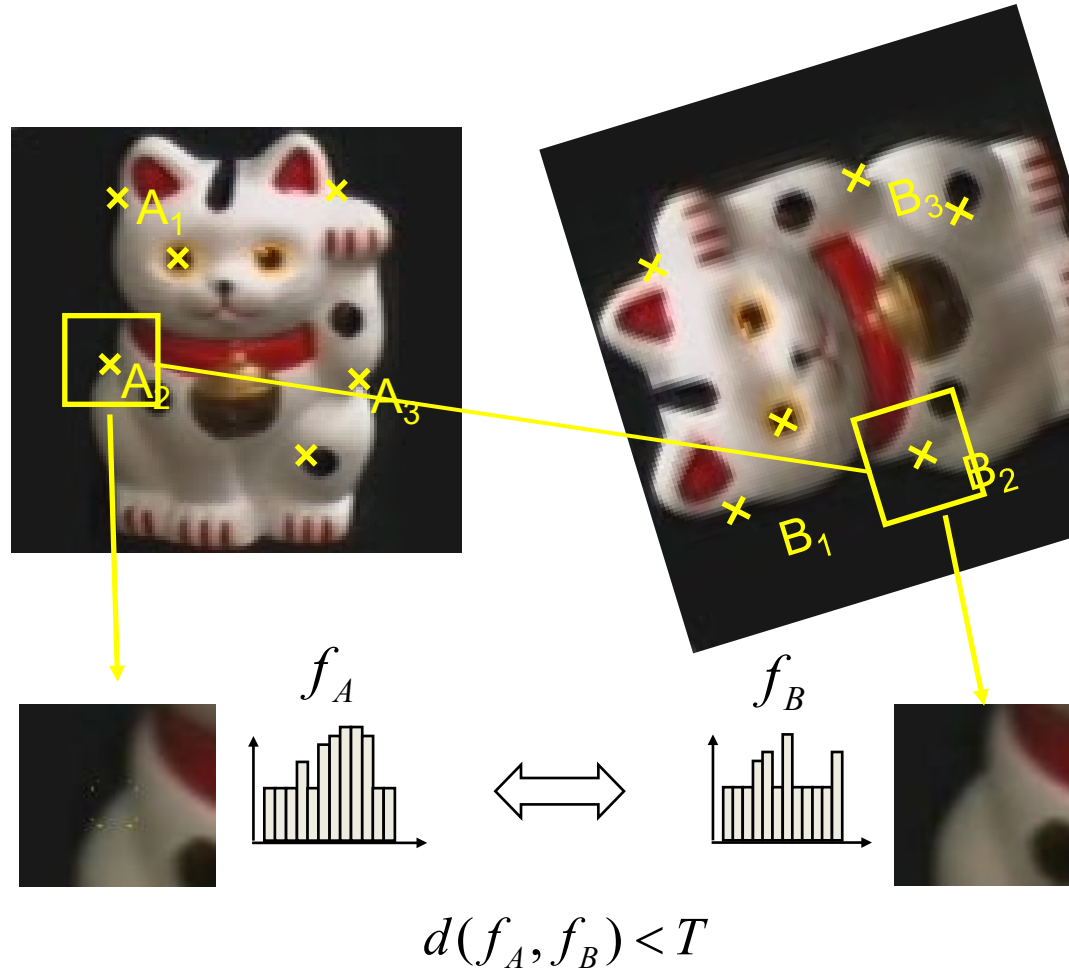


Image Stitching



Computational Photography
Derek Hoiem, University of Illinois

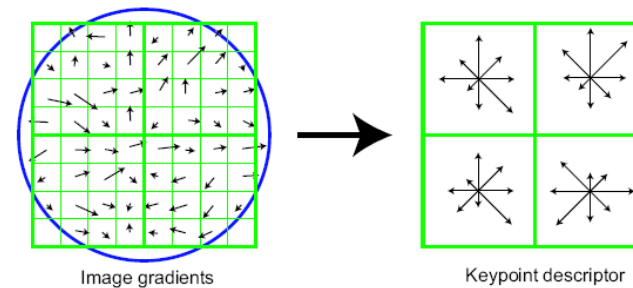
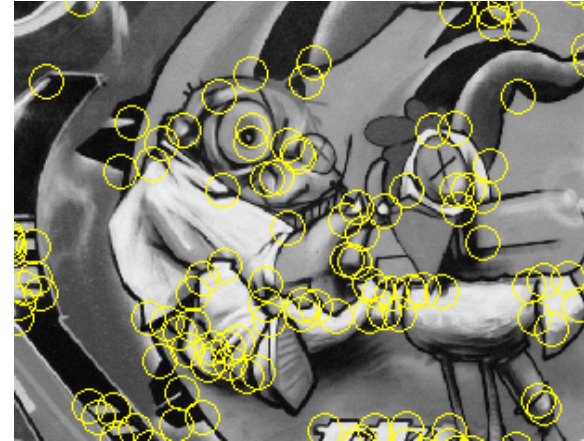
Last Class: Keypoint Matching



1. Find a set of distinctive keypoints
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

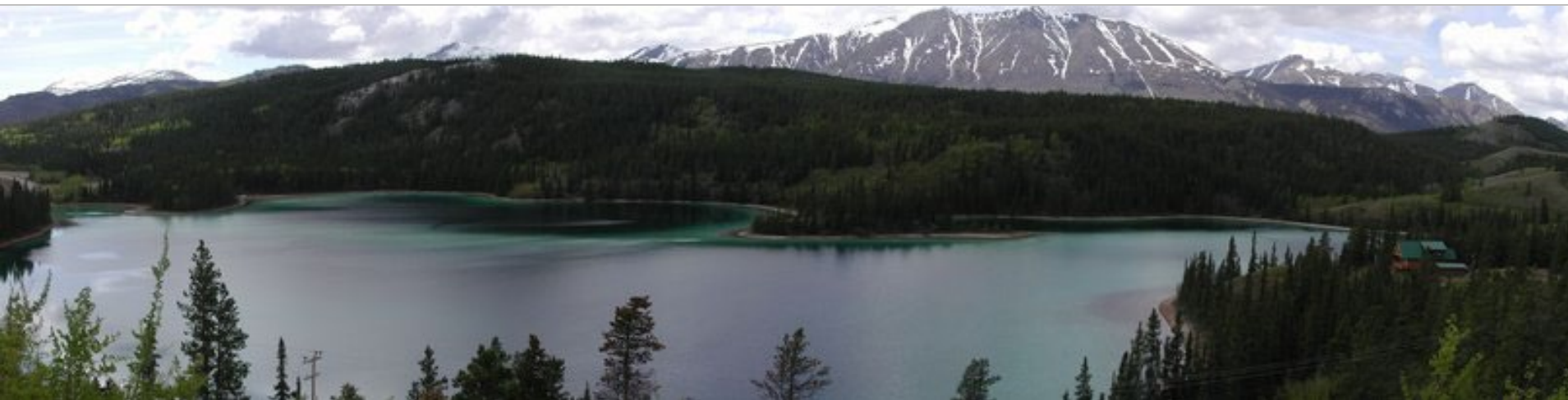
Last Class: Summary

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

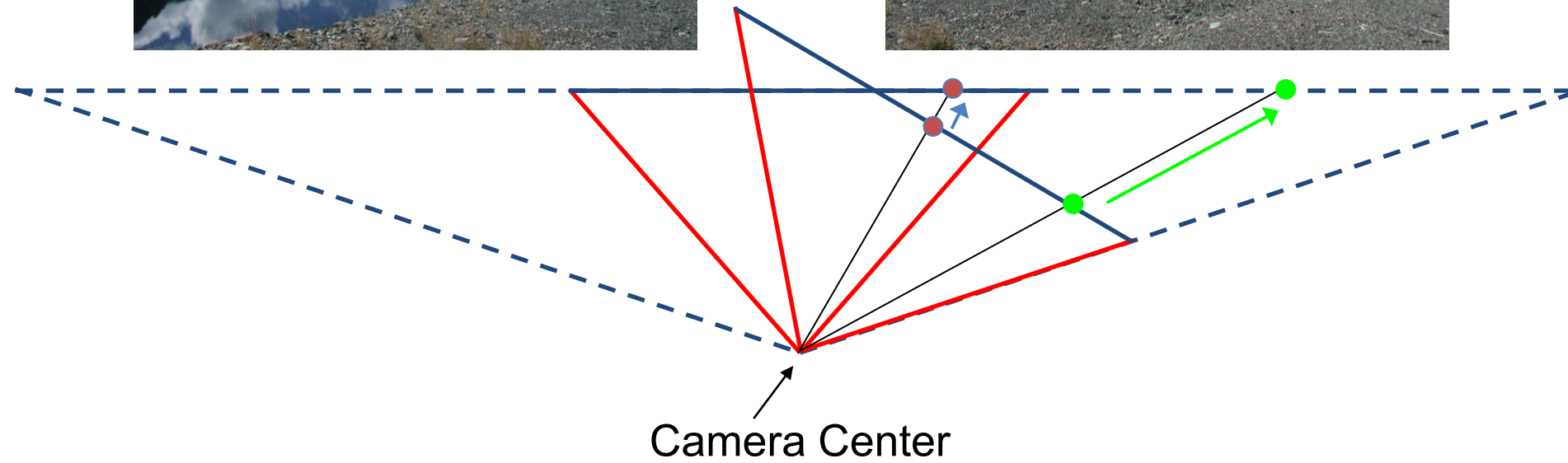
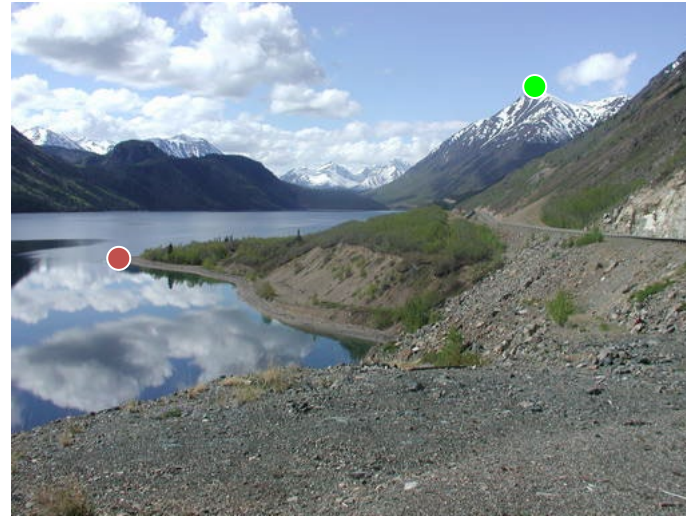
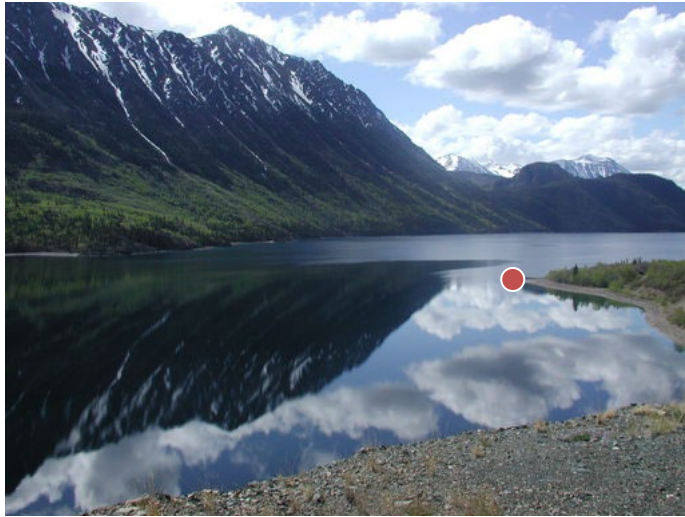


Today: Image Stitching

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

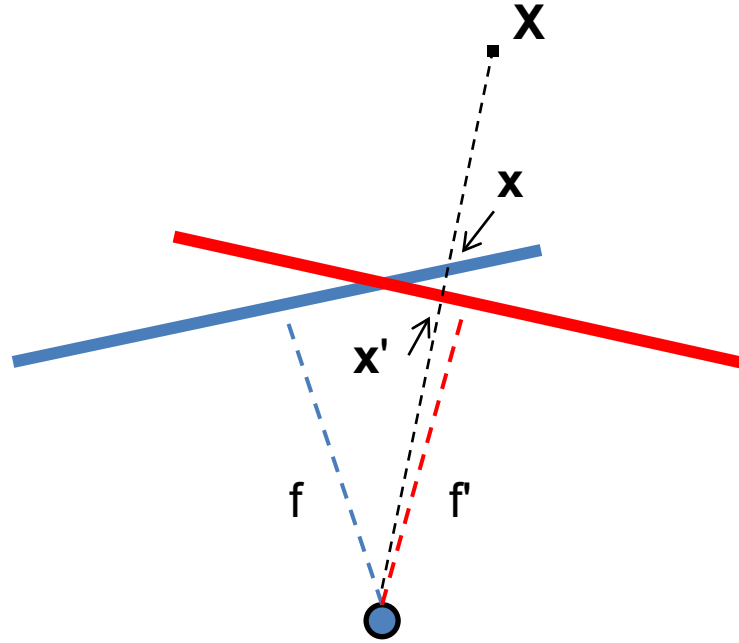


Views from rotating camera



Correspondence of rotating camera

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



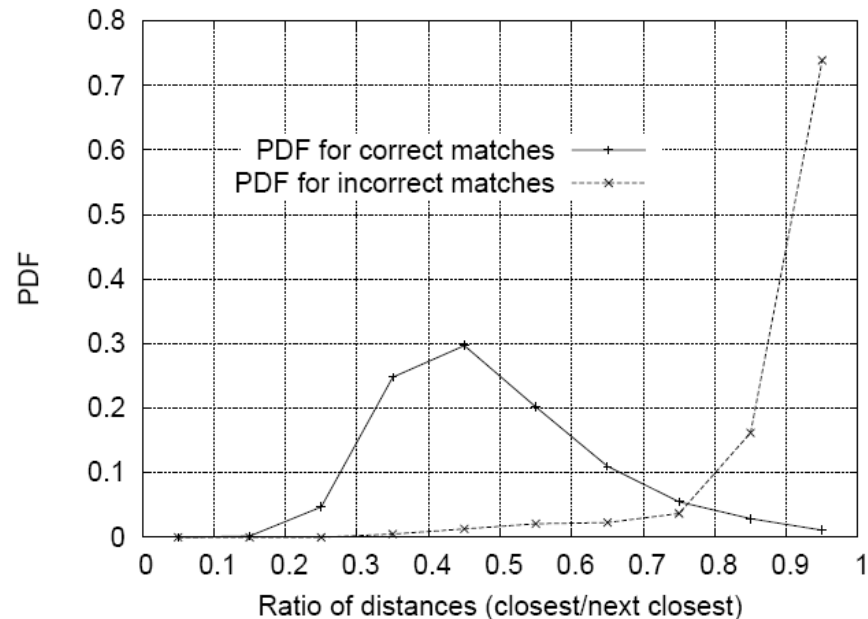
- $x' = Hx$ where $H = K' R' R^{-1} K^{-1}$
- Typically only R and f will change (4 parameters), but, in general, H has 8 parameters

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 2nd most similar) $\frac{d1}{d2} < thresh$



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

$$\begin{bmatrix} -u_1 & -v_1 & -1 & 0 & 0 & 0 & u_1u'_1 & v_1u'_1 & u'_1 \\ 0 & 0 & 0 & -u_1 & -v_1 & -1 & u_1v'_1 & v_1v'_1 & v'_1 \\ & & & \vdots & & & & & \\ 0 & 0 & 0 & -u_n & -v_n & -1 & u_nv'_n & v_nv'_n & v'_n \end{bmatrix} \mathbf{h} = \mathbf{0} \Rightarrow \mathbf{A}\mathbf{h} = \mathbf{0}$$

- Apply SVD: $\mathbf{UDV}^T = \mathbf{A}$
- $\mathbf{h} = \mathbf{V}_{\text{smallest}}$ (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 ≈ 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 \mathbf{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 **H**?

Automatic Homography Estimation with RANSAC

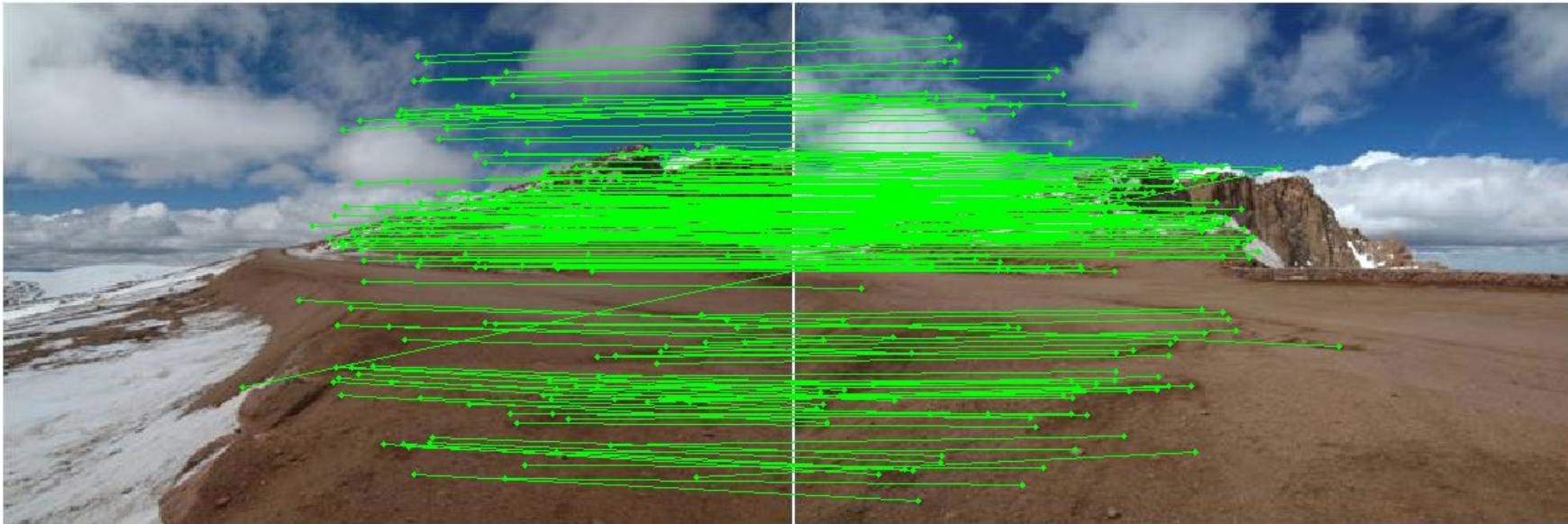
1. Choose number of iterations N
2. Choose 4 random potential matches
3. Compute **H** using normalized DLT
4. Project points from \mathbf{x} to \mathbf{x}' for each potentially matching pair: $\mathbf{x}^p_i = \mathbf{H}\mathbf{x}_i$
5. Count points with projected distance $< t$
 - E.g., $t = 3$ pixels
$$\sqrt{(u'_i - u_i^p)^2 + (v'_i - v_i^p)^2} < t$$
6. Repeat steps 2-5 N times
 - Choose **H** with most inliers

Automatic Image Stitching

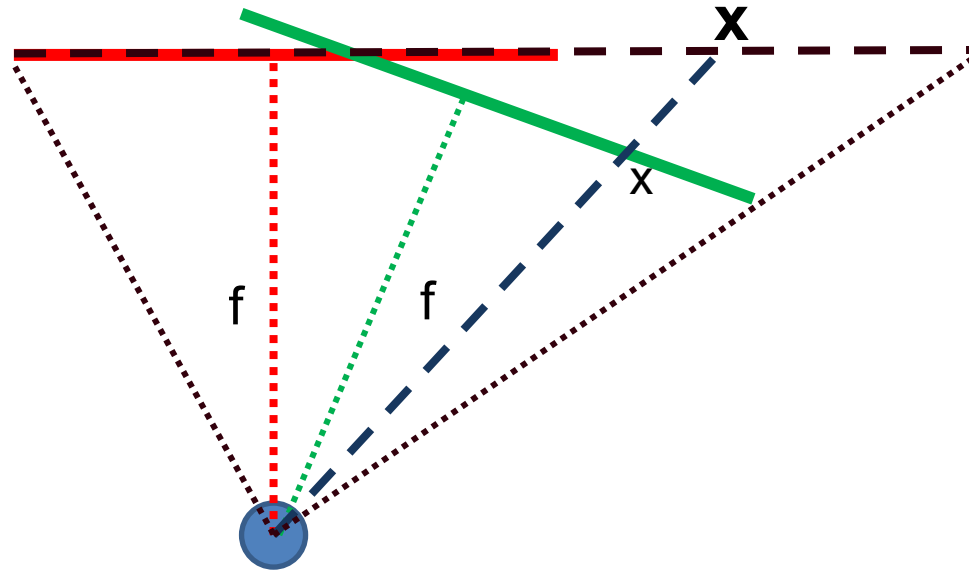
1. Compute interest points on each image
2. Find candidate matches
3. Estimate homography \mathbf{H} using matched points and RANSAC with normalized DLT
4. Project each image onto the same surface and blend

Choosing a Projection Surface

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

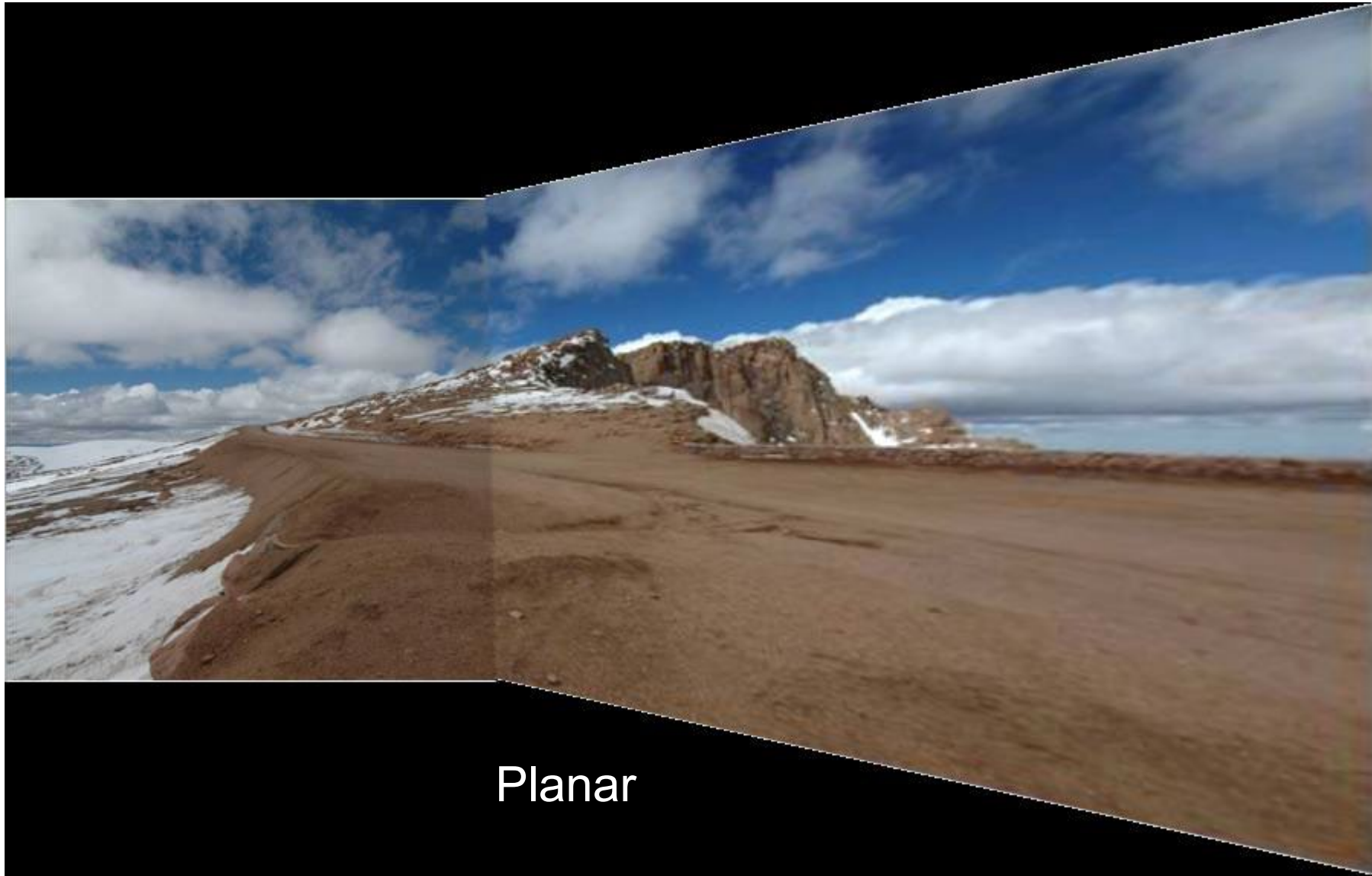


Planar Mapping



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

Planar Projection

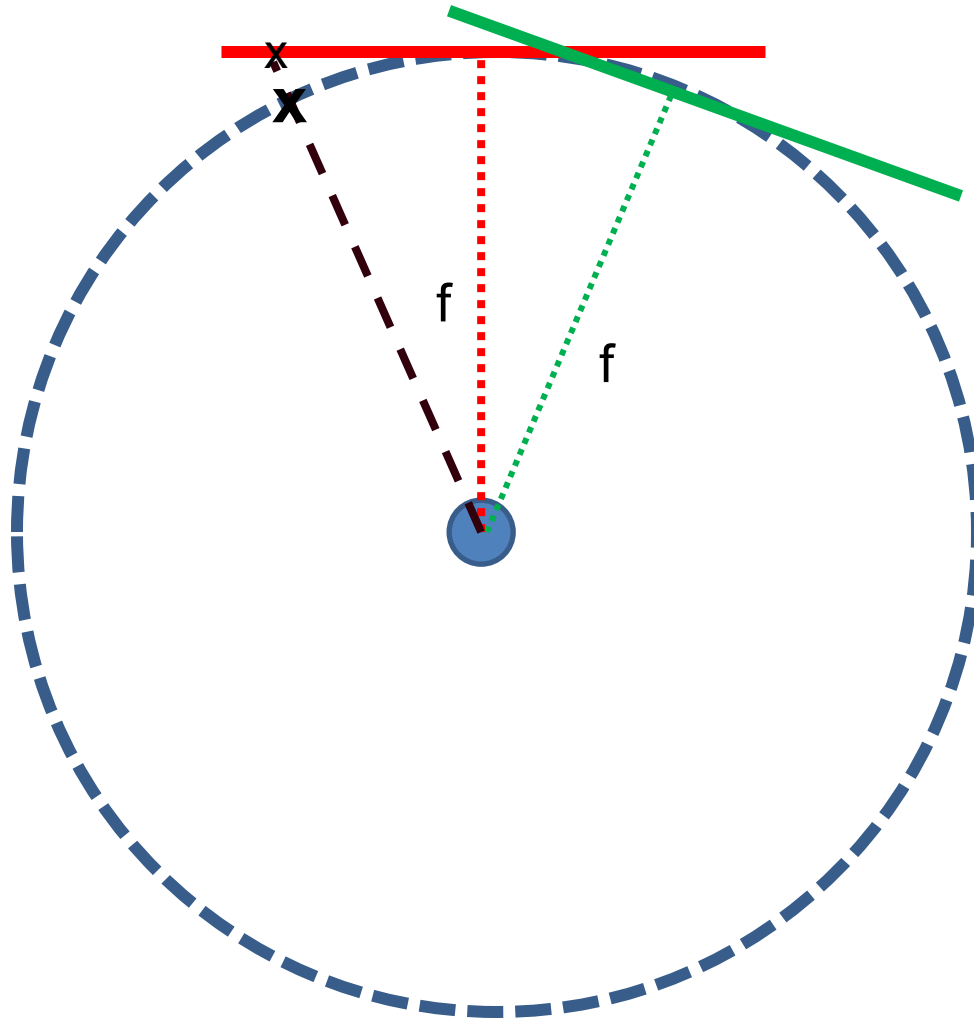


Planar Projection

Planar



Cylindrical Mapping



- 1) Convert pixel coordinate \mathbf{p} to bearing: $\tilde{\mathbf{p}} = K^{-1}\mathbf{p}$, $\hat{\mathbf{p}} = \tilde{\mathbf{p}}/\|\tilde{\mathbf{p}}\|$
- 2) Compute theta, h: $\theta = \tan^{-1}(\tilde{\mathbf{p}}_x/f)$, $h = \tilde{\mathbf{p}}_y f$
- 3) For non-reference image, adjust for relative rotation/height

Cylindrical Projection

Cylindrical

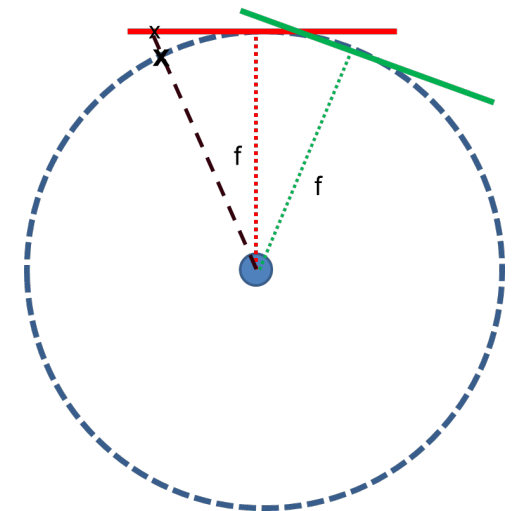
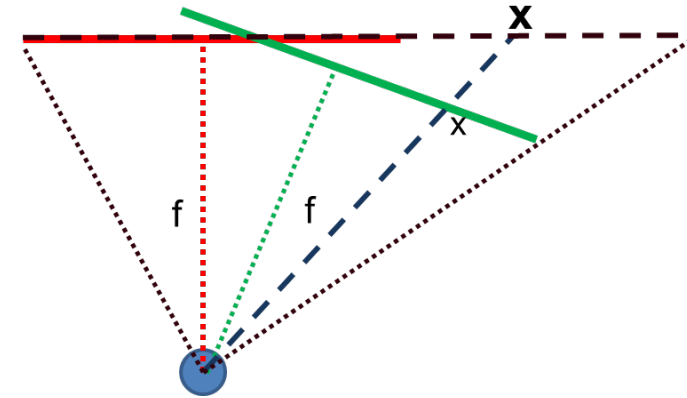


Cylindrical Projection

Cylindrical

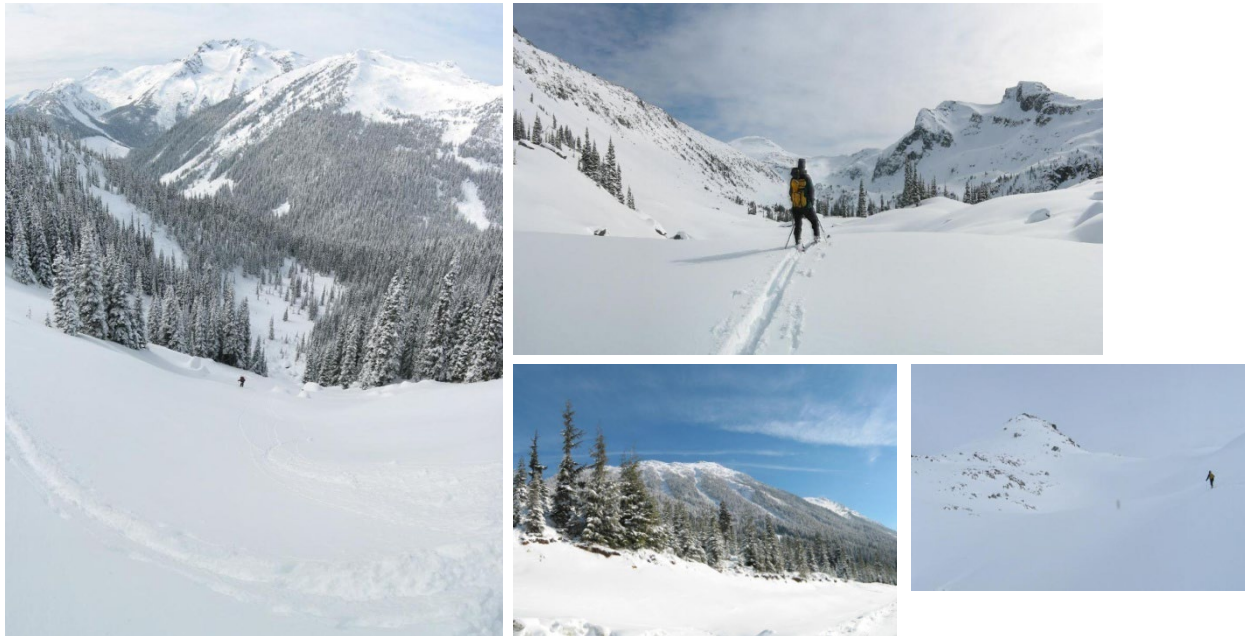


Planar vs. Cylindrical Projection



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



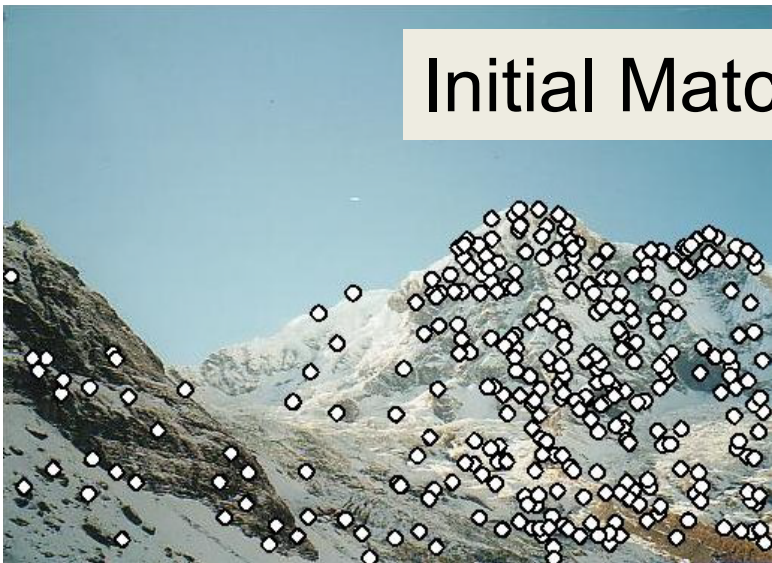
keypoints in
overlapping area



RANSAC for Homography



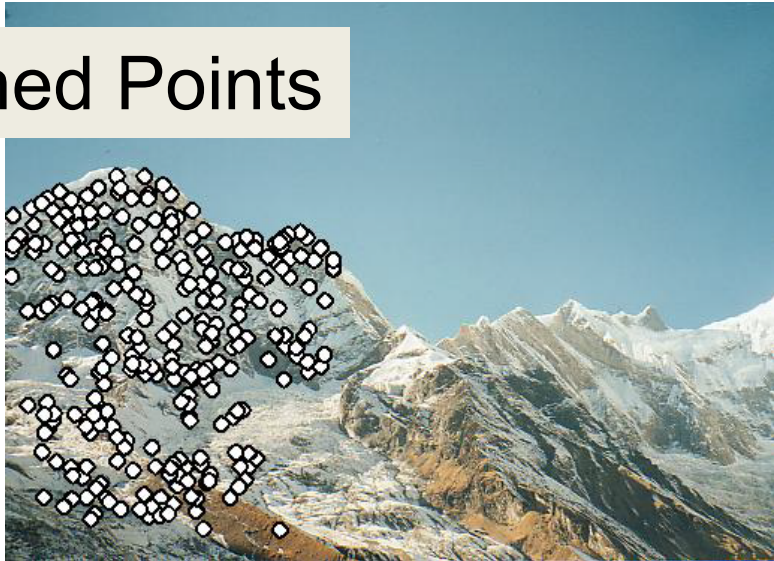
Initial Matched Points



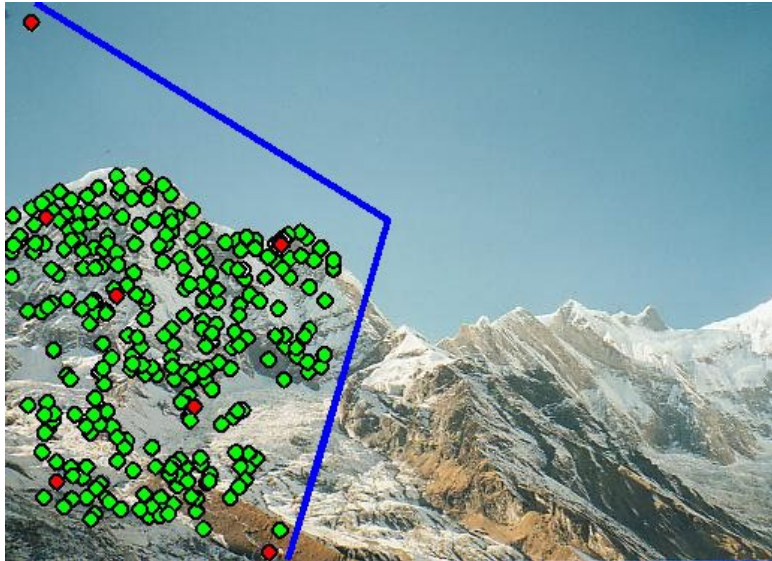
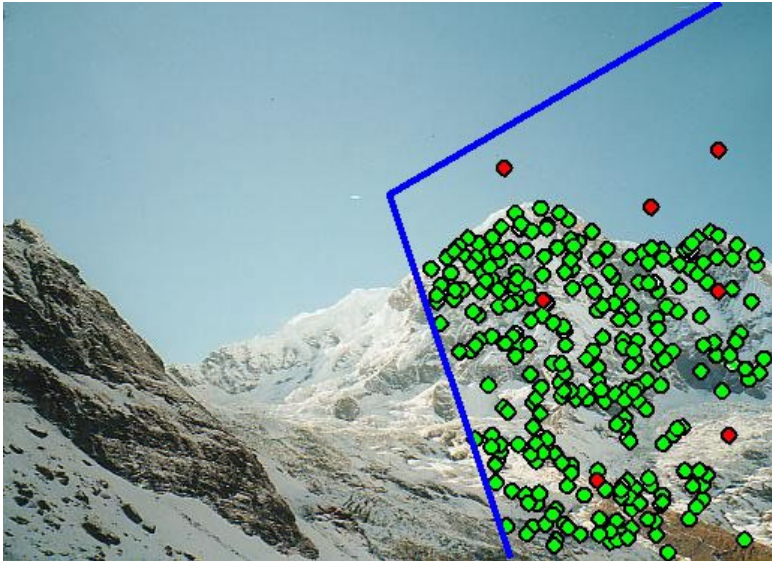
RANSAC for Homography



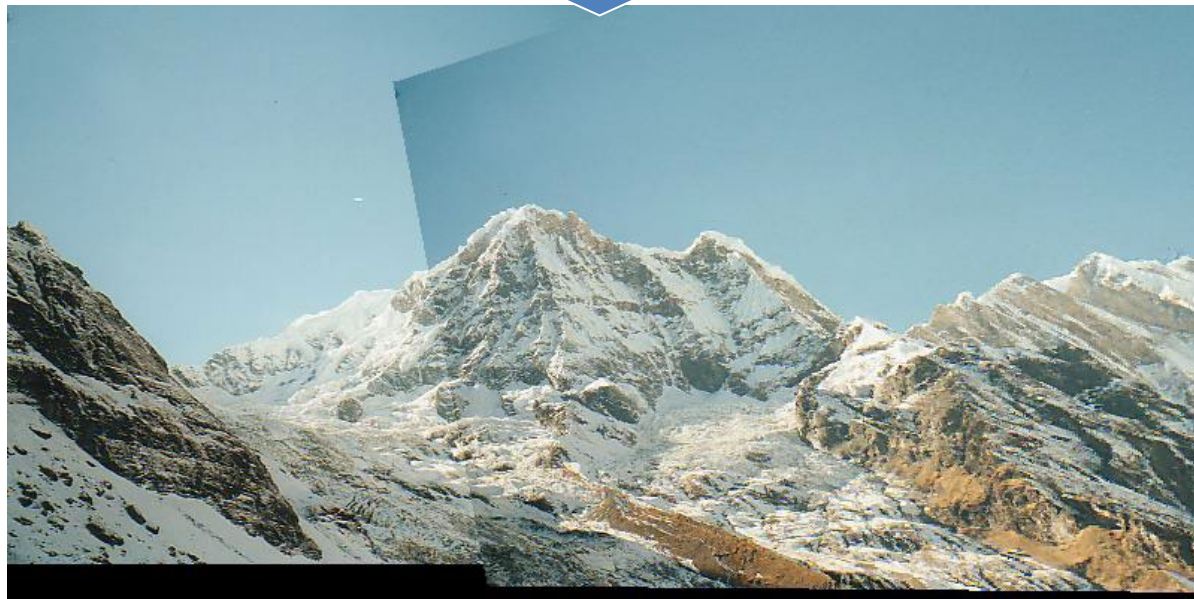
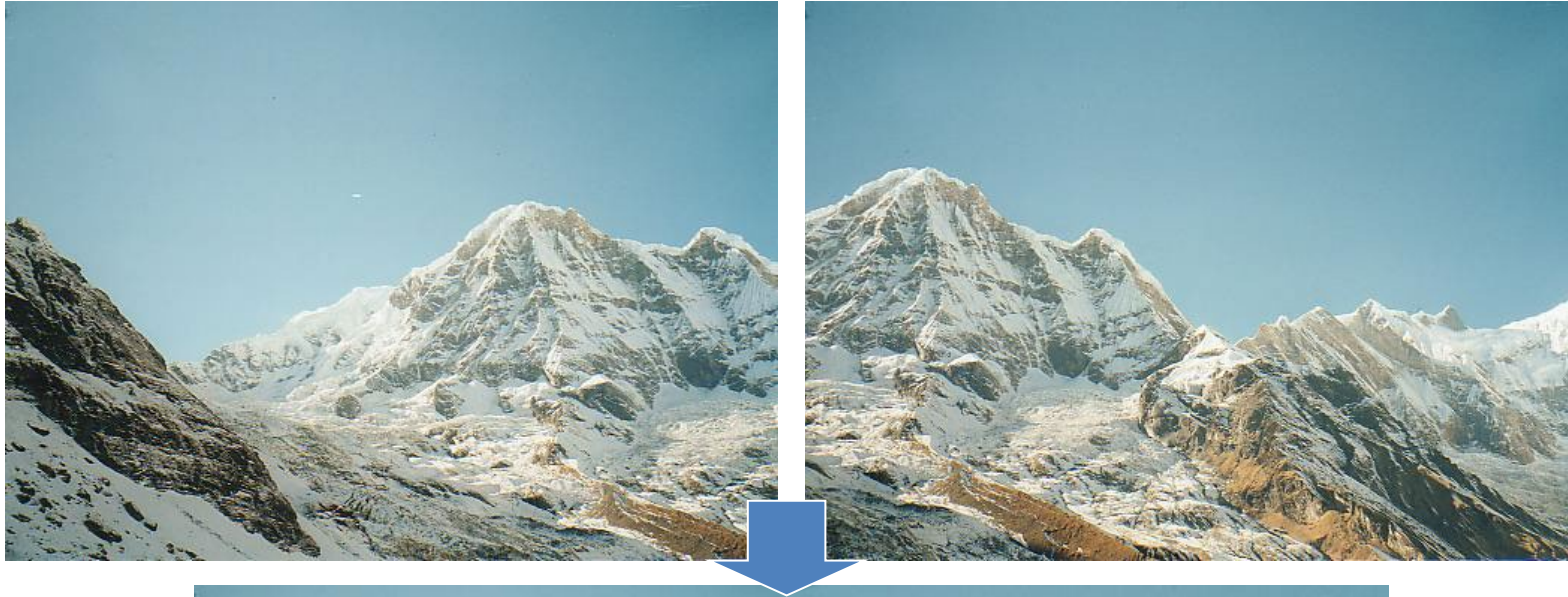
Final Matched Points



Verification



RANSAC for Homography

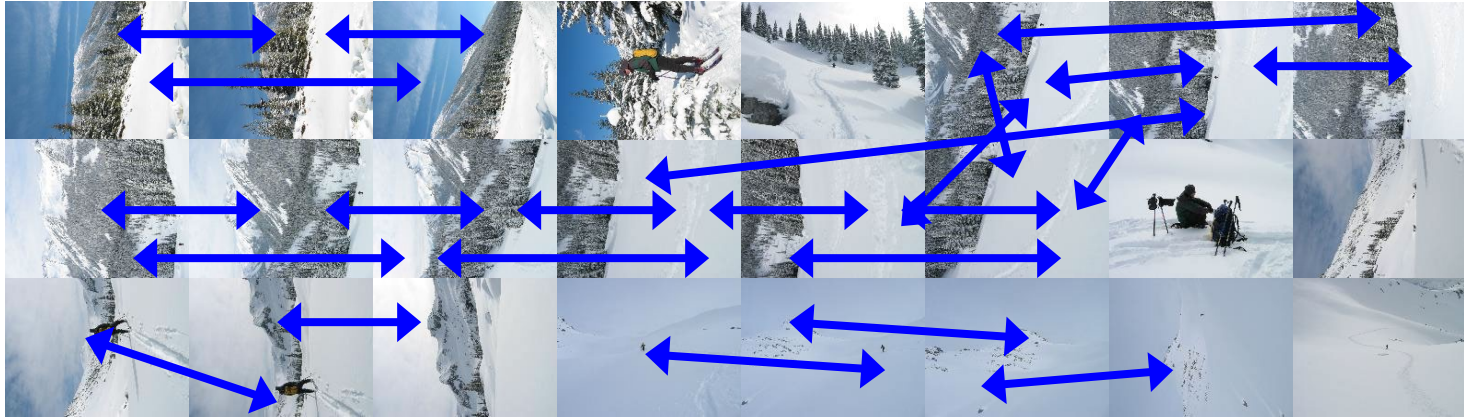


Recognizing Panoramas (cont.)

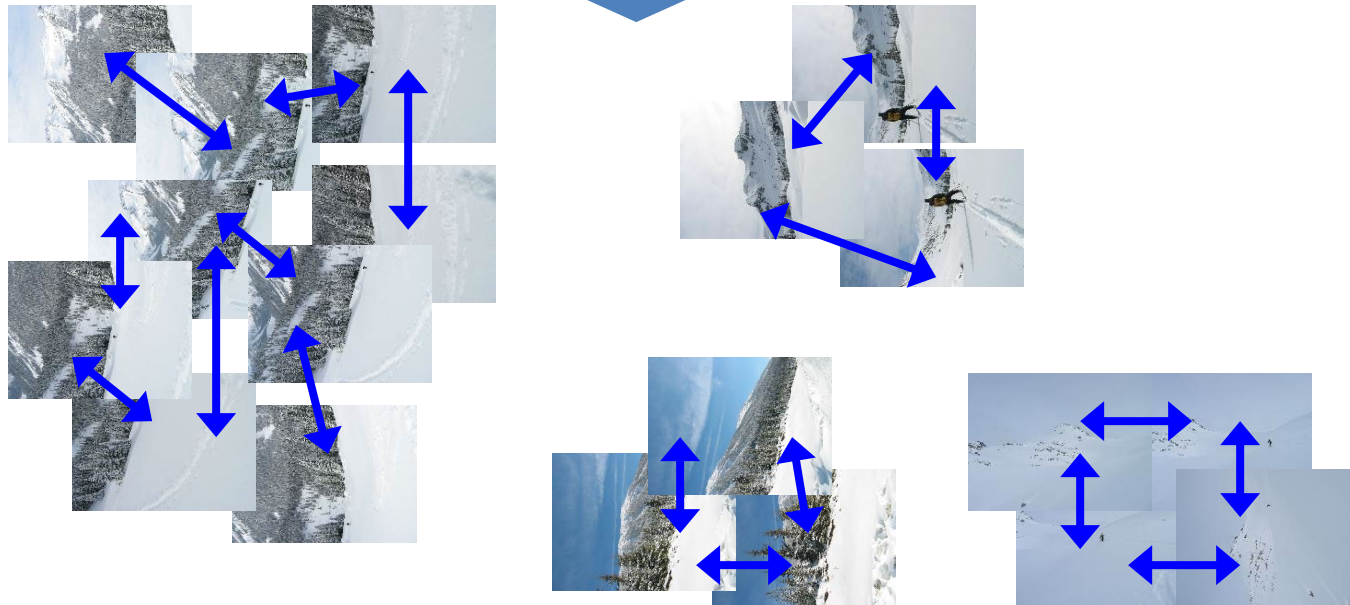
(now we have matched pairs of images)

4. Find connected components

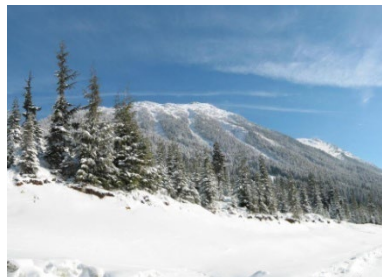
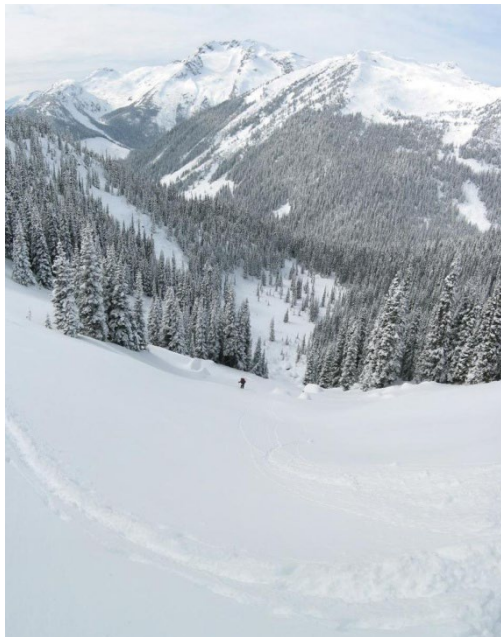
Finding the panoramas



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



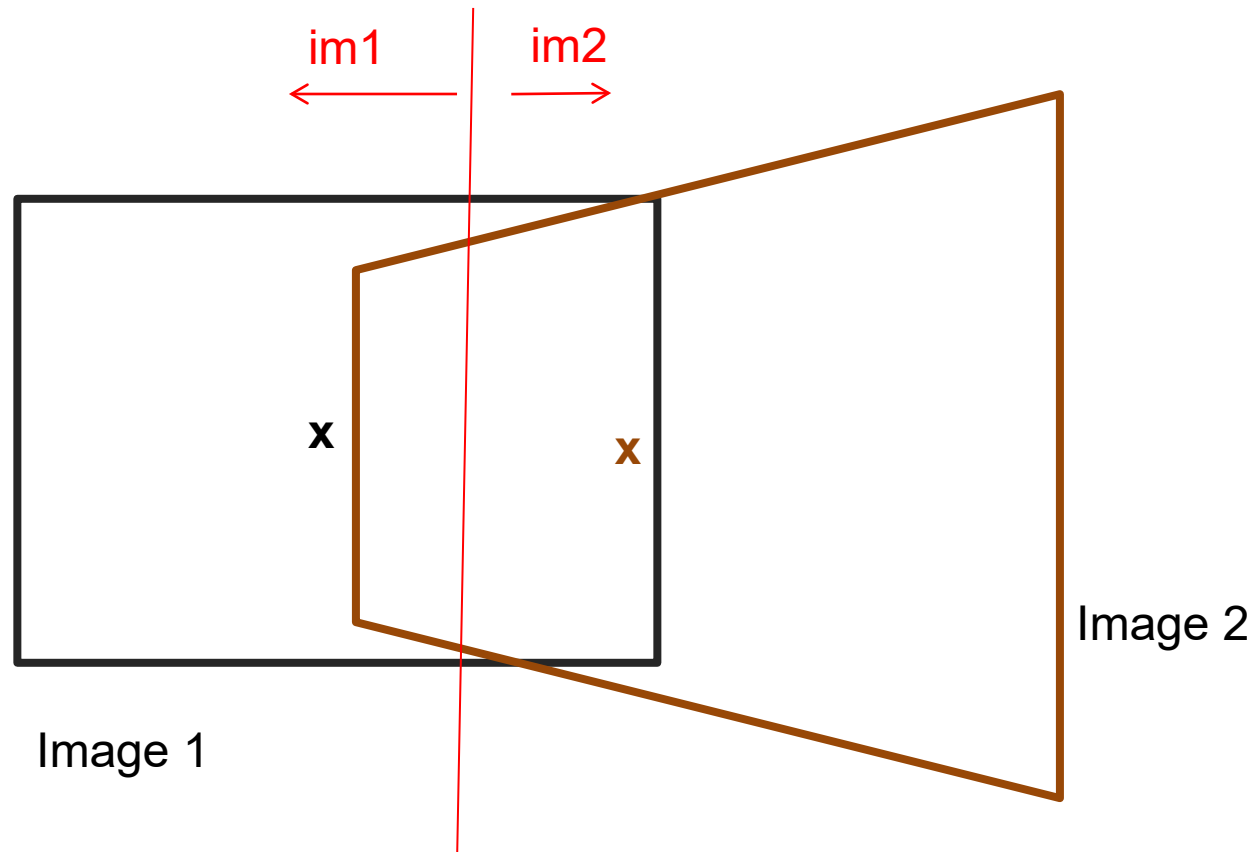
Details to make it look good



- Choosing seams
- Blending

Choosing seams

- Easy method
 - Assign each pixel to image with nearest center



Choosing seams

- Easy method

- Assign each pixel to image with nearest center

- Create a mask:

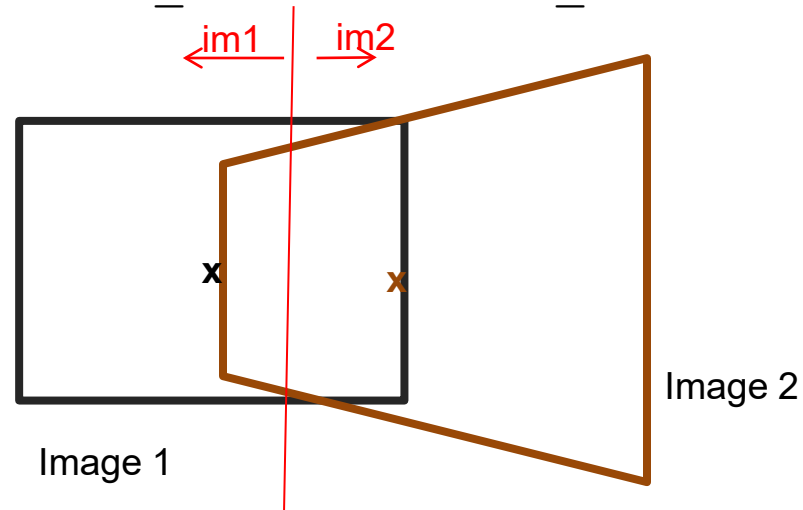
- $\text{mask}(y, x) = 1$ iff pixel should come from im1

- Smooth boundaries (called “feathering”):

- $\text{mask_sm} = \text{imfilter}(\text{mask}, \text{gausfil});$

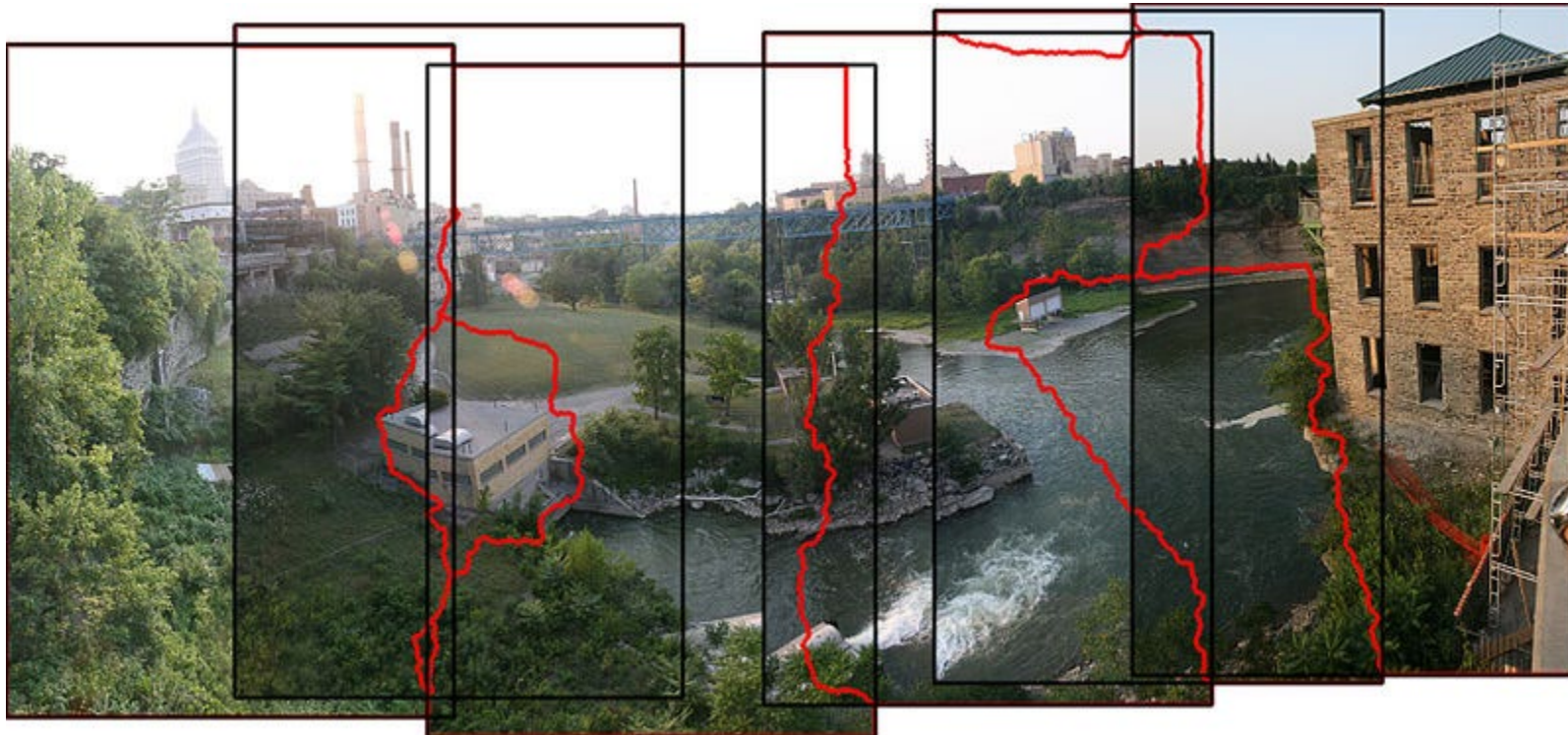
- Composite

- $\text{imblend} = \text{im1_c}.*\text{mask} + \text{im2_c}.*(1-\text{mask});$



Choosing seams

- Better method: dynamic program to find seam along well-matched regions



Gain compensation

- Simple gain adjustment
 - Compute average RGB intensity of each image in overlapping region
 - Normalize intensities by ratio of averages



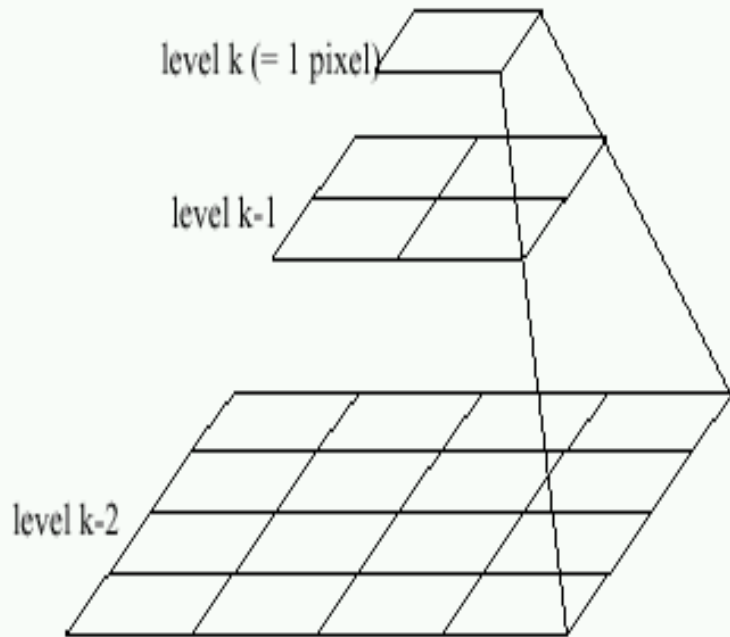
Multi-band (aka Laplacian Pyramid) Blending

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

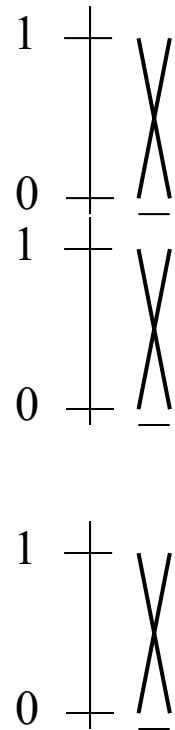


Multiband Blending with Laplacian Pyramid

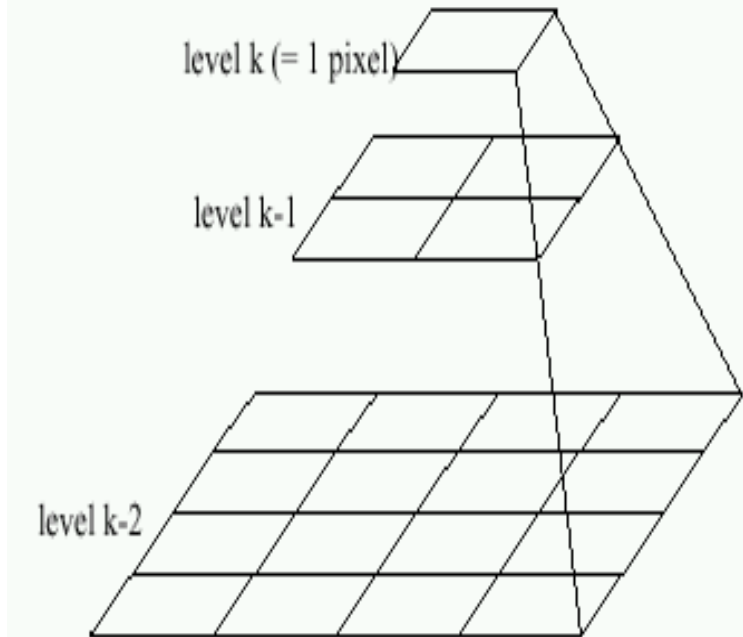
- At low frequencies, blend slowly
- At high frequencies, blend quickly



Left pyramid



blend



Right pyramid

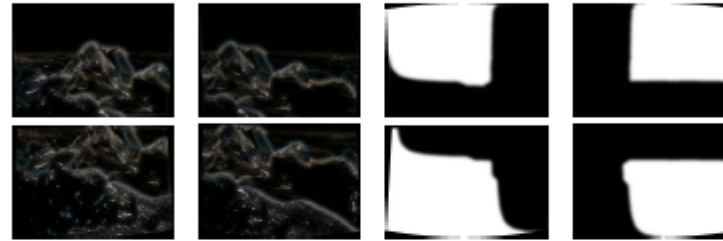
Multiband blending

1. Compute Laplacian pyramid of images and mask
2. Create blended image at each level of pyramid
3. Reconstruct complete image

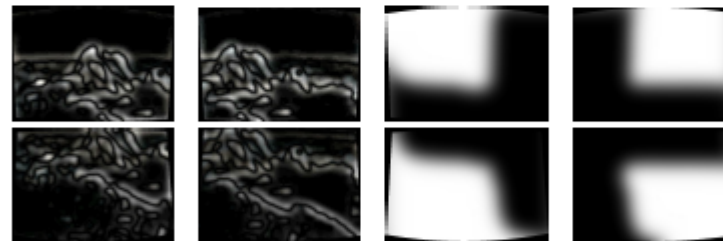
Laplacian pyramids



(a) Original images and blended result



(b) Band 1 (scale 0 to σ)



(c) Band 2 (scale σ to 2σ)



(d) Band 3 (scale lower than 2σ)

Blending comparison (IJCV 2007)



(a) Linear blending



(b) Multi-band blending

Blending Comparison



(b) Without gain compensation



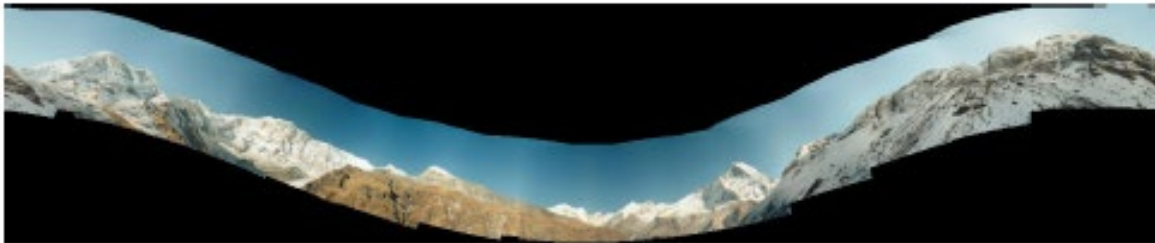
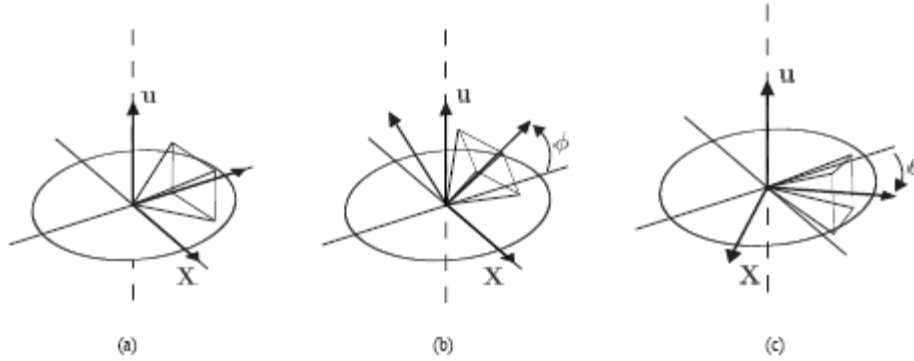
(c) With gain compensation



(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
- [Recognising Panoramas](#): Brown and Lowe, IJCV 2007 (also bundle adjustment)

How does iphone panoramic stitching work?

- Capture images at 30 fps
- Stitch the central 1/8 of a selection of images
 - Select which images to stitch using the accelerometer and frame-to-frame matching
 - Faster and avoids radial distortion that often occurs towards corners of images
- Alignment
 - Initially, perform cross-correlation of small patches aided by accelerometer to find good regions for matching
 - Register by matching points (KLT tracking or RANSAC with FAST (similar to SIFT) points) or correlational matching
- Blending
 - Linear (or similar) blending, using a face detector to avoid blurring face regions and choose good face shots (not blinking, etc)

Project 5

Input video:

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>

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



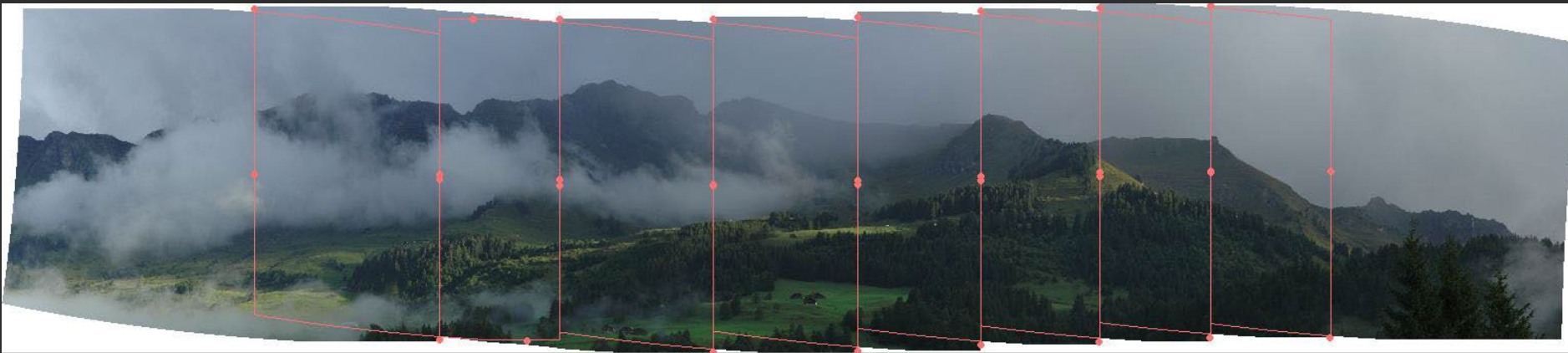
Howth, Ireland



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



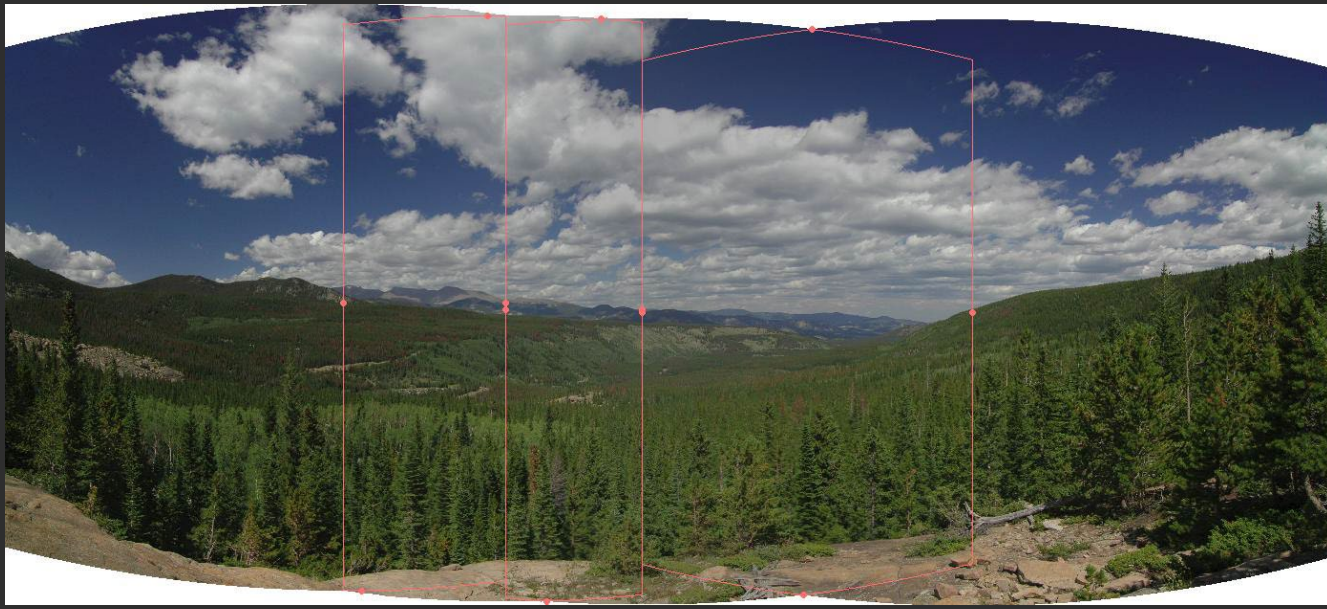


Photo: Russell J. Hewett

Ghosting From Motion



Motion Between Frames



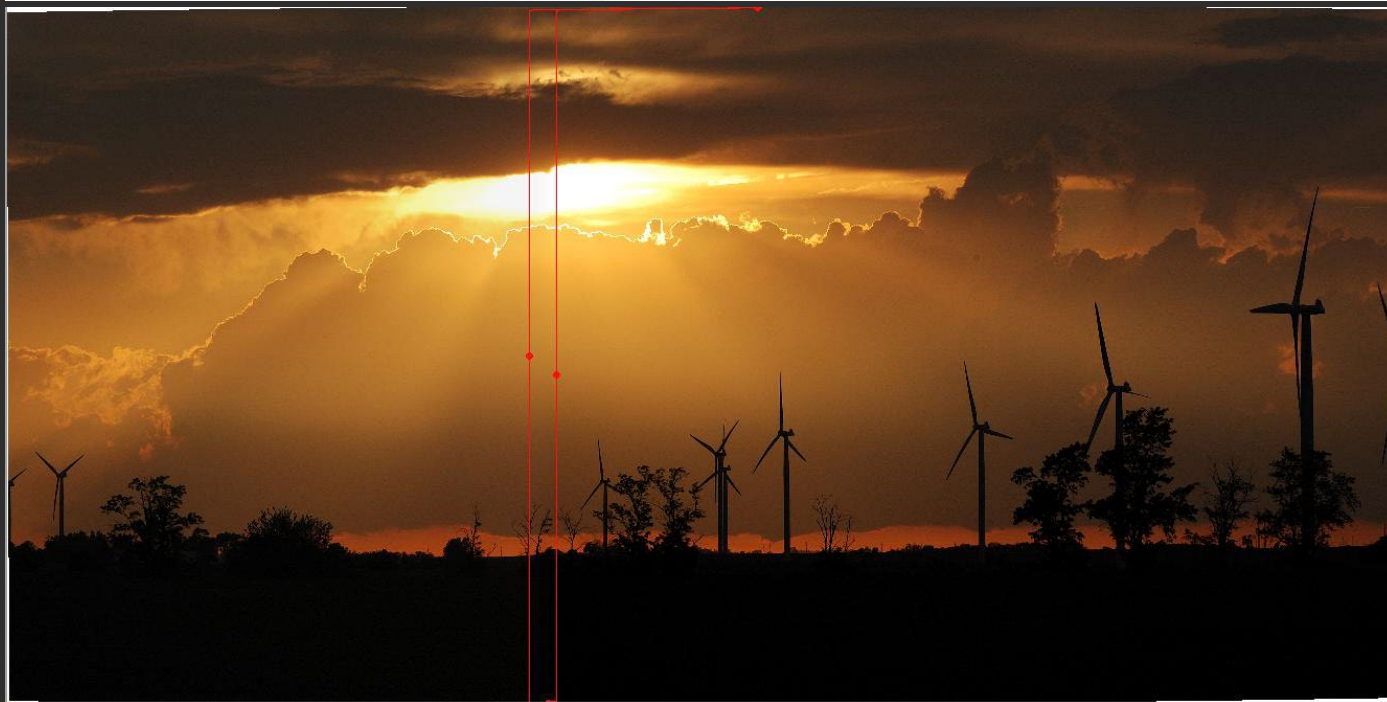


Photo: Russell J. Hewett

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
- Refinement methods (blending, straightening, etc.)

Next class

- Object recognition and retrieval