

# Image Stitching



Computational Photography  
Derek Hoiem, University of Illinois

# Project 3 Results

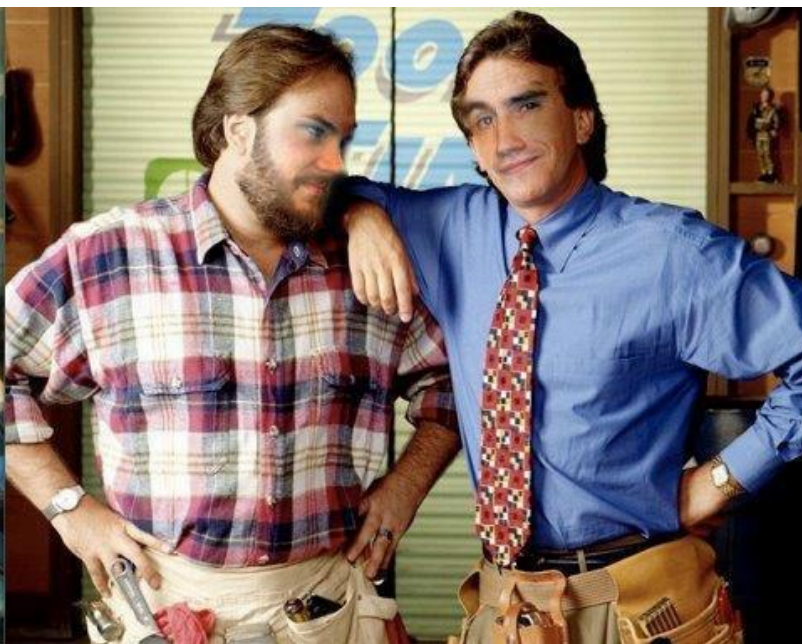
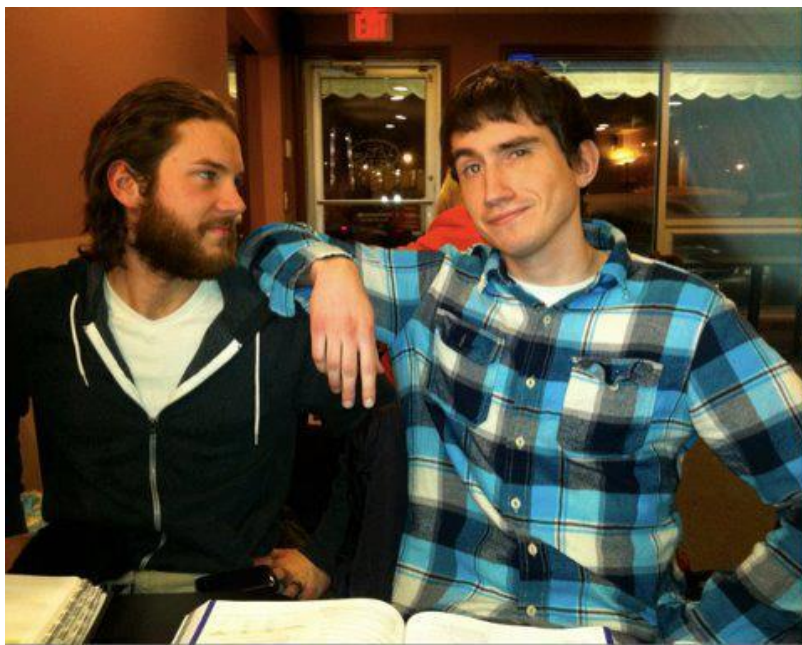


Donald Cha

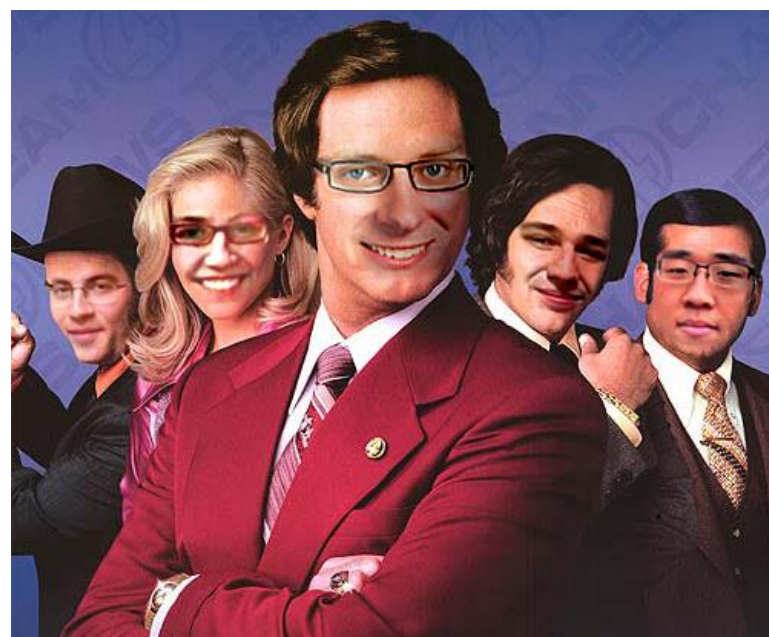


Hao Goa: which one is fake?









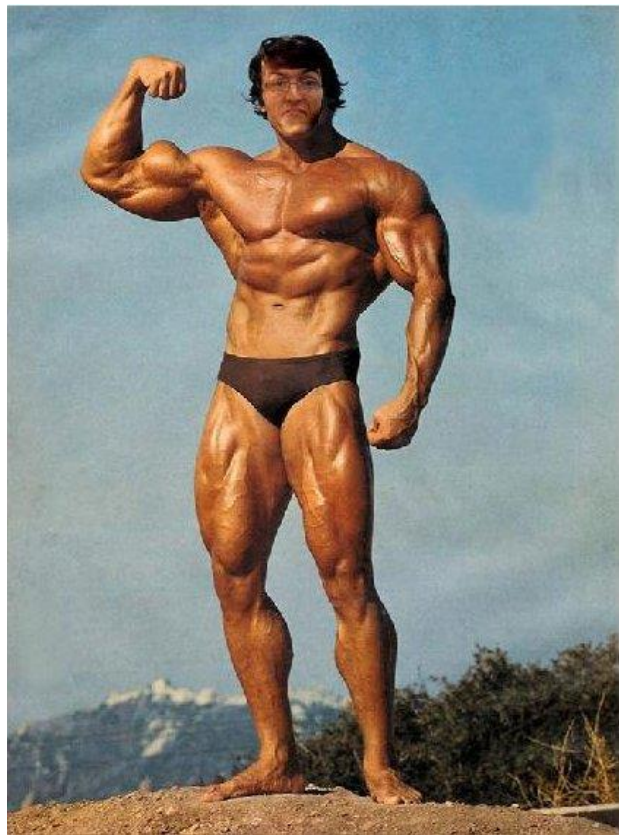
Daeyun Shin



Jane Wang



Yizhi Zhu



Sam Ricker



Joonyoung Seo





Rachit Nayudu



Joonyoung Seo



Jeremy Goodsitt



Jeremy Goodsitt



Zhang, Ruichuan



Elizabeth Weeks



Emmanuel Arregoitia-Diaz



Aditya Deshpande

# Highlighted projects

- Eric Huber: <http://web.engr.illinois.edu/~echuber2/cs498dwh/proj3/>
  - Description of several iterations of attempts with various modifications
- Sam Ricker: <http://web.engr.illinois.edu/~sricker2/cs498dwh/proj3/>
  - Fun results
- Daeyun Shin: <http://web.engr.illinois.edu/~dshin11/cs498dwh/proj3/>
  - Nice results + extras (maybe)
- Elizabeth Weeks: <http://web.engr.illinois.edu/~eweeks2/cs498dwh/proj3/>
  - Lots of results, non-photorealistic rendering
- Jeremy Goodsitt: <http://web.engr.illinois.edu/~goodsit2/cs498dwh/proj3/>
  - Blending comparison, NPR



# Project 5

Input video:

[https://www.youtube.com/watch?v=agl5za\\_gHHU](https://www.youtube.com/watch?v=agl5za_gHHU)

Aligned frames:

<https://www.youtube.com/watch?v=Uahy6kPotaE>

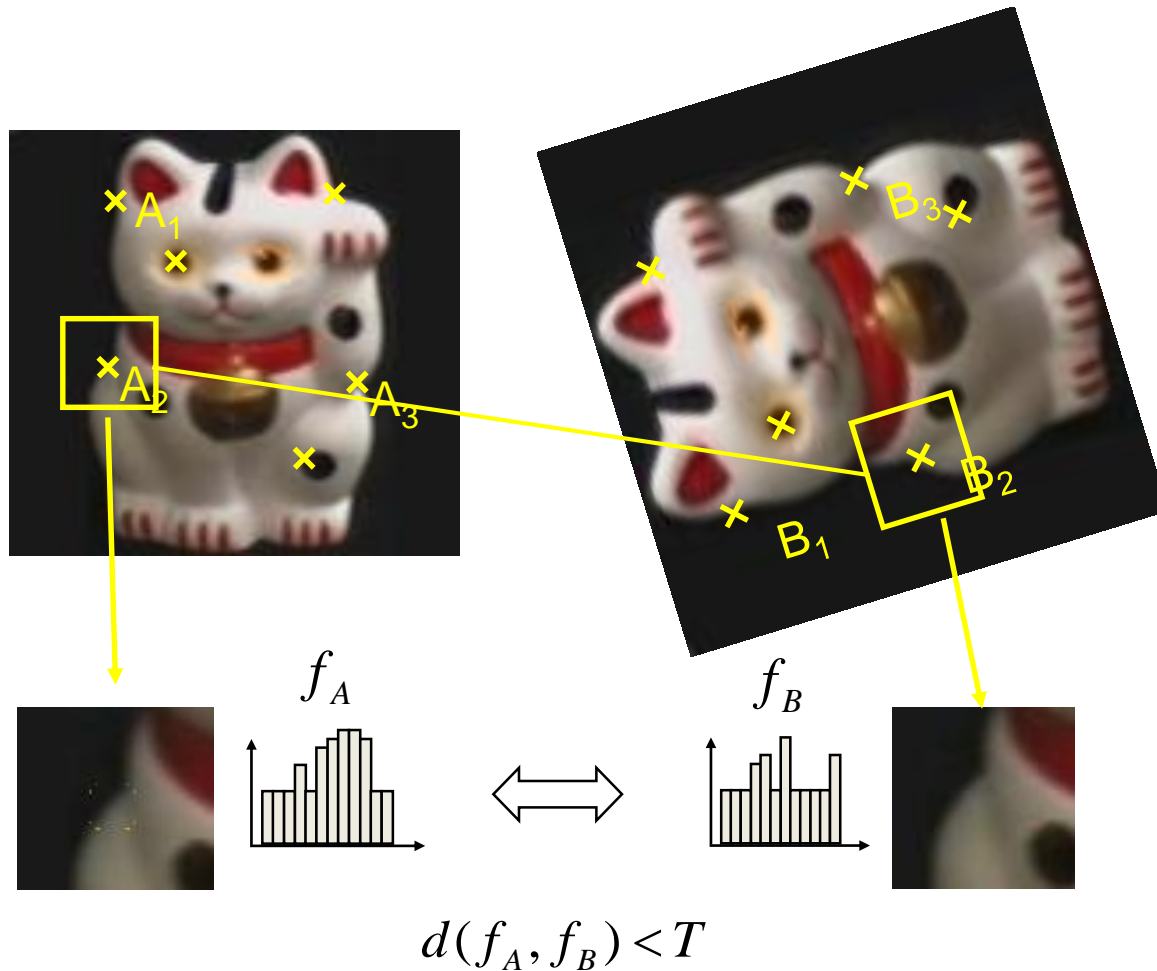
Background:

<https://www.youtube.com/watch?v=Vt9vv1zCnLA>

Foreground:

<https://www.youtube.com/watch?v=OICkKNndEt4>

# Last Class: Keypoint Matching

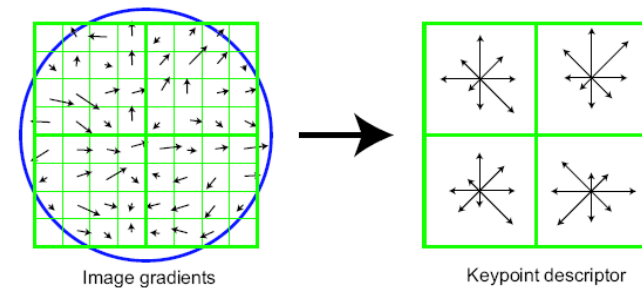
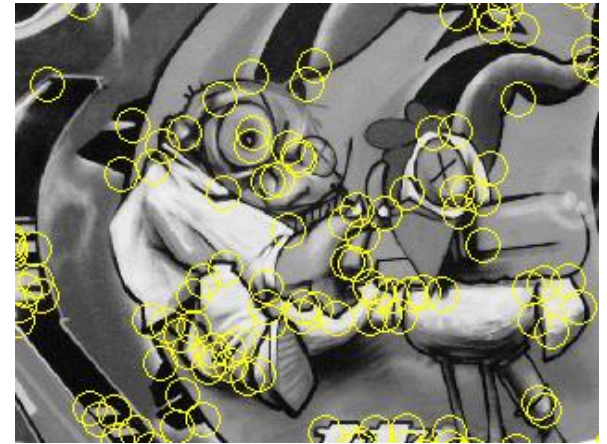


1. Find a set of distinctive key-points
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors



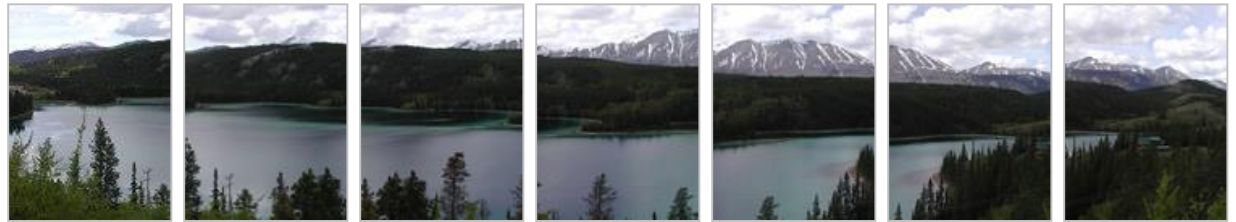
# Last Class: Summary

- Keypoint detection: repeatable and distinctive
  - Corners, blobs
  - Harris, DoG
- Descriptors: robust and selective
  - SIFT: spatial histograms of gradient orientation



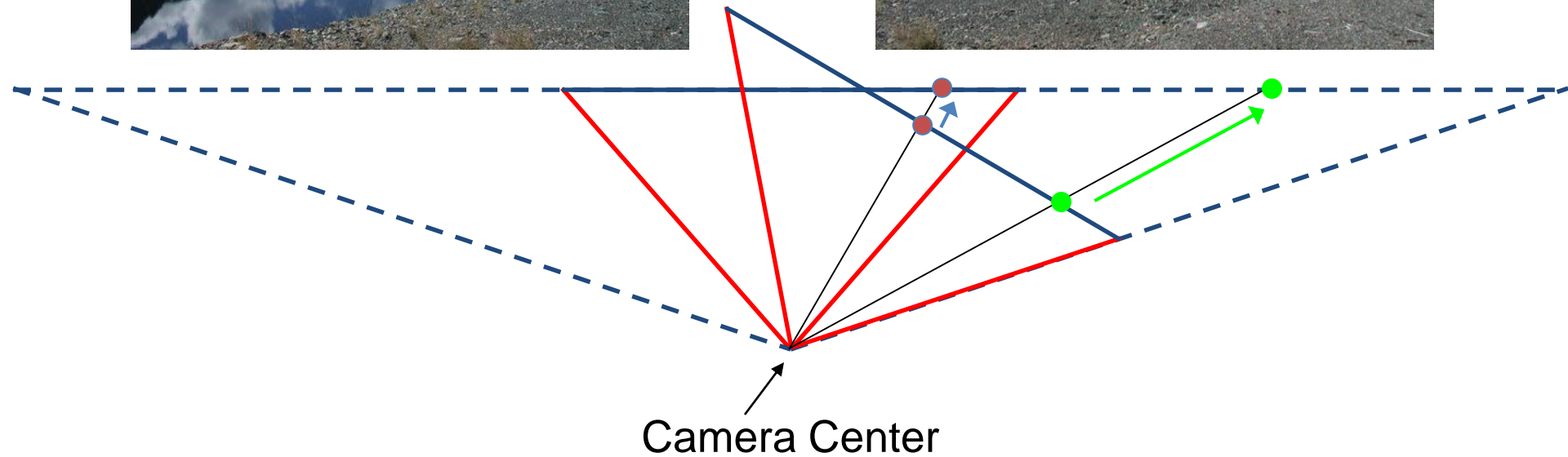
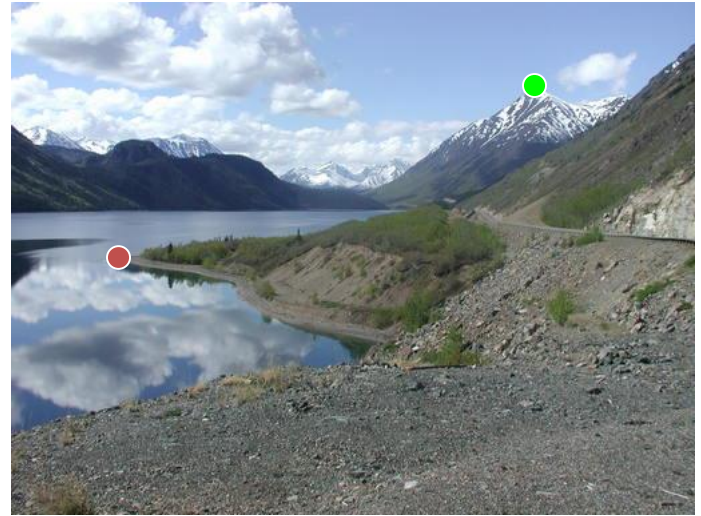
# Today: Image Stitching

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





# Views from rotating camera



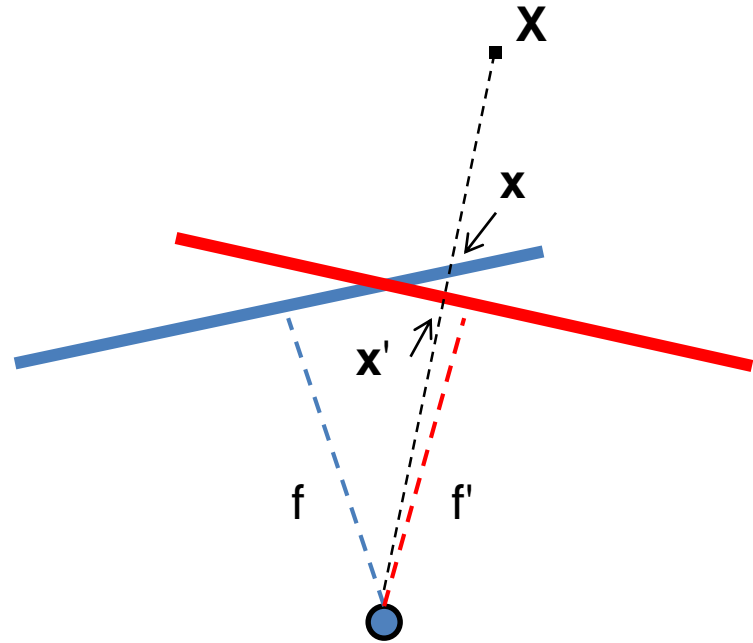
# Problem basics

- Do on board



# Basic problem

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



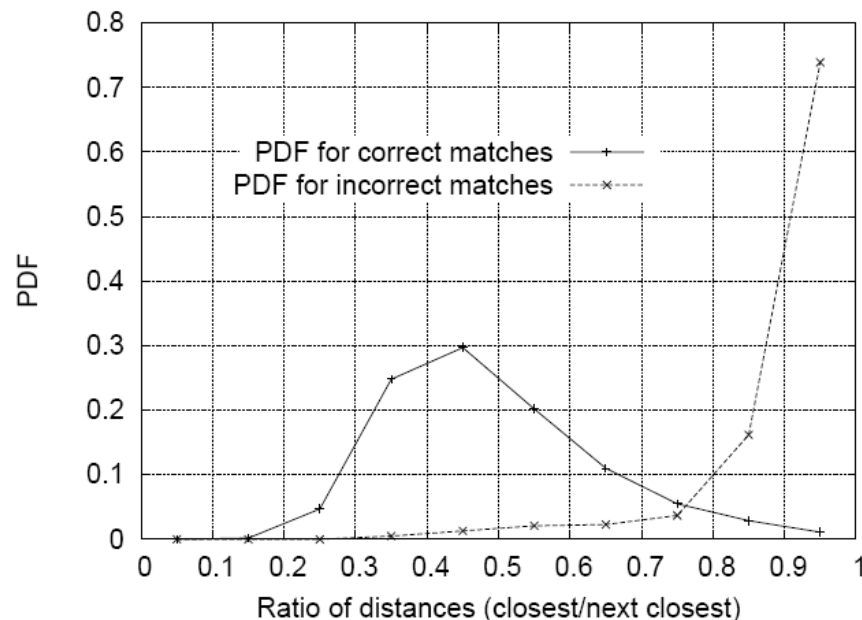
- $x' = Hx$  where  $H = K' R' R^{-1} K^{-1}$
- Typically only  $R$  and  $f$  will change (4 parameters), but, in general,  $H$  has 8 parameters

# Image Stitching Algorithm Overview

1. Detect keypoints
2. Match keypoints
3. Estimate homography with four matched keypoints (using RANSAC)
4. Project onto a surface and blend

# Image Stitching Algorithm Overview

1. Detect/extract keypoints (e.g., DoG/SIFT)
2. Match keypoints (most similar features, compared to 2<sup>nd</sup> most similar)





# Computing homography

Assume we have four matched points: How do we compute homography  $\mathbf{H}$ ?

Direct Linear Transformation (DLT)

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

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

# Computing homography

## Direct Linear Transform

$$\begin{bmatrix} -u_1 & -v_1 & -1 & 0 & 0 & 0 & u_1 u'_1 & v_1 u'_1 & u'_1 \\ 0 & 0 & 0 & -u_1 & -v_1 & -1 & u_1 v'_1 & v_1 v'_1 & v'_1 \\ & & & \vdots & & & & & \\ 0 & 0 & 0 & -u_n & -v_n & -1 & u_n v'_n & v_n v'_n & v'_n \end{bmatrix} \mathbf{h} = \mathbf{0} \Rightarrow \mathbf{A} \mathbf{h} = \mathbf{0}$$

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

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

Matlab

```
[U, S, V] = svd(A);  
h = V(:, end);
```

# Computing homography

Assume we have four matched points: How do we compute homography  $\mathbf{H}$ ?

## Normalized DLT

### 1. Normalize coordinates for each image

- a) Translate for zero mean
- b) Scale so that  $u$  and  $v$  are  $\sim 1$  on average

$$\tilde{\mathbf{x}} = \mathbf{T}\mathbf{x} \quad \tilde{\mathbf{x}}' = \mathbf{T}'\mathbf{x}'$$

- This makes problem better behaved numerically (see Hartley and Zisserman p. 107-108)

### 2. Compute $\tilde{\mathbf{H}}$ using DLT in normalized coordinates

### 3. Unnormalize: $\mathbf{H} = \mathbf{T}'^{-1}\tilde{\mathbf{H}}\mathbf{T}$

$$\mathbf{x}'_i = \mathbf{H}\mathbf{x}_i$$



# Computing homography

- Assume we have matched points with outliers:  
How do we compute homography **H**?

Automatic Homography Estimation with RANSAC

# RANSAC: RANdom SAmple Consensus

Scenario: We've got way more matched points than needed to fit the parameters, but we're not sure which are correct


## RANSAC Algorithm

- Repeat N times
  1. Randomly select a sample
    - Select just enough points to recover the parameters
  2. Fit the model with random sample
  3. See how many other points agree
- Best estimate is one with most agreement
  - can use agreeing points to refine estimate

# Computing homography

- Assume we have matched points with outliers: How do we compute homography  $\mathbf{H}$ ?

## Automatic Homography Estimation with RANSAC

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

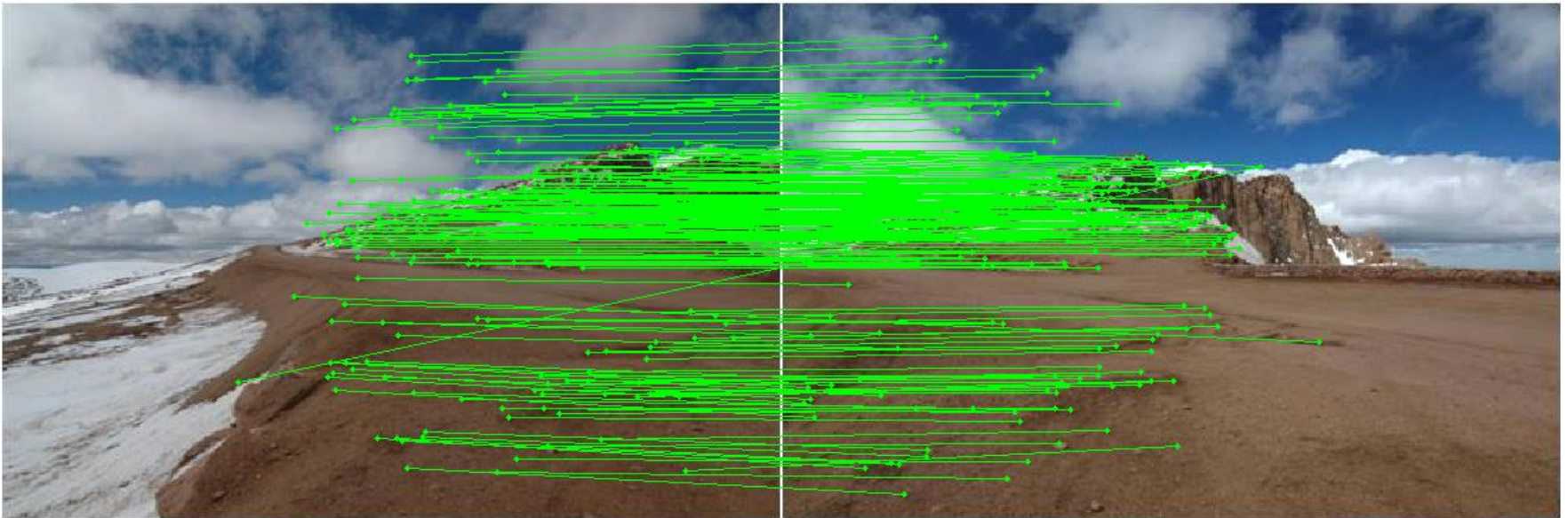


# Automatic Image Stitching

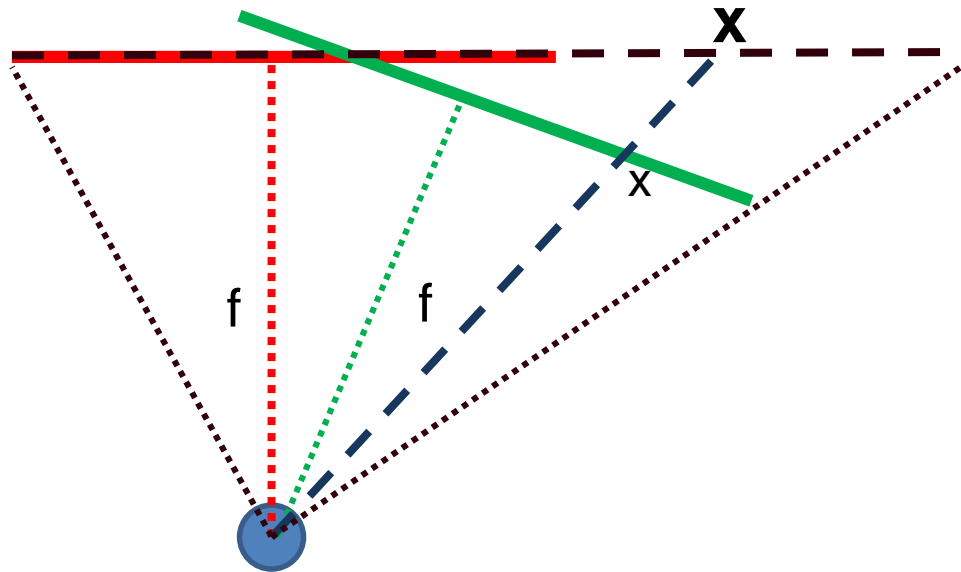
1. Compute interest points on each image
2. Find candidate matches
3. Estimate homography  $\mathbf{H}$  using matched points and RANSAC with normalized DLT
4. Project each image onto the same surface and blend

# Choosing a Projection Surface

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



# Planar Mapping



- 1) For red image: pixels are already on the planar surface
- 2) For green image: map to first image plane



# Planar vs. Cylindrical Projection

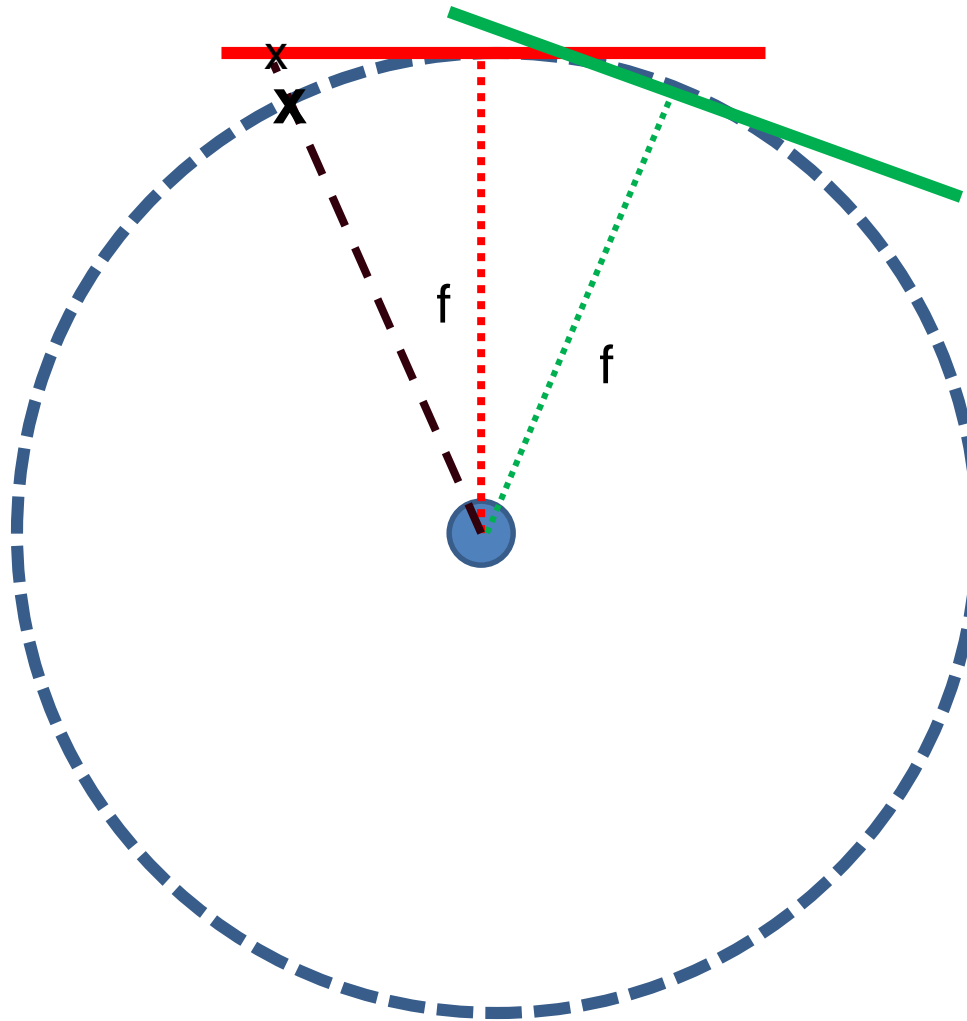


# Planar vs. Cylindrical Projection

Planar



# Cylindrical Mapping



- 1) For red image: compute  $h$ ,  $\theta$  on cylindrical surface from  $(u, v)$
- 2) For green image: map to first image plane, then map to cylindrical surface

# Planar vs. Cylindrical Projection

Cylindrical



# Planar vs. Cylindrical Projection

Cylindrical







**Planar**



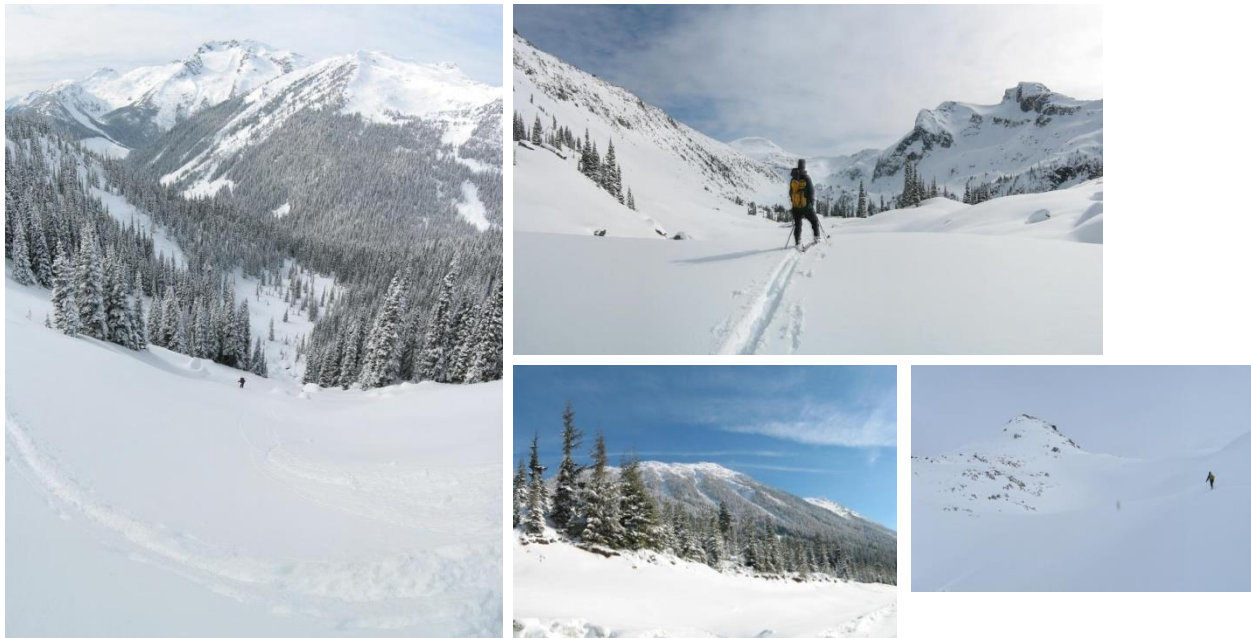
**Cylindrical**

# Simple gain adjustment



# Automatically choosing images to stitch

# Recognizing Panoramas





# Recognizing Panoramas

Input: N images

1. Extract SIFT points, descriptors from all images
2. Find K-nearest neighbors for each point ( $K=4$ )
3. For each image
  - a) Select M candidate matching images by counting matched keypoints ( $M=6$ )
  - b) Solve homography  $\mathbf{H}_{ij}$  for each matched image



# Recognizing Panoramas

Input: N images

1. Extract SIFT points, descriptors from all images
2. Find K-nearest neighbors for each point (K=4)
3. For each image
  - a) Select M candidate matching images by counting matched keypoints (M=6)
  - b) Solve homography  $\mathbf{H}_{ij}$  for each matched image
  - c) Decide if match is valid ( $n_i > 8 + 0.3 n_f$ )

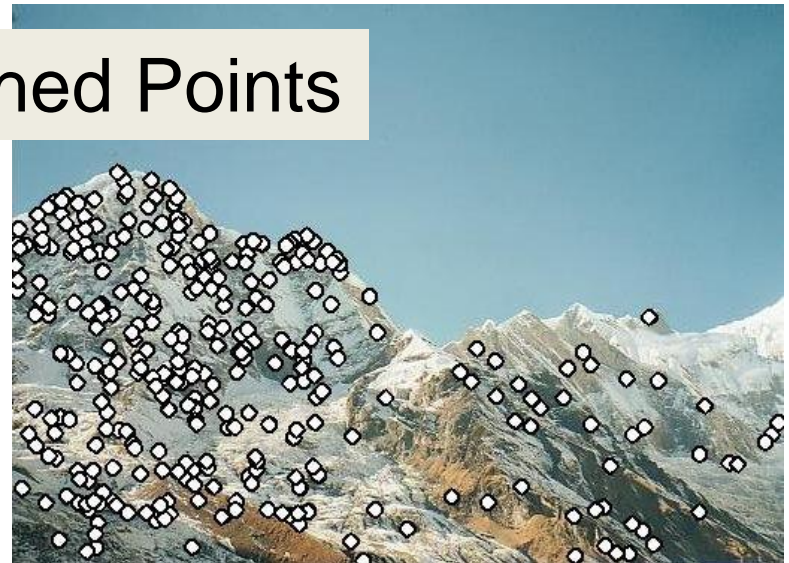
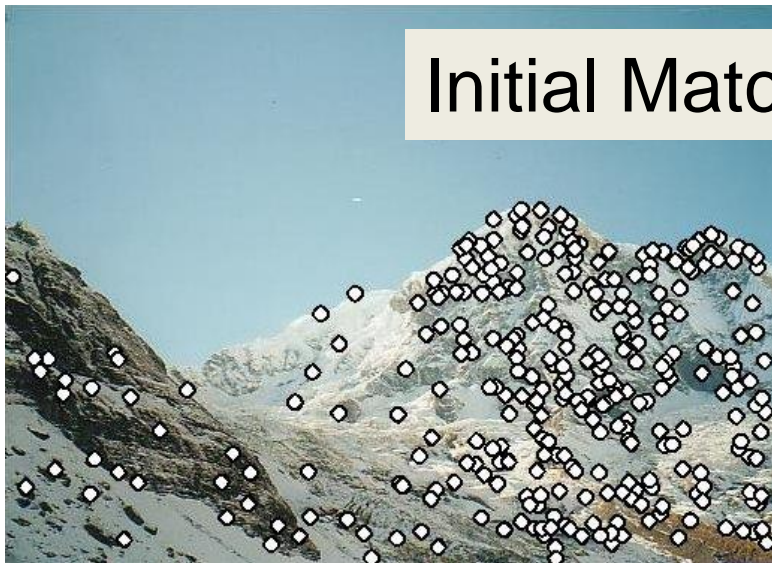
# inliers

# keypoints in  
overlapping area

# RANSAC for Homography



Initial Matched Points

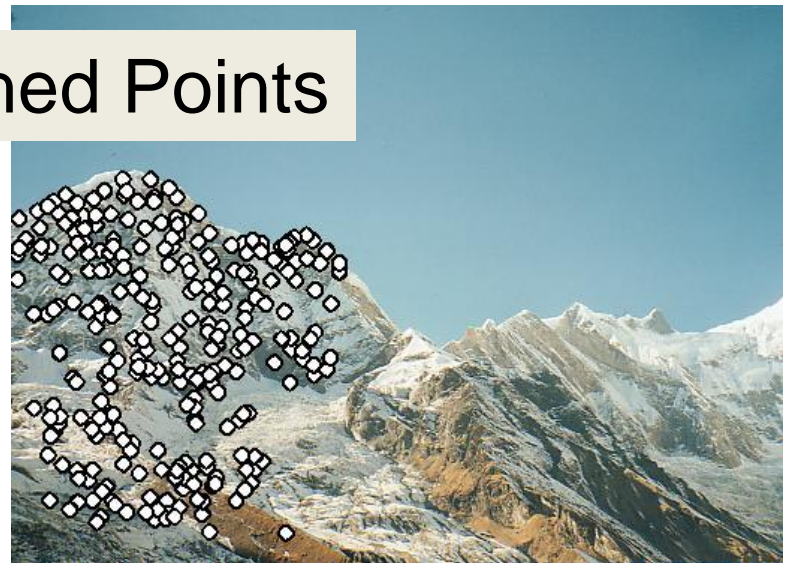
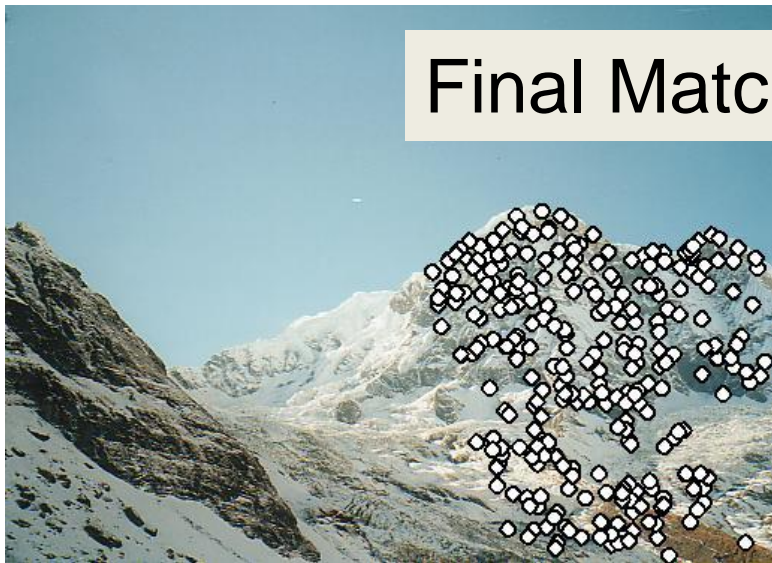




# RANSAC for Homography

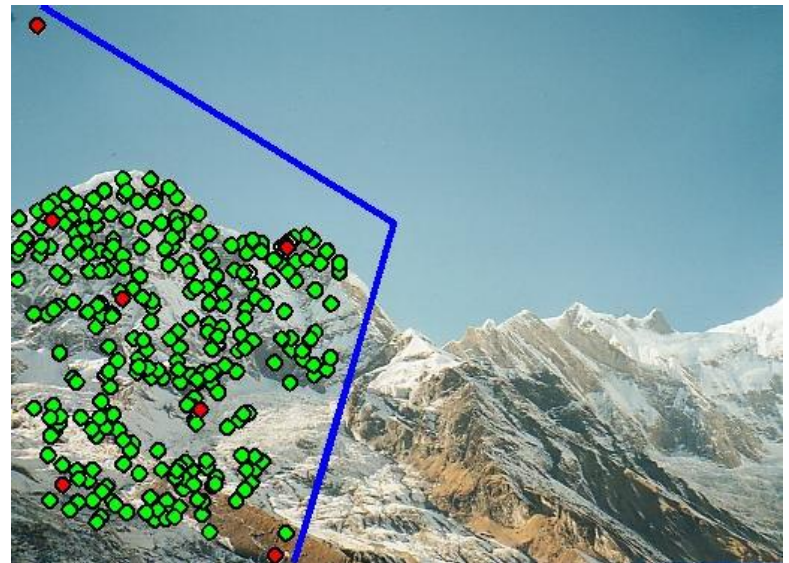
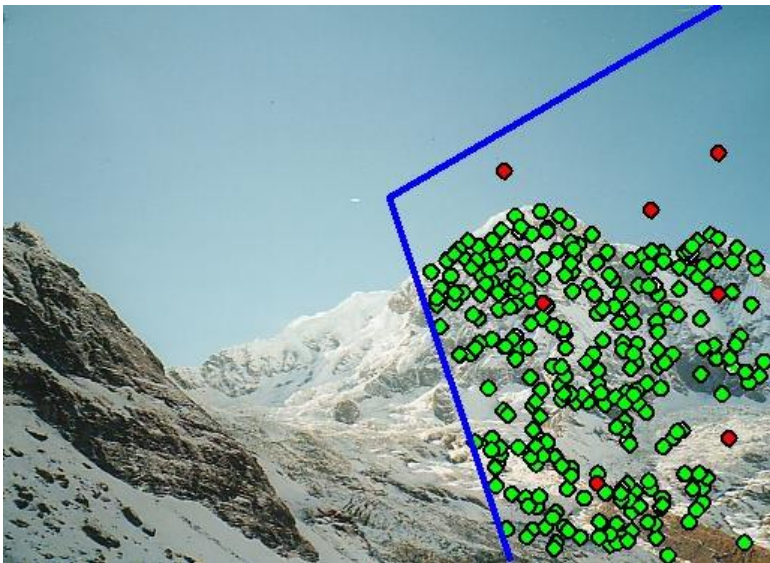


Final Matched Points

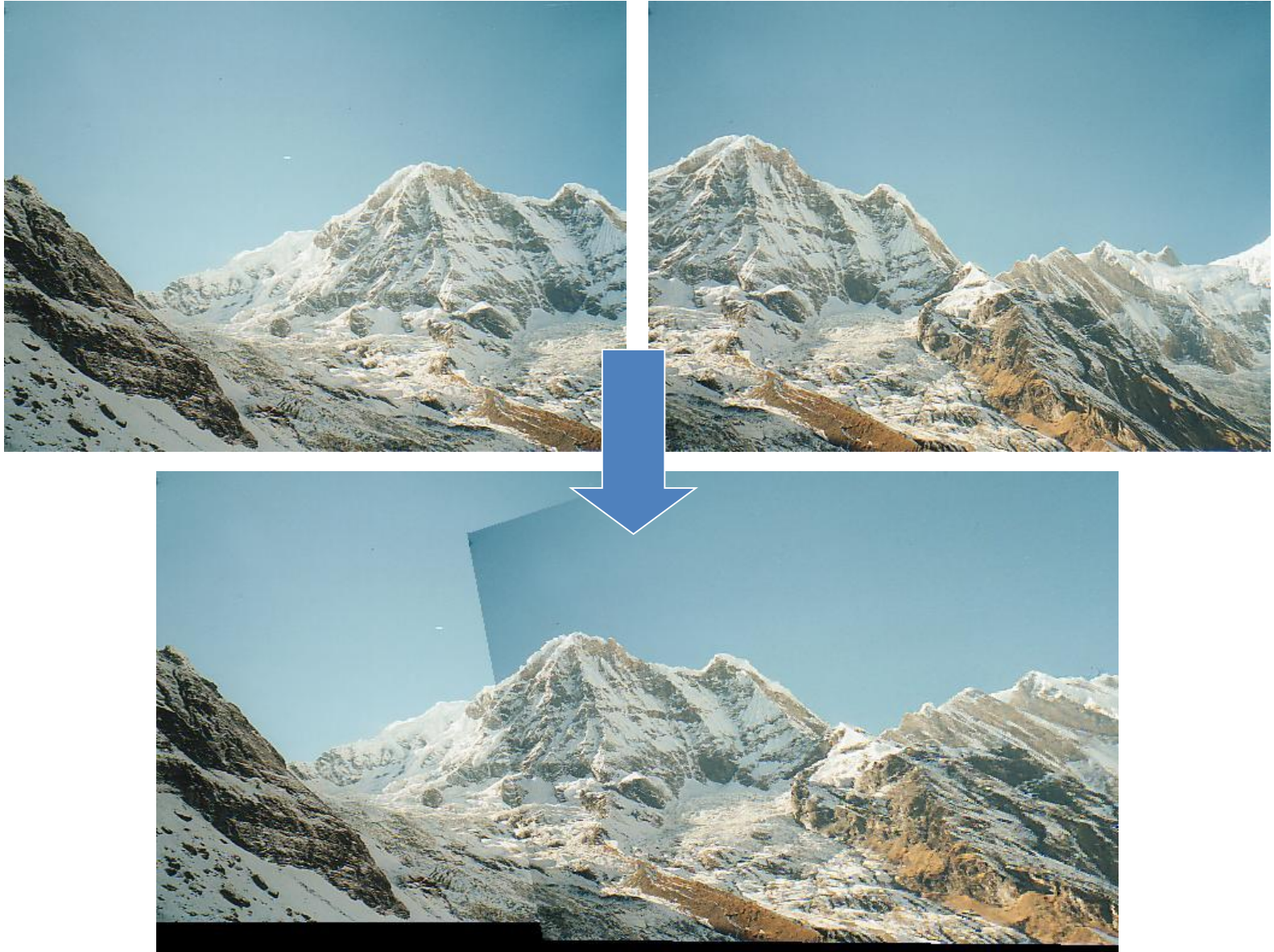




# Verification



# RANSAC for Homography



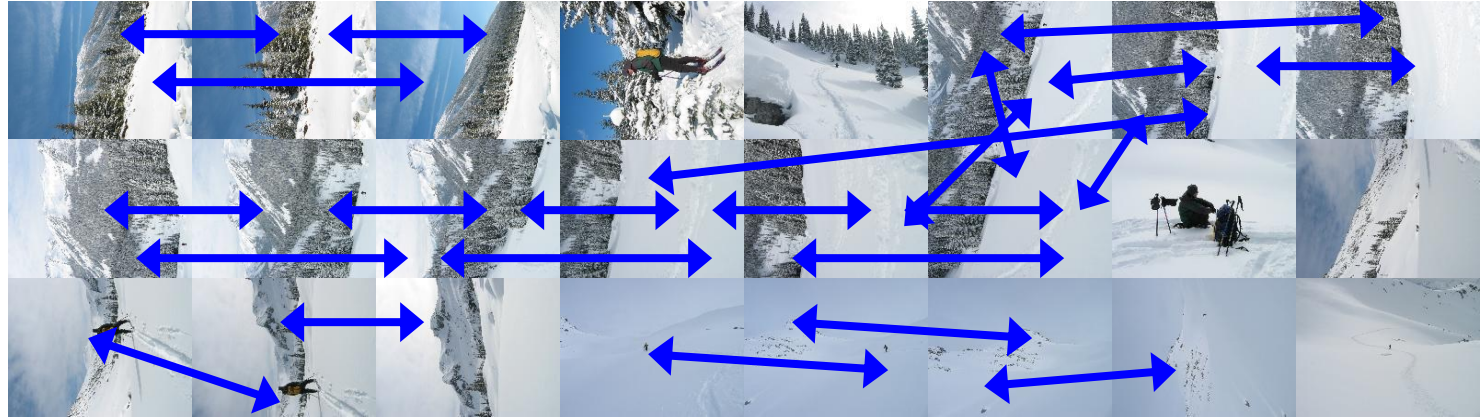


# Recognizing Panoramas (cont.)

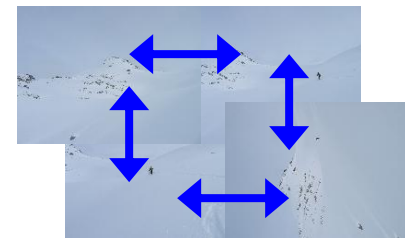
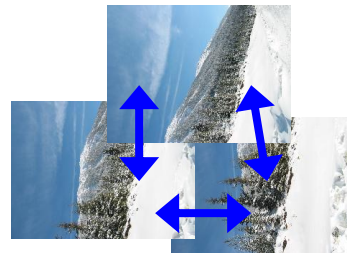
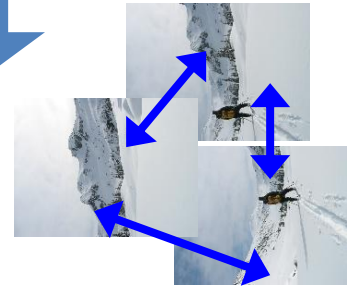
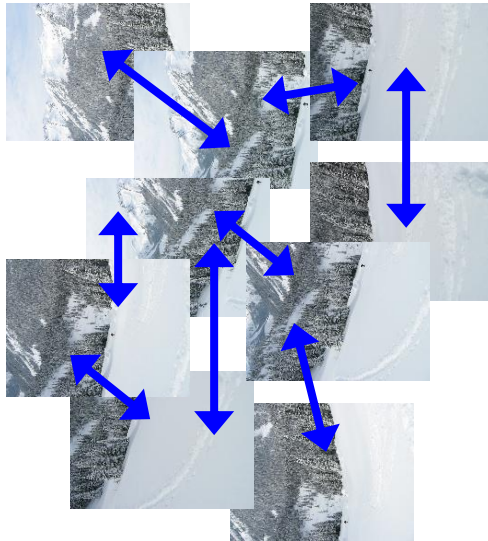
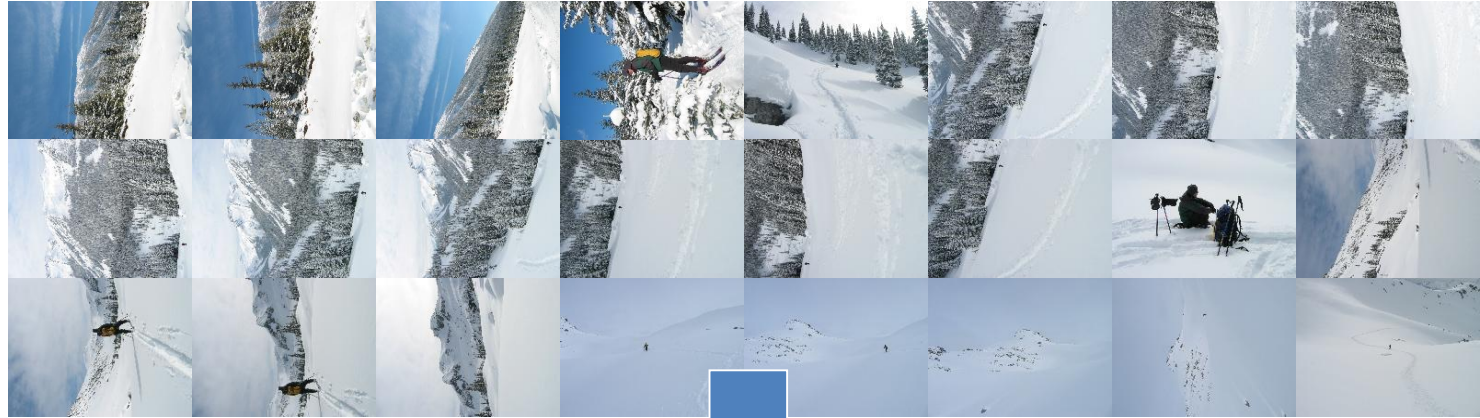
(now we have matched pairs of images)

4. Find connected components

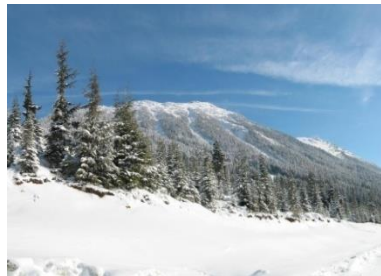
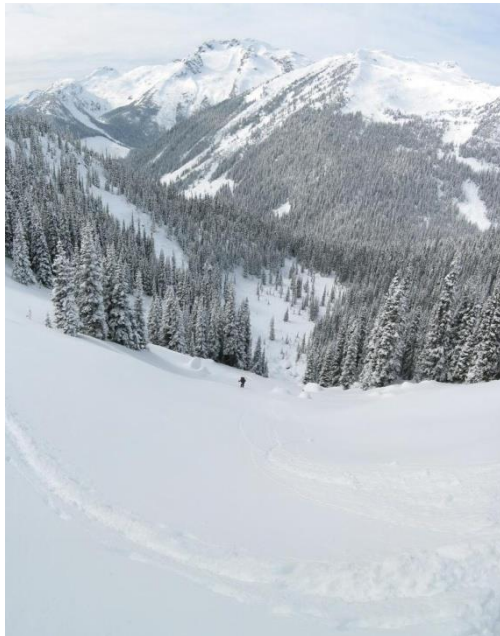
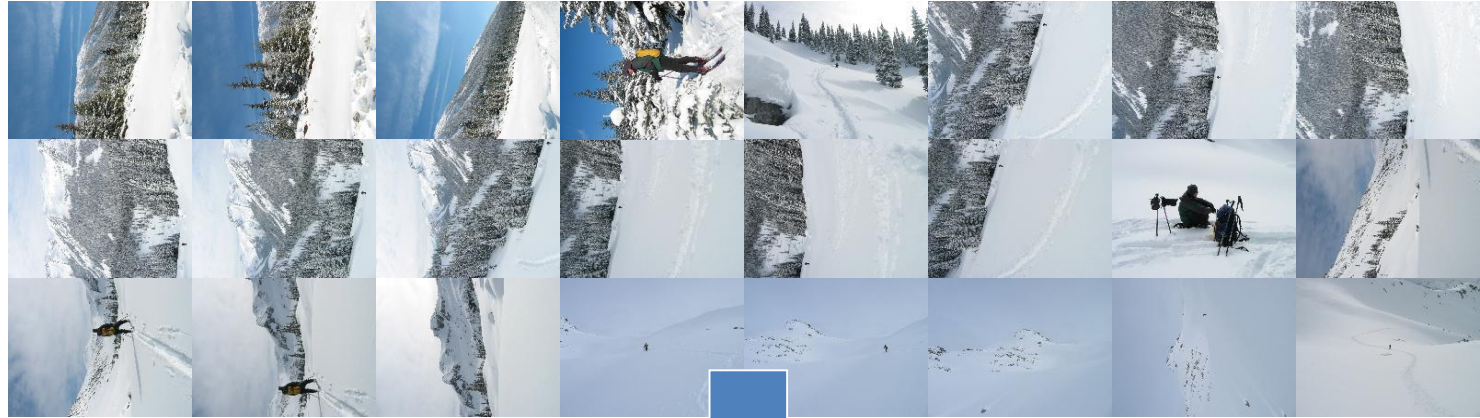
# Finding the panoramas



# Finding the panoramas



# Finding the panoramas



# Recognizing Panoramas (cont.)

(now we have matched pairs of images)

4. Find connected components
5. For each connected component
  - a) Perform bundle adjustment to solve for rotation  $(\theta_1, \theta_2, \theta_3)$  and focal length  $f$  of all cameras
  - b) Project to a surface (plane, cylinder, or sphere)
  - c) Render with multiband blending



# 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  $\mathbf{H} = \mathbf{K}' \mathbf{R}' \mathbf{R}^{-1} \mathbf{K}^{-1}$

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

$$error = \sum_1^N \sum_j^{M_i} \sum_k dist(\mathbf{x}', \hat{\mathbf{x}}')$$

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

# Bundle Adjustment

New images initialized with rotation, focal length of the best matching image



# Bundle Adjustment

New images initialized with rotation, focal length of the best matching image



# Blending

- Gain compensation: minimize intensity difference of overlapping pixels
- Blending
  - Pixels near center of image get more weight
  - Multiband blending to prevent blurring

# Multi-band Blending (Laplacian Pyramid)

- Burt & Adelson 1983
  - Blend frequency bands over range  $\propto \lambda$

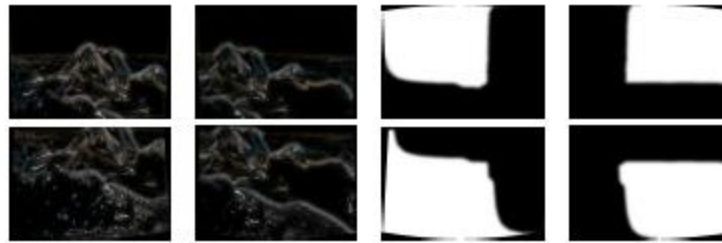




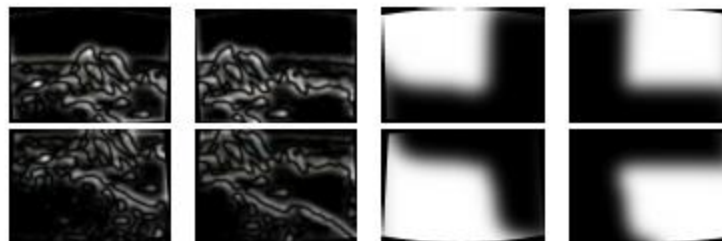
# Multiband blending



(a) Original images and blended result



(b) Band 1 (scale 0 to  $\sigma$ )



(c) Band 2 (scale  $\sigma$  to  $2\sigma$ )



(d) Band 3 (scale lower than  $2\sigma$ )

# Blending comparison (IJCV 2007)



(a) Linear blending



(b) Multi-band blending

# Blending Comparison



(b) Without gain compensation



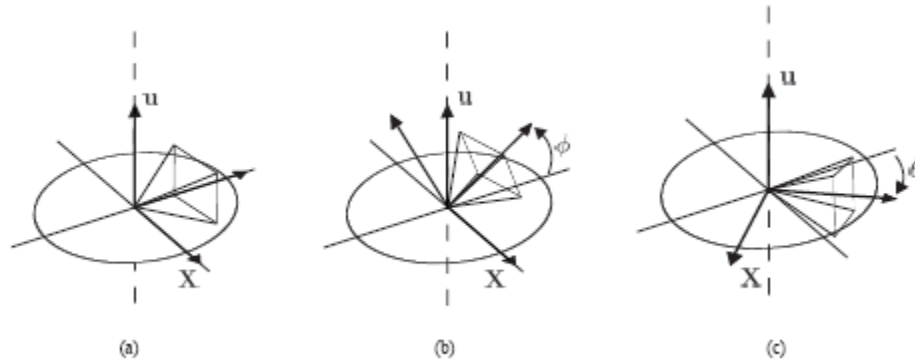
(c) With gain compensation



(d) With gain compensation and multi-band blending

# Straightening

Rectify images so that “up” is vertical



(a) Without automatic straightening



(b) With automatic straightening

# Further reading

Harley and Zisserman: Multi-view Geometry book

- DLT algorithm: HZ p. 91 (alg 4.2), p. 585
- Normalization: HZ p. 107-109 (alg 4.2)
- RANSAC: HZ Sec 4.7, p. 123, alg 4.6
- Tutorial:  
[http://users.cecs.anu.edu.au/~hartley/Papers/CVPR99-tutorial/tut\\_4up.pdf](http://users.cecs.anu.edu.au/~hartley/Papers/CVPR99-tutorial/tut_4up.pdf)
- [Recognising Panoramas](#): Brown and Lowe, IJCV 2007 (also bundle adjustment)



# Tips and Photos from Russ Hewett

# Capturing Panoramic Images

- Tripod vs Handheld
  - Help from modern cameras
  - Leveling tripod
  - GigaPan
  - Or wing it
- Image Sequence
  - Requires a reasonable amount of overlap (at least 15-30%)
  - Enough to overcome lens distortion
- Exposure
  - Consistent exposure between frames
  - Gives smooth transitions
  - Manual exposure
  - Makes consistent exposure of dynamic scenes easier
  - But scenes don't have constant intensity everywhere
- Caution
  - Distortion in lens (Pin Cushion, Barrel, and Fisheye)
  - Polarizing filters
  - Sharpness in image edge / overlap region

# Pike's Peak Highway, CO



# Pike's Peak Highway, CO





# 360 Degrees, Tripod Leveled



# Howth, Ireland

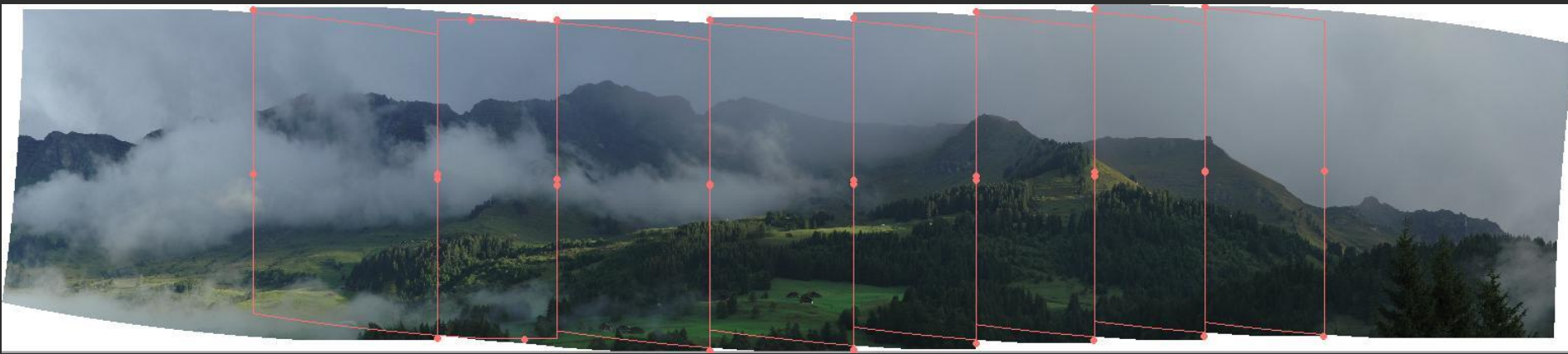




# Handheld Camera



# Handheld Camera





# Les Diablerets, Switzerland



# Macro



# Side of Laptop



# Considerations For Stitching

- Variable intensity across the total scene
- Variable intensity and contrast between frames
- Lens distortion
  - Pin Cushion, Barrel, and Fisheye
  - Profile your lens at the chosen focal length (read from EXIF)
  - Or get a profile from LensFun
- Dynamics/Motion in the scene
  - Causes ghosting
  - Once images are aligned, simply choose from one or the other
- Misalignment
  - Also causes ghosting
  - Pick better control points
- Visually pleasing result
  - Super wide panoramas are not always 'pleasant' to look at
  - Crop to golden ratio, 10:3, or something else visually pleasing



# Ghosting and Variable Intensity





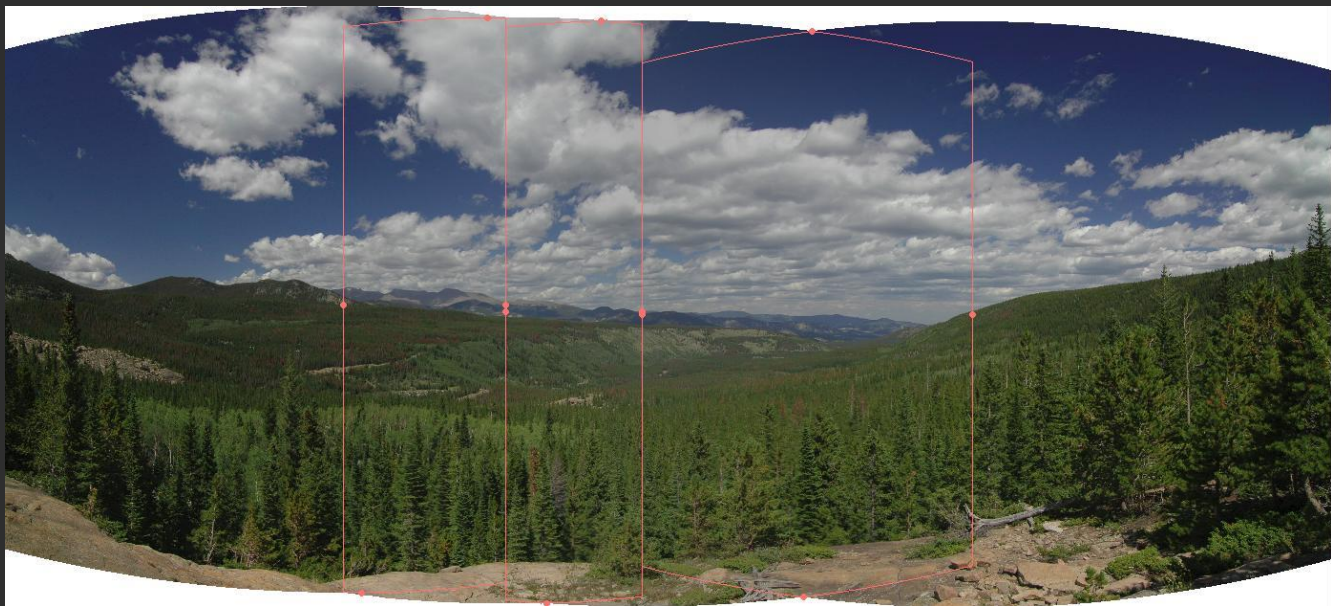


Photo: Russell J. Hewett



# Ghosting From Motion



# Motion Between Frames





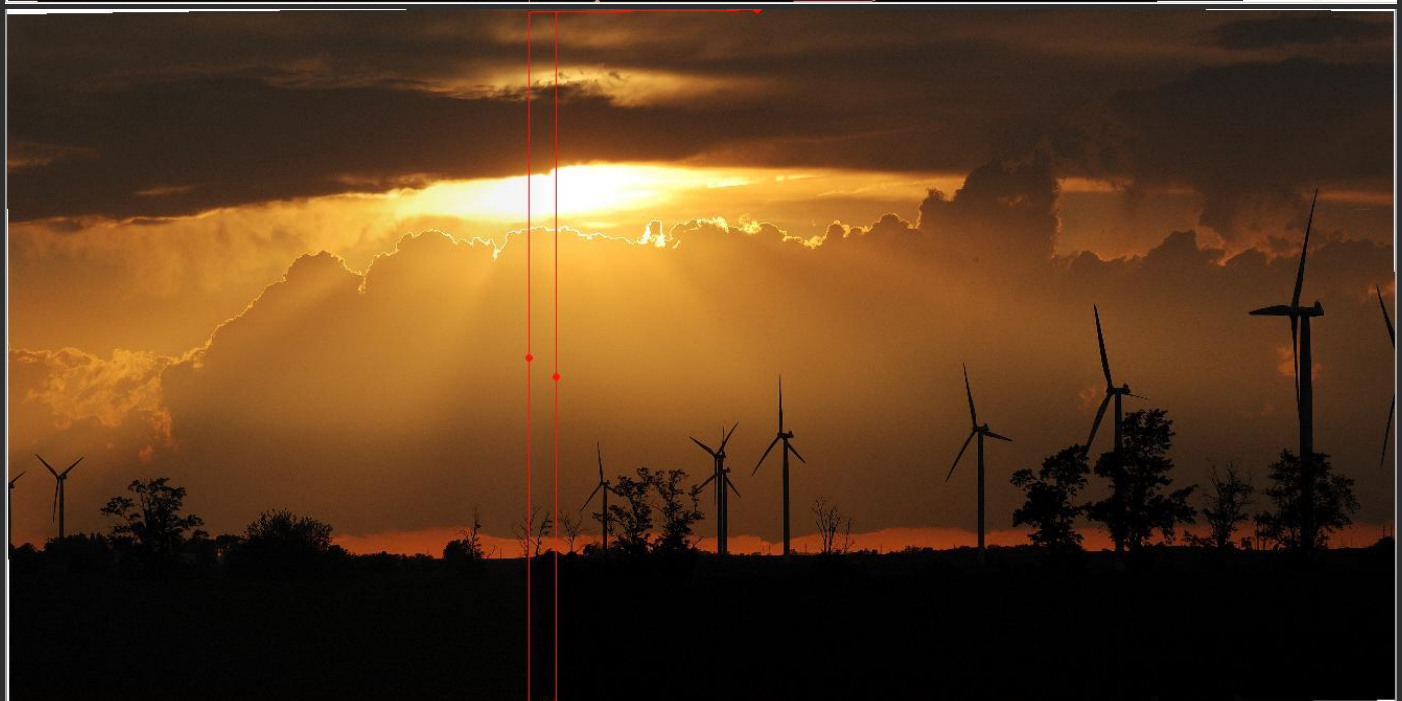


Photo: Russell J. Hewett



# Gibson City, IL



# Mount Blanca, CO



# Mount Blanca, CO



# Things to remember

- Homography relates rotating cameras
  - Homography is plane to plane mapping
- Recover homography using RANSAC and normalized DLT
- Can choose surface of projection: cylinder, plane, and sphere are most common
- Lots of room for tweaking (blending, straightening, etc.)



# Next class

- Object recognition and retrieval