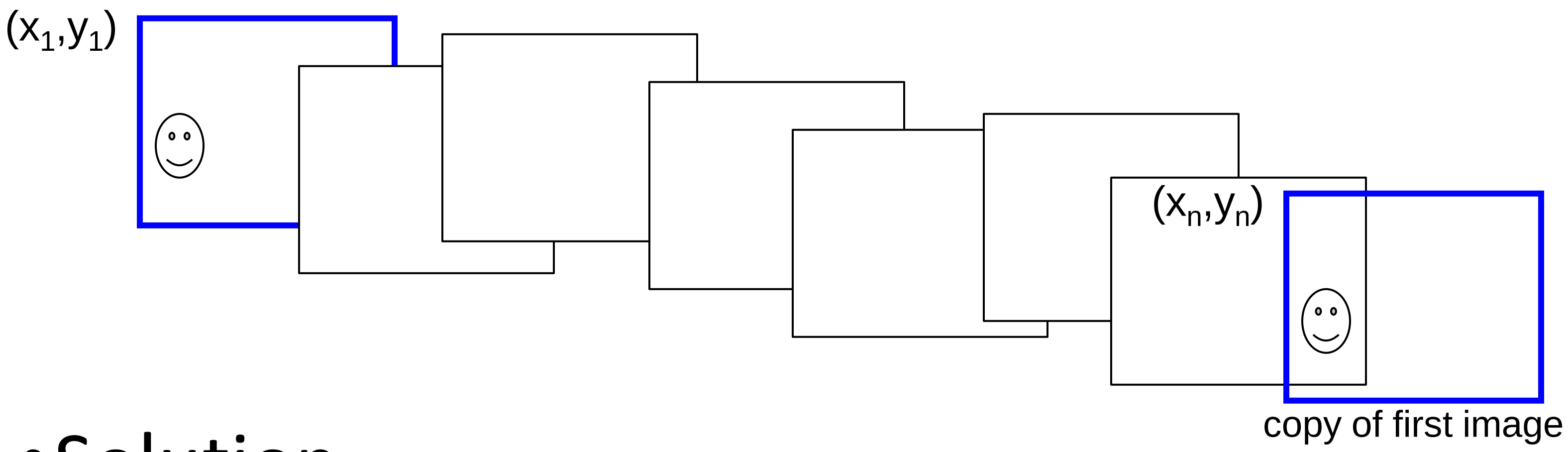
- Error accumulation
  - small errors accumulate over time

# Problem: Drift



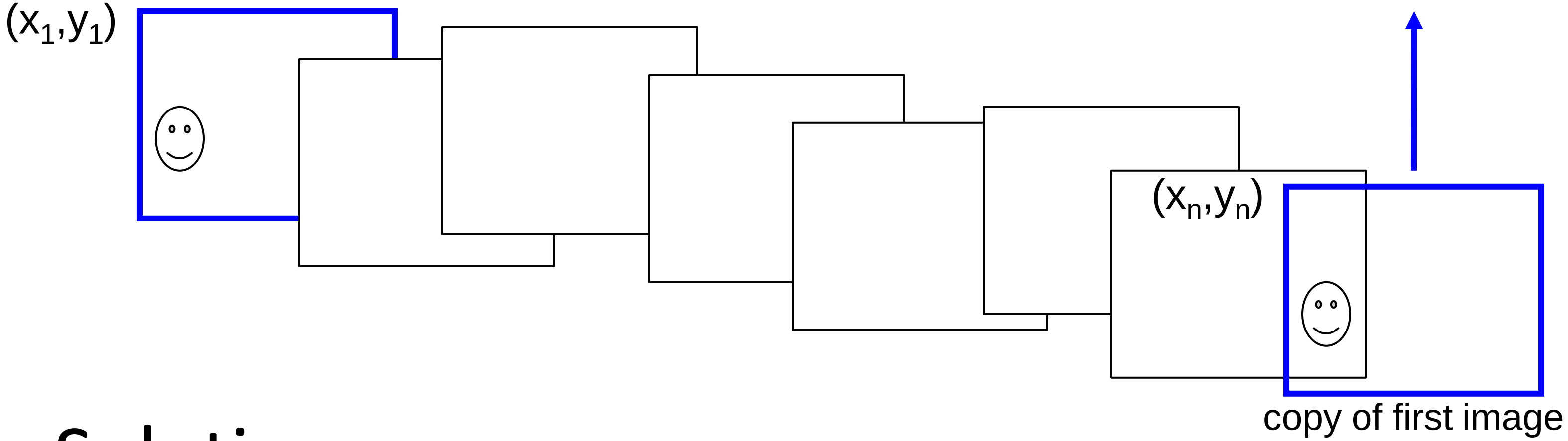

- Solution ?

# Problem: Drift



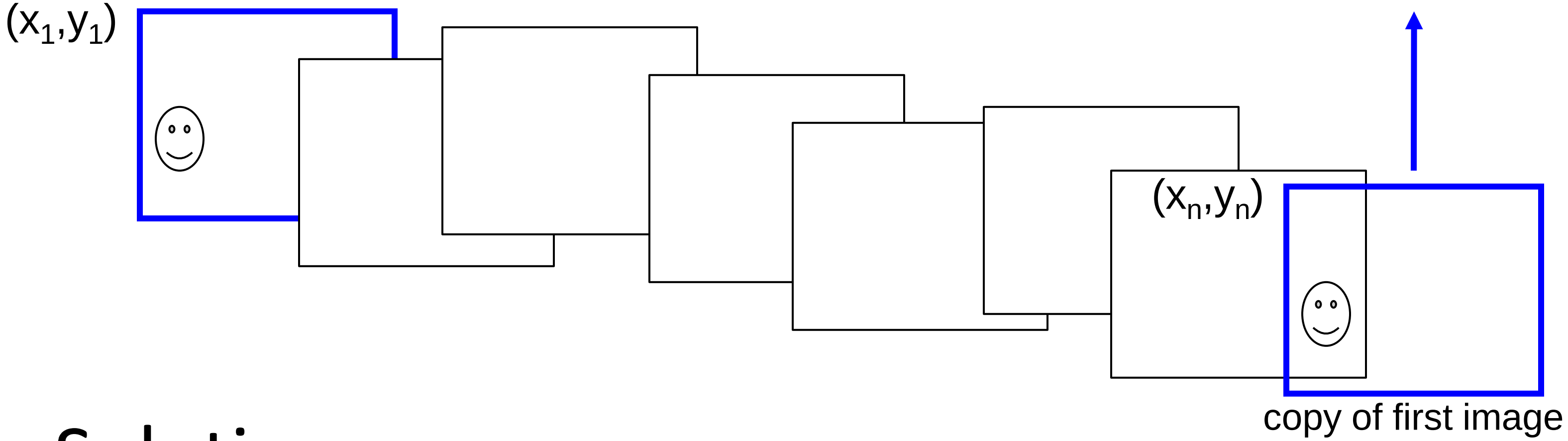

- Solution
  - add another copy of first image at the end

# Problem: Drift



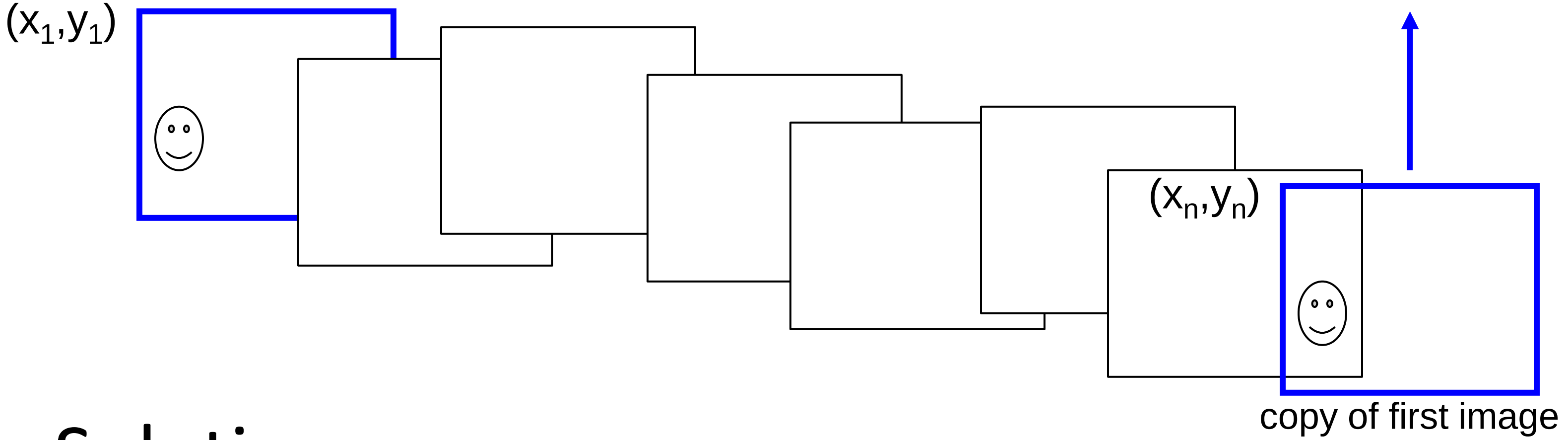

- Solution
  - add another copy of first image at the end
  - this gives a constraint: $y_n = y_1$

# Problem: Drift




- Solution
  - add another copy of first image at the end
  - this gives a constraint: $y_n = y_1$
  - there are a bunch of ways to solve this problem
    - add displacement of $(y_1 - y_n)/(n - 1)$ to each image after first

# Problem: Drift




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

# Bundle Adjustment for stitching

- Non-linear minimization of re-projection error

$$\mathbf{R}_i = e^{[\boldsymbol{\theta}_i]_\times}, \;\; [\boldsymbol{\theta}_i]_\times = \begin{bmatrix} 0 & -\theta_{i3} & \theta_{i2} \\ \theta_{i3} & 0 & -\theta_{i1} \\ -\theta_{i2} & \theta_{i1} & 0 \end{bmatrix}$$

- $\hat{\mathbf{x}}' = \mathbf{H}\mathbf{x}$ where **H = K' R' R$^{-1}$ K$^{-1}$**

$$\mathbf{K}_i = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$error = \sum dist(\mathbf{x}', \hat{\mathbf{x}}')$$

# Bundle Adjustment for stitching

- Non-linear minimization of re-projection error

$$\mathbf{R}_i = e^{[\boldsymbol{\theta}_i]_\times}, \quad [\boldsymbol{\theta}_i]_\times = \begin{bmatrix} 0 & -\theta_{i3} & \theta_{i2} \\ \theta_{i3} & 0 & -\theta_{i1} \\ -\theta_{i2} & \theta_{i1} & 0 \end{bmatrix}$$

- $\hat{\mathbf{x}}' = \mathbf{H}\mathbf{x}$ where **H = K' R' R$^{-1}$ K$^{-1}$**

$$\mathbf{K}_i = \begin{bmatrix} f_i & 0 & 0 \\ 0 & f_i & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$error = \sum dist(\mathbf{x}', \hat{\mathbf{x}}')$$

- Solve non-linear least squares (Levenberg-Marquardt algorithm)
  - See paper for details

Matthew Brown and David G. Lowe, Automatic Panoramic Image Stitching using Invariant Features, IJCV 2007. http://matthewalunbrown.com/papers/ijcv2007.pdf

# Bundle Adjustment

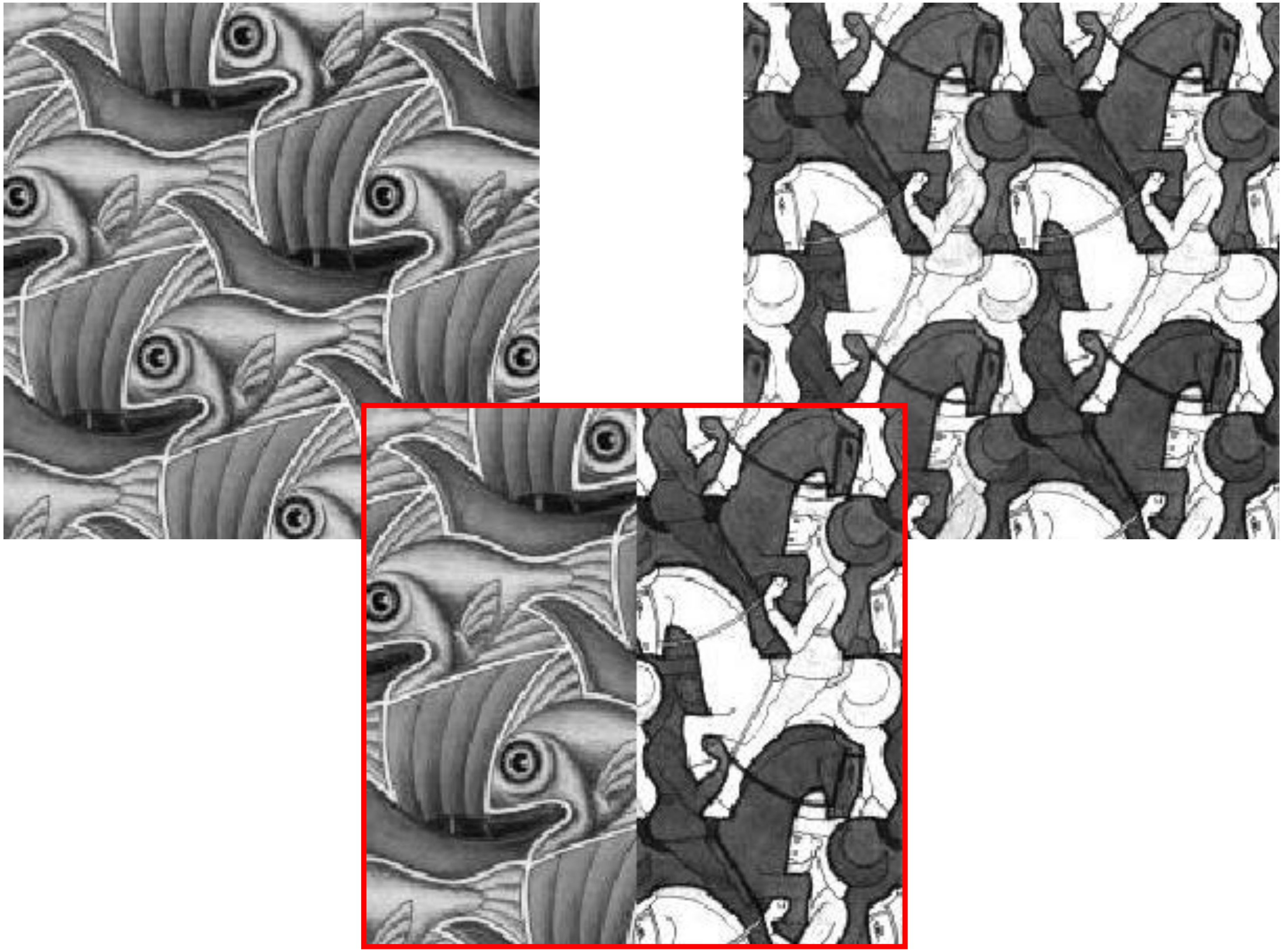
- New images initialised with rotation, focal length of best matching image

- We’ve aligned the images – now blending.

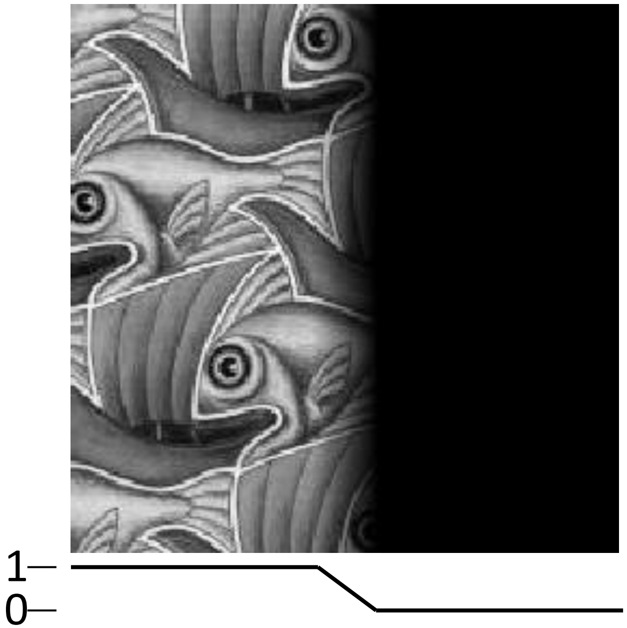
# Blending

- Want to seamlessly blend them together

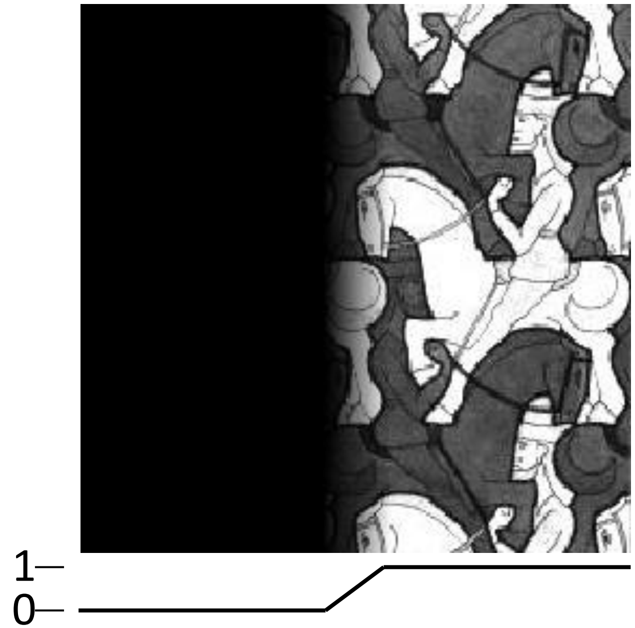
# Image Blending



# Feathering

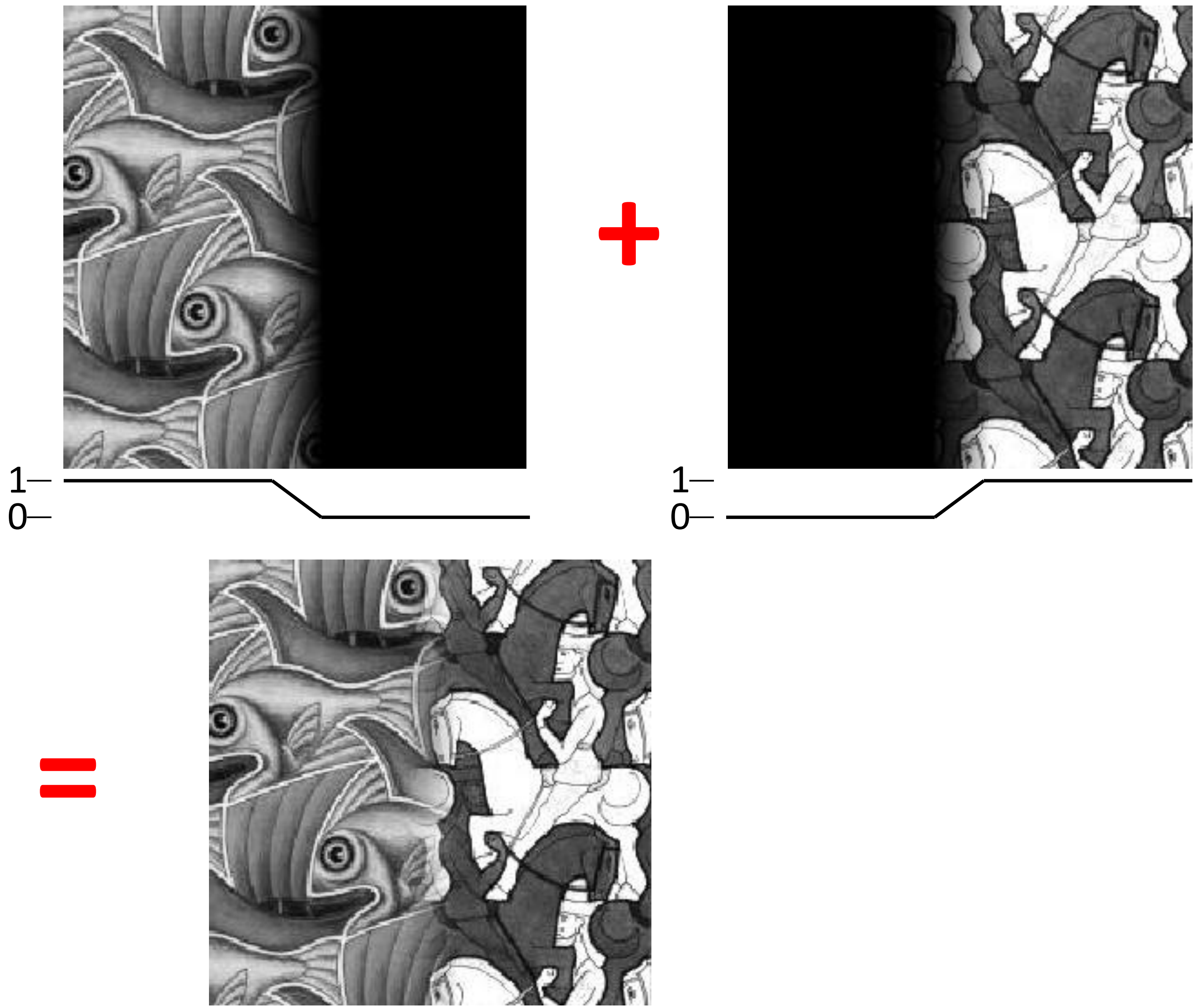



+

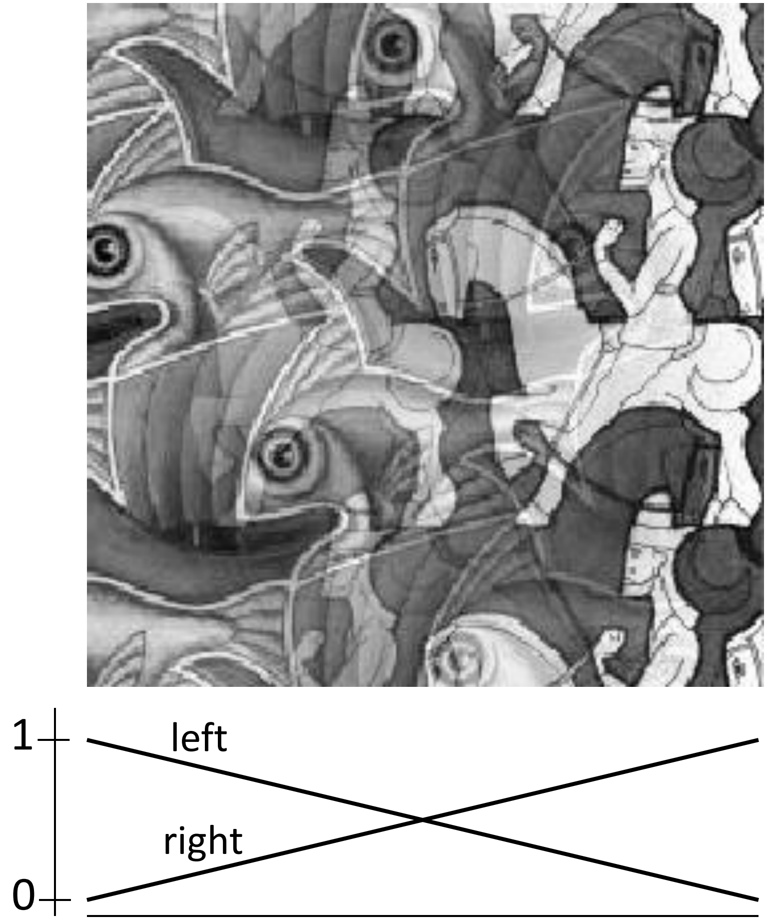


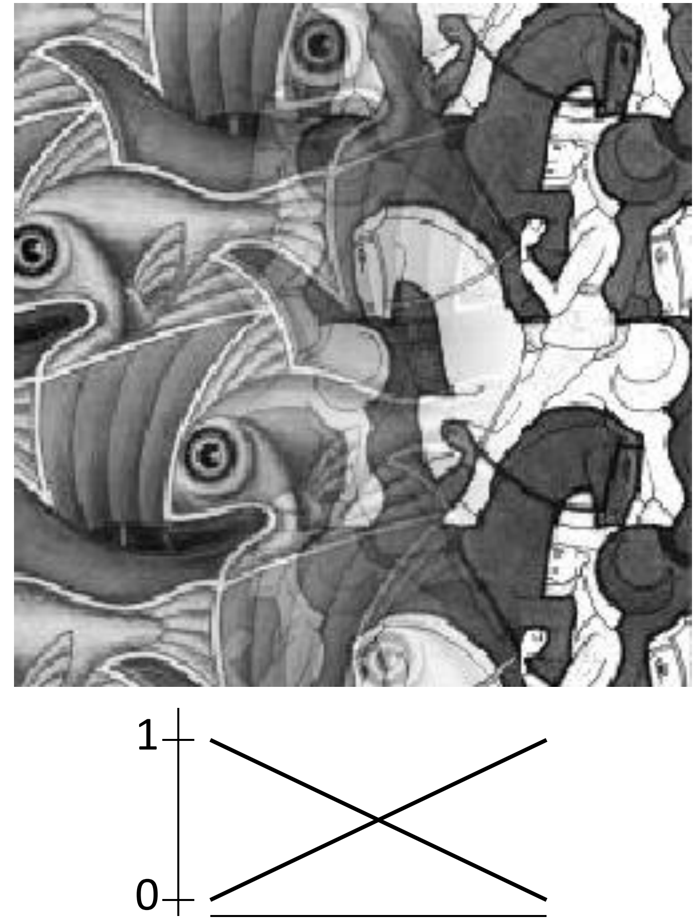

=

# Feathering



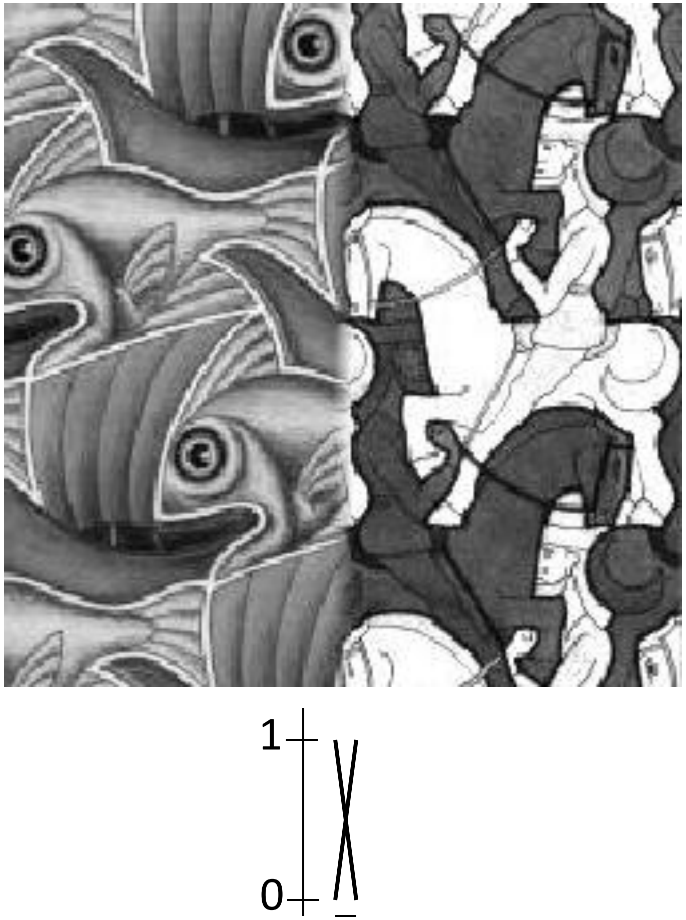
# Effect of window size

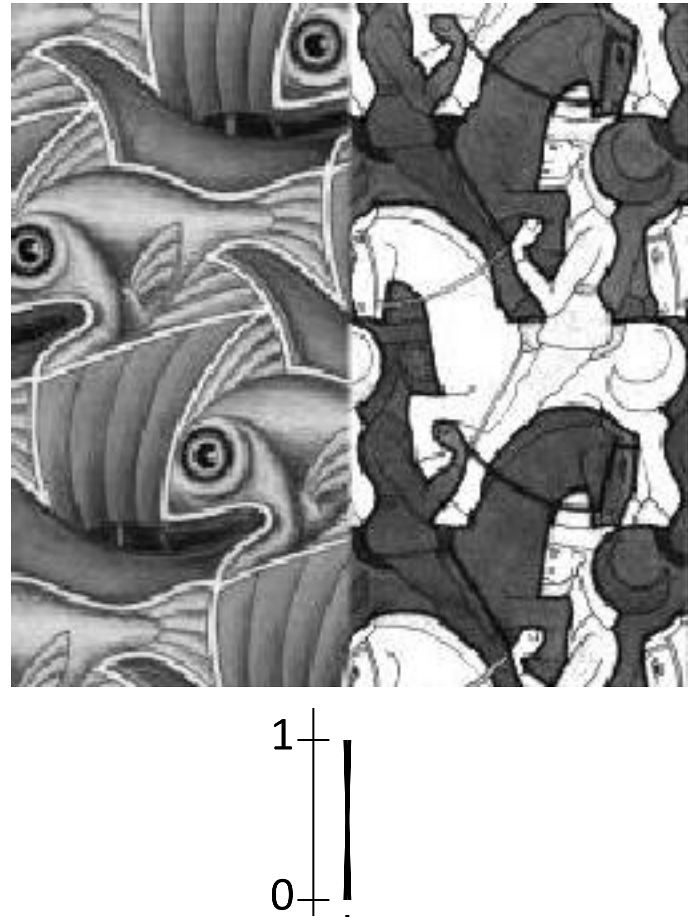


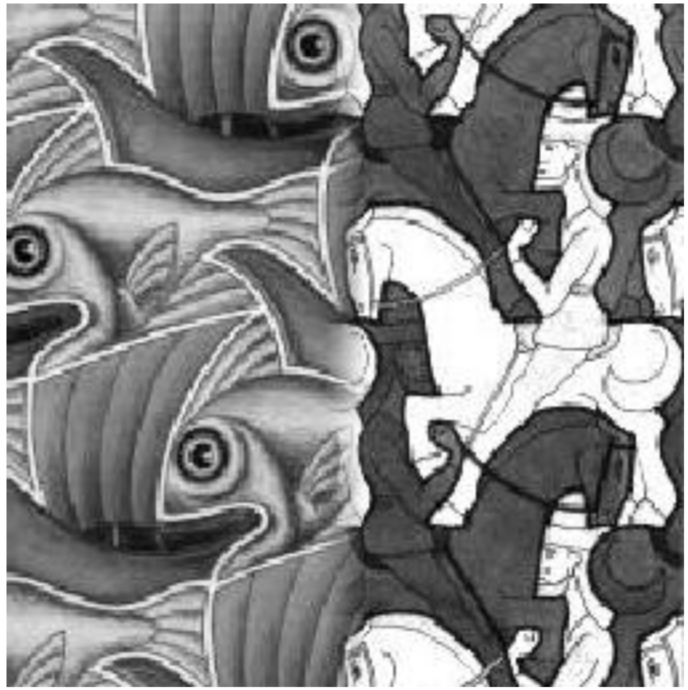

Ghosting Effect

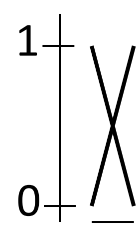
# Effect of window size



# Good window size






"Optimal" window: smooth but not ghosted

- Doesn't always work...

# Multi-band Blending

**The idea behind multi-band blending:**

Blend low frequencies over a large spatial range

&

Blend high frequencies over a short range

# Simplification: Two-band Blending

- Brown & Lowe, 2003
  - Only use two bands: high freq. and low freq.
  - Blends low freq. smoothly
  - Blend high freq. with no smoothing: use binary alpha

# Simplification: Two-band Blending

- Brown & Lowe, 2003
  - Only use two bands: high freq. and low freq.
  - Blends low freq. smoothly
  - Blend high freq. with no smoothing: use binary alpha

Blend the low frequency information using a linear weighted sum, and
select the high frequency information from the image with the maximum weight

# 2-band Blending

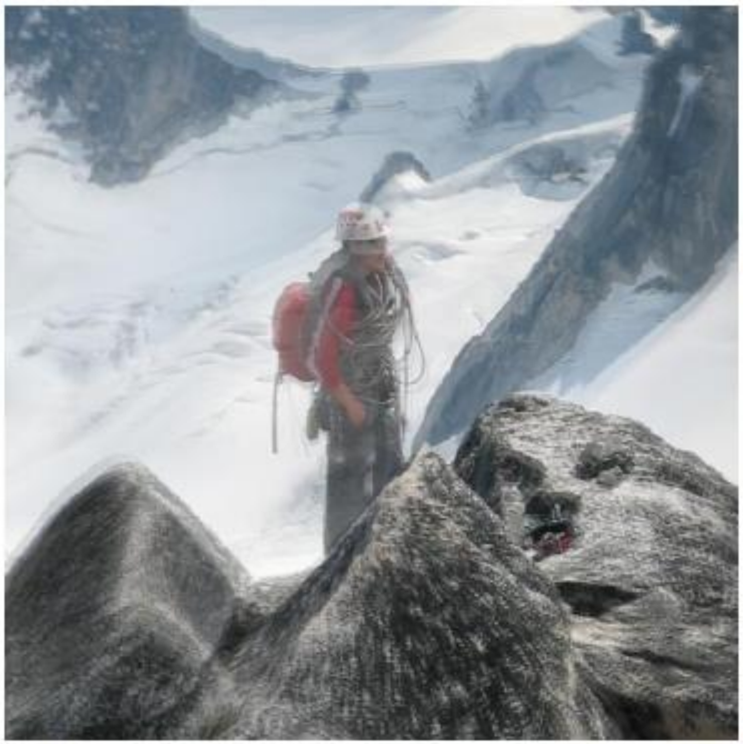
Low frequency

High frequency

# Linear Blending

# 2-band Blending

# Blending comparison (IJCV 2007)



(a) Linear blending

(b) Multi-band blending

# Things to remember

- Homography relates rotating cameras
- Recover homography using RANSAC and normalized DLT
- Bundle adjustment minimizes reprojection error for set of related images
- Details to make it look nice (e.g., blending)

# Acknowledgements


- Thanks to the following researchers for making their teaching/research material online
  - Forsyth
  - Steve Seitz
  - Noah Snavely
  - J.B. Huang
  - Derek Hoiem
  - J. Hays
  - J. Johnson
  - R. Girshick
  - S. Lazebnik
  - K. Grauman
  - Antonio Torralba
  - Rob Fergus
  - Leibe
  - And many more ……….

# Next Class

Epipolar Geometry and Stereo Vision