

Lecture 12: Local Descriptors, SIFT & Single Object Recognition

Professor Fei-Fei Li

Stanford Vision Lab

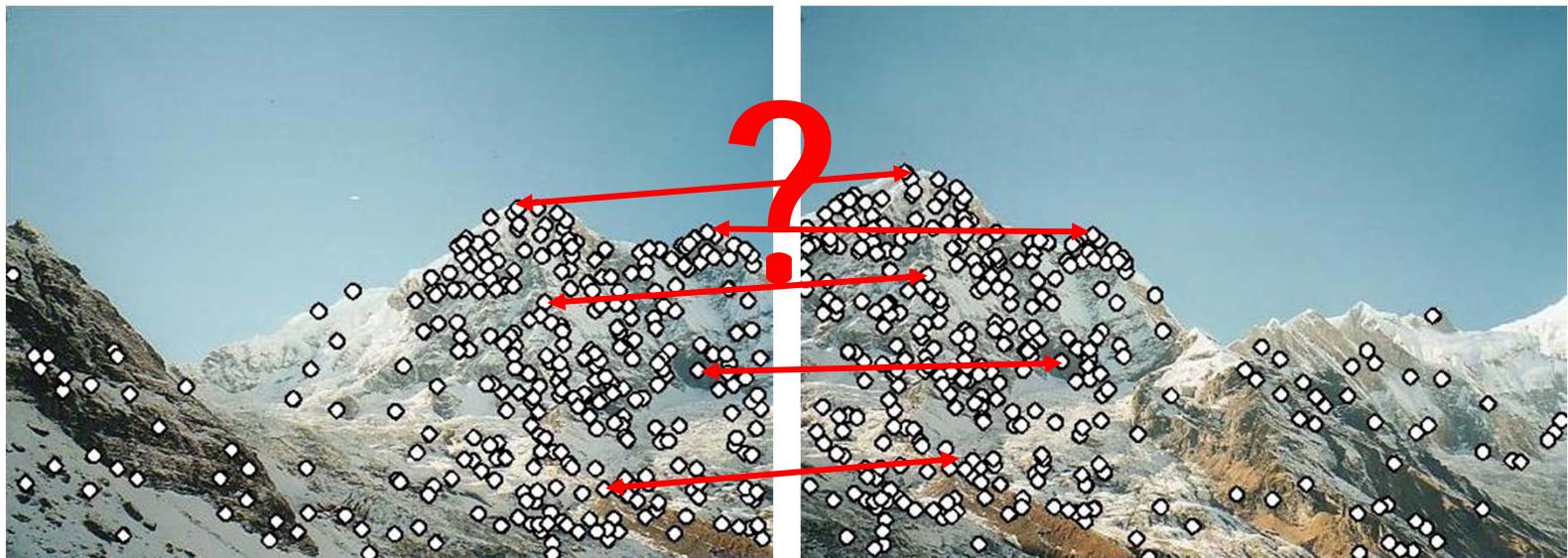
What we will learn today

- Local descriptors
 - SIFT (**Problem Set 3 (Q2)**)
 - An assortment of other descriptors (**Problem Set 3 (Q4)**)
- Recognition and matching with local features
(Problem Set 3 (Q3))
 - Matching objects with local features: David Lowe
 - Panorama stitching: Brown & Lowe
 - Applications

Local Descriptors

- We know how to detect points Last lecture (#11)
- Next question:

How to *describe* them for matching?

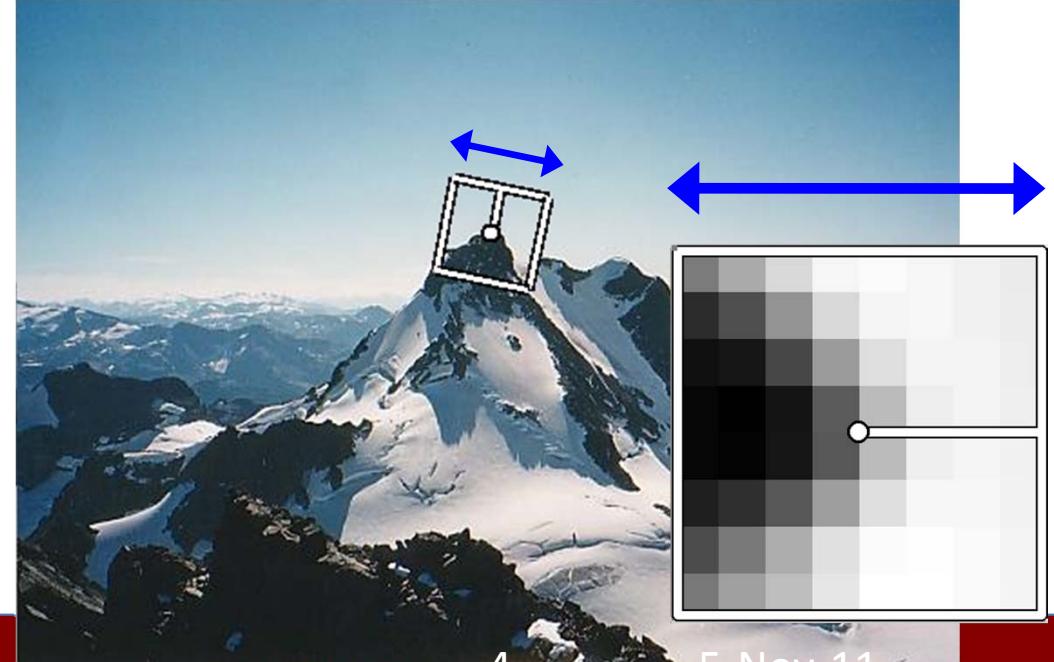
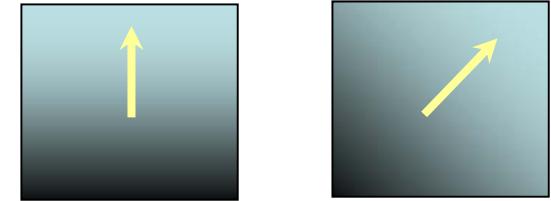


This lecture (#12)

Point descriptor should be:
1. Invariant
2. Distinctive

Rotation Invariant Descriptors

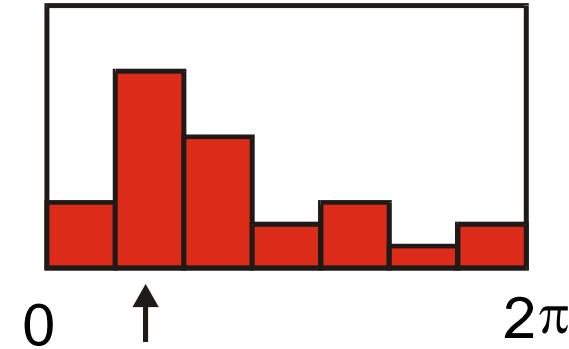
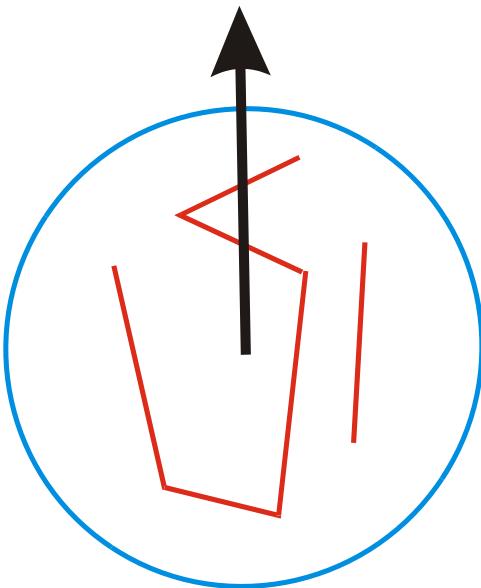
- Find local orientation
 - Dominant direction of gradient for the image patch
- Rotate patch according to this angle
 - This puts the patches into a canonical orientation.



Orientation Normalization: Computation

[Lowe, SIFT, 1999]

- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation



Slide adapted from David Lowe

The Need for Invariance

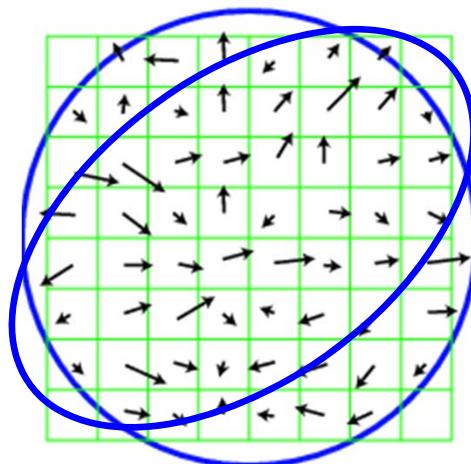


- Up to now, we had invariance to
 - Translation
 - Scale
 - Rotation
- Not sufficient to match regions under viewpoint changes
 - For this, we need also affine adaptation

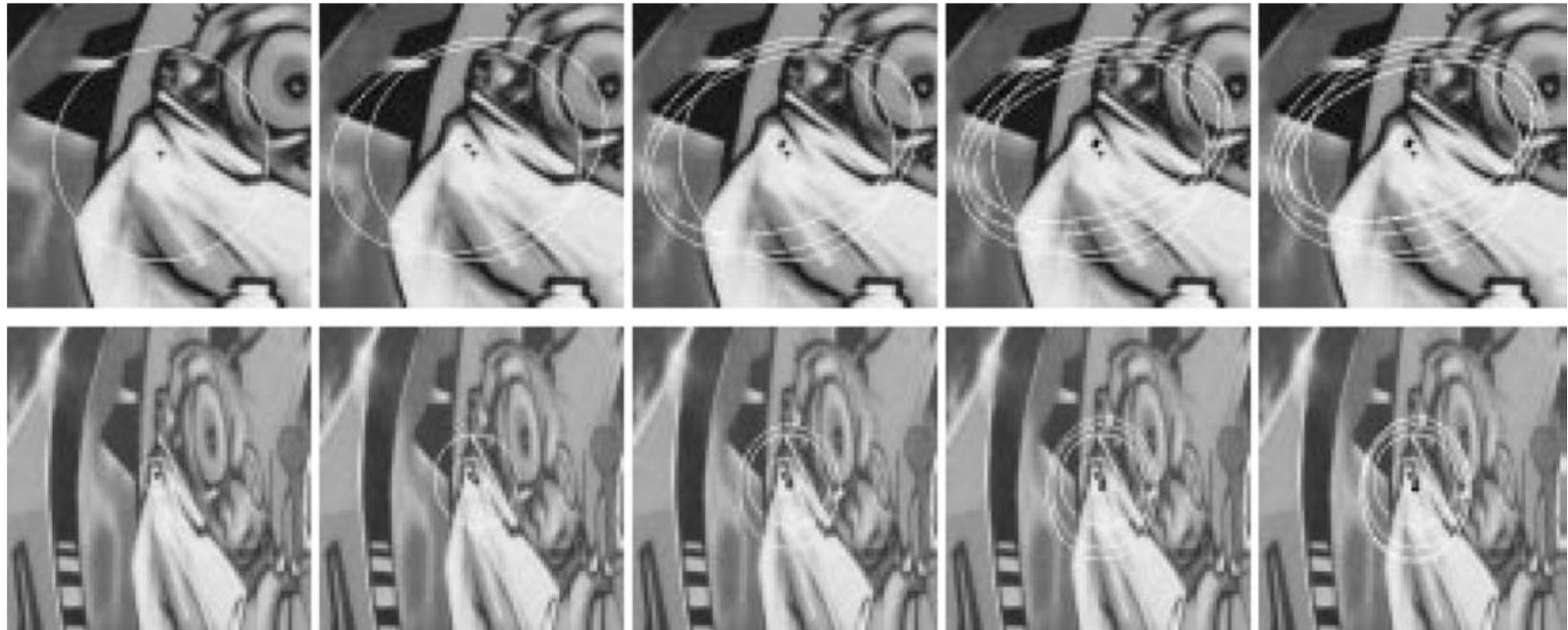
Slide credit: Tinne Tuytelaars

Affine Adaptation

- Problem:
 - Determine the characteristic shape of the region.
 - Assumption: shape can be described by “local affine frame”.
- Solution: iterative approach
 - Use a circular window to compute second moment matrix.
 - Compute eigenvectors to adapt the circle to an ellipse.
 - Recompute second moment matrix using new window and iterate...



Iterative Affine Adaptation



1. Detect keypoints, e.g. multi-scale Harris
2. Automatically select the scales
3. Adapt affine shape based on second order moment matrix
4. Refine point location

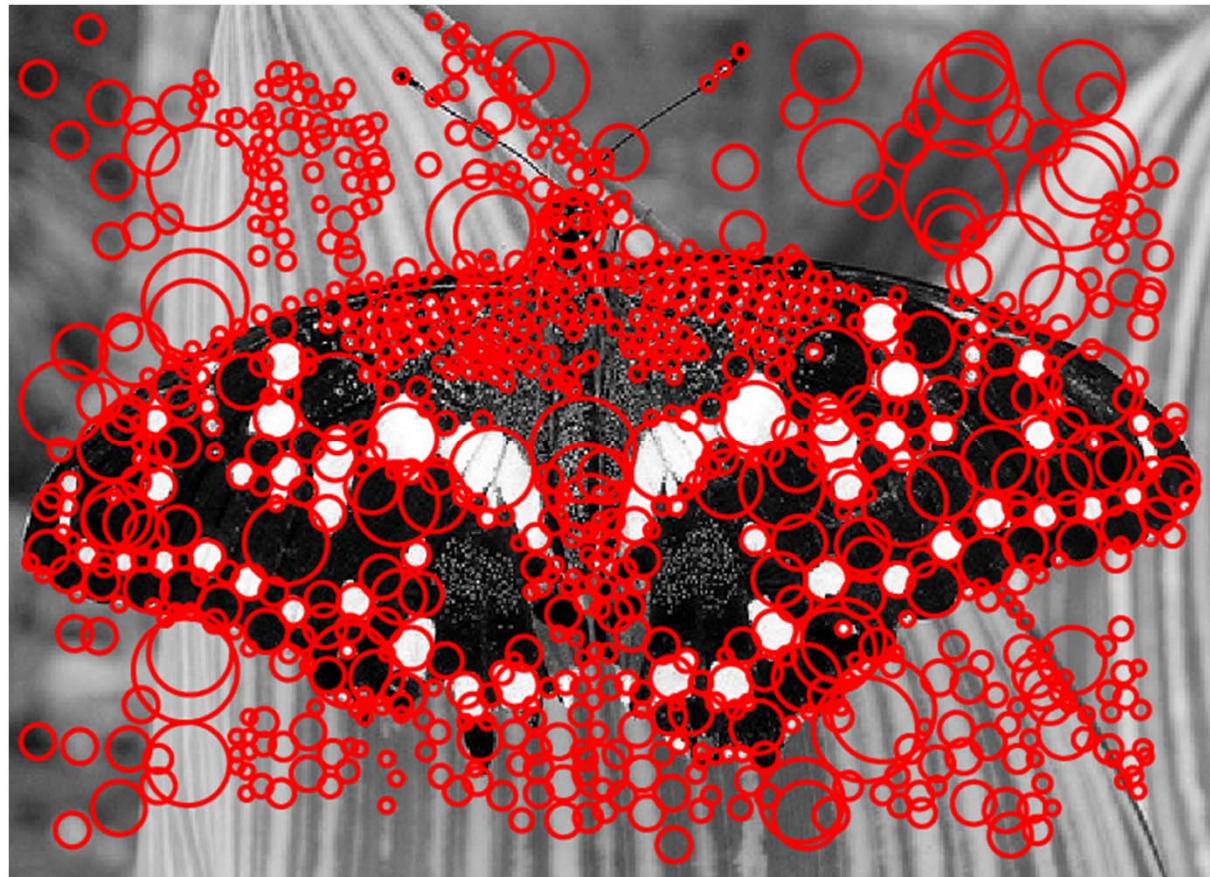
K. Mikolajczyk and C. Schmid, [Scale and affine invariant interest point detectors](#), IJCV 60(1):63-86, 2004.

Affine Normalization/Deskewing



- Steps
 - Rotate the ellipse's main axis to horizontal
 - Scale the x axis, such that it forms a circle

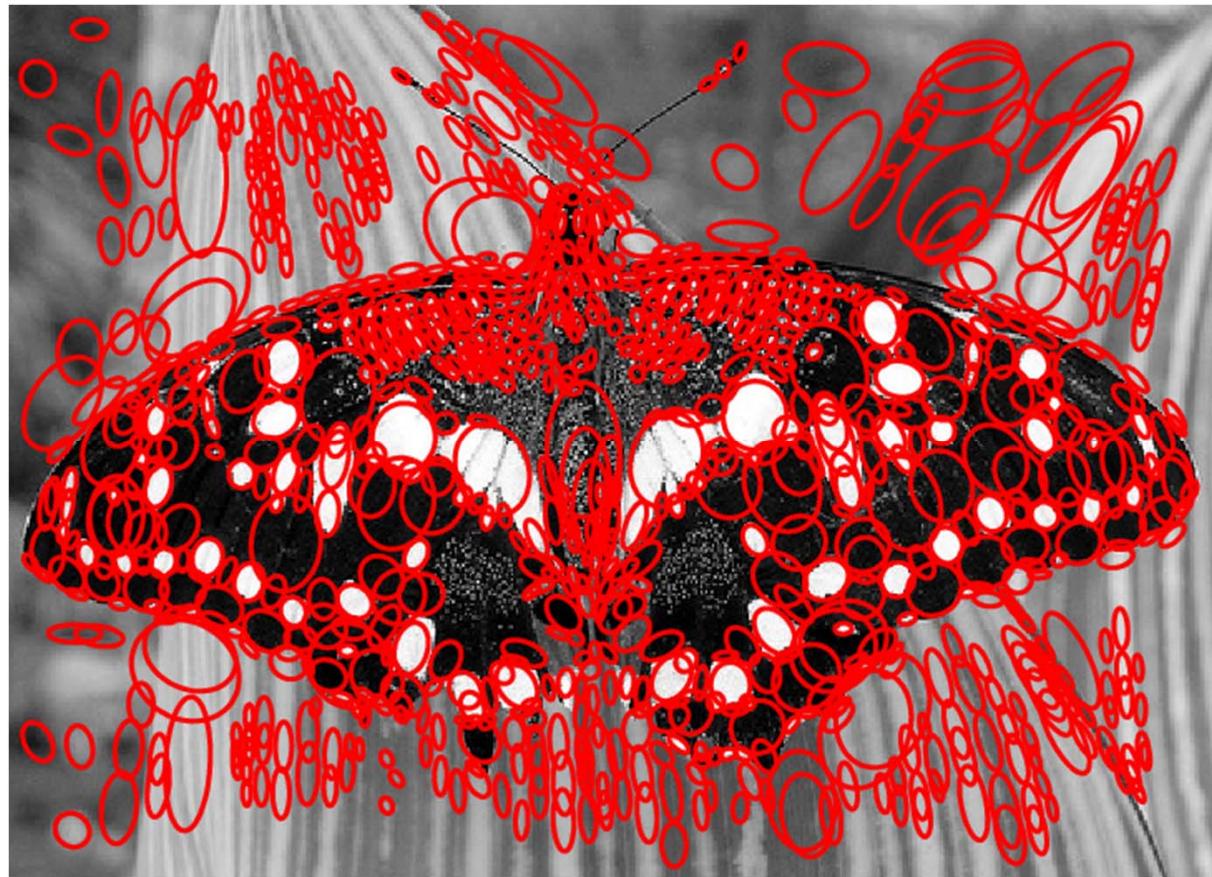
Affine Adaptation Example



Scale-invariant regions (blobs)

Slide credit: Svetlana Lazebnik

Affine Adaptation Example



Affine-adapted blobs

Slide credit: Svetlana Lazebnik

Summary: Affine-Inv. Feature Extraction

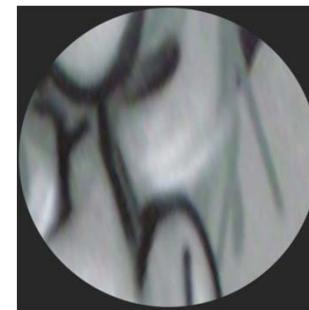
Extract affine regions



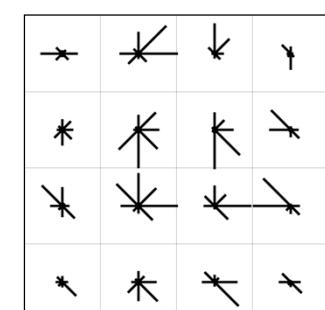
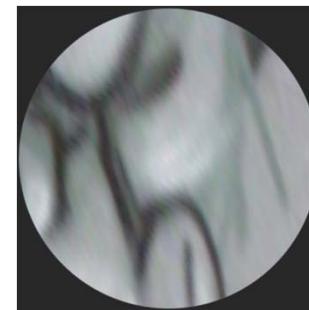
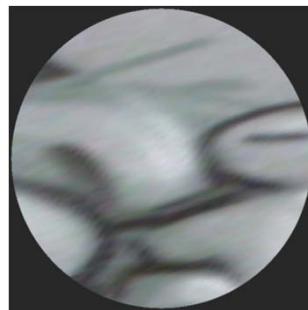
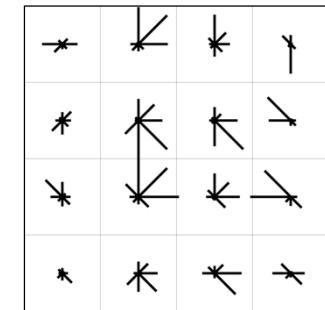
Normalize regions



Eliminate rotational ambiguity



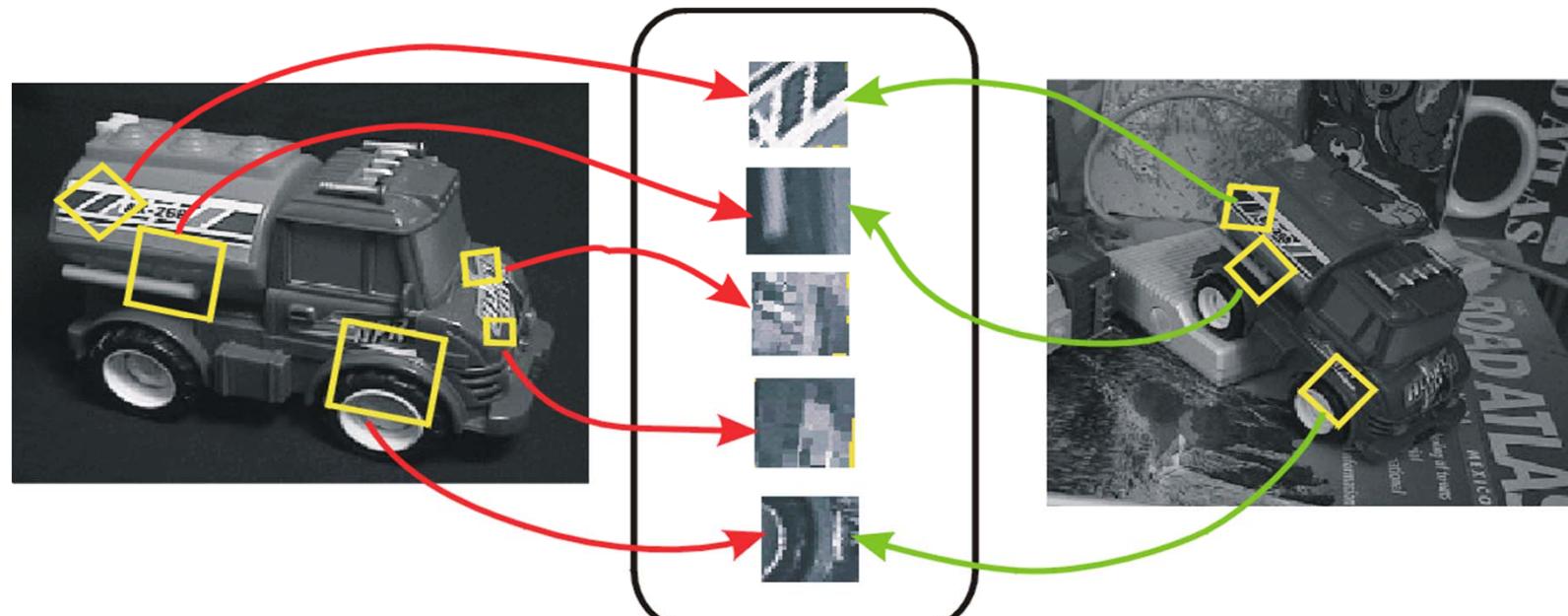
Compare descriptors



Slide credit: Svetlana Lazebnik

Invariance vs. Covariance

- Invariance:
 - $\text{features}(\text{transform}(\text{image})) = \text{features}(\text{image})$
- Covariance:
 - $\text{features}(\text{transform}(\text{image})) = \text{transform}(\text{features}(\text{image}))$

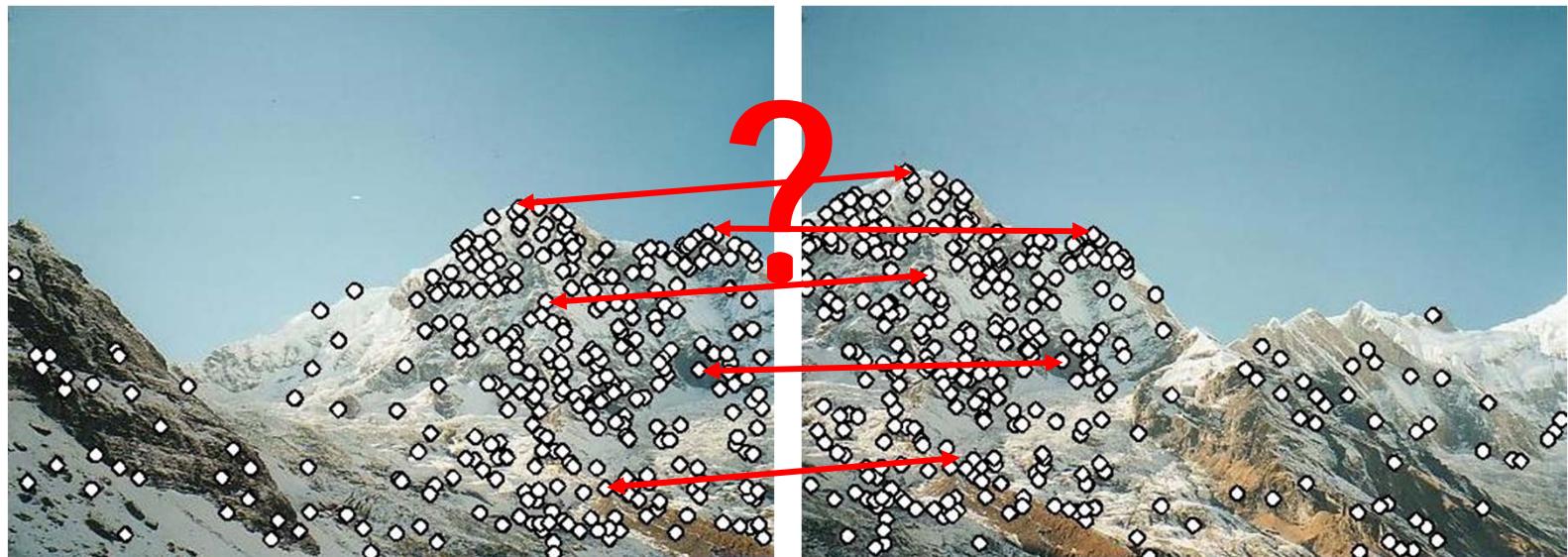


Covariant detection \Rightarrow invariant description

Local Descriptors

- We know how to detect points
- Next question:

How to describe them for matching?



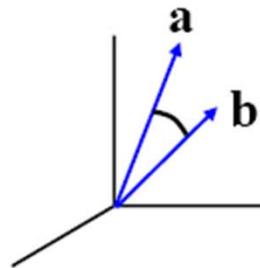
Point descriptor should be:
1. Invariant
2. Distinctive

Local Descriptors

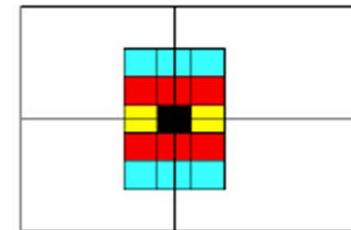
- Simplest descriptor: list of intensities within a patch.
- What is this going to be invariant to?

Write regions as vectors

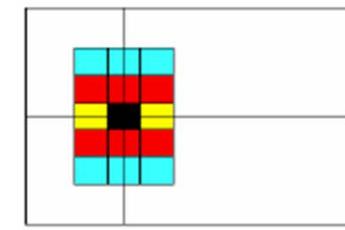
$$A \rightarrow \mathbf{a}, \quad B \rightarrow \mathbf{b}$$



region A



region B

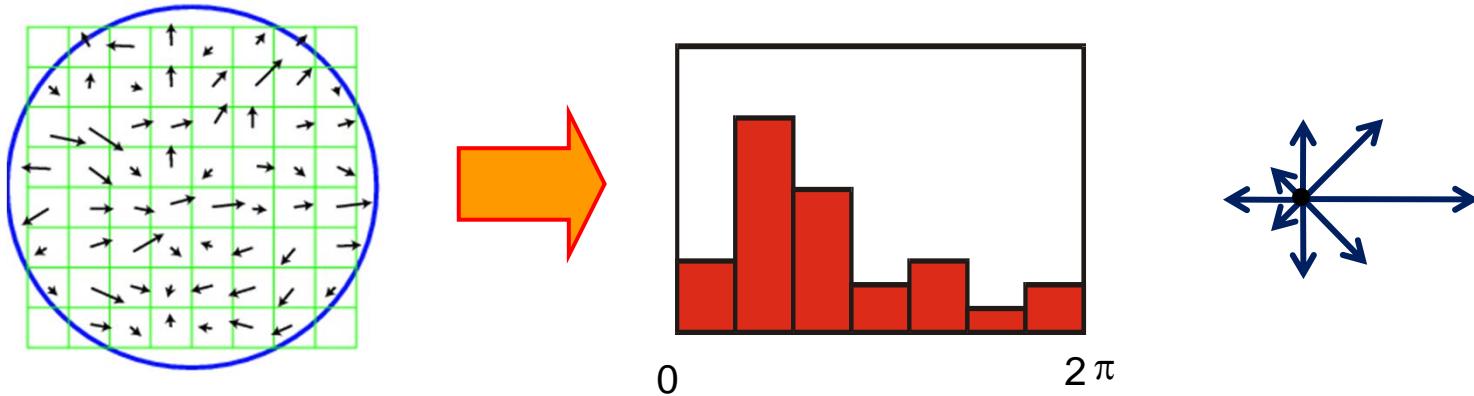


Feature Descriptors

- Disadvantage of patches as descriptors:
 - Small shifts can affect matching score a lot

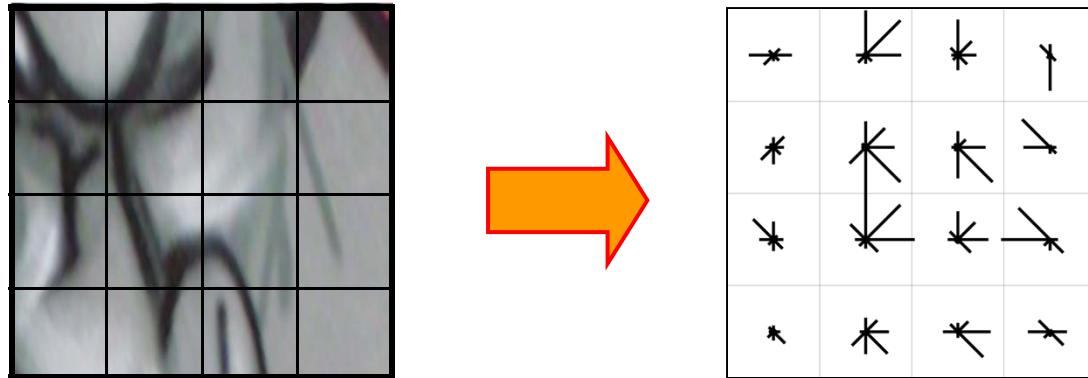


- Solution: histograms



Feature Descriptors: SIFT

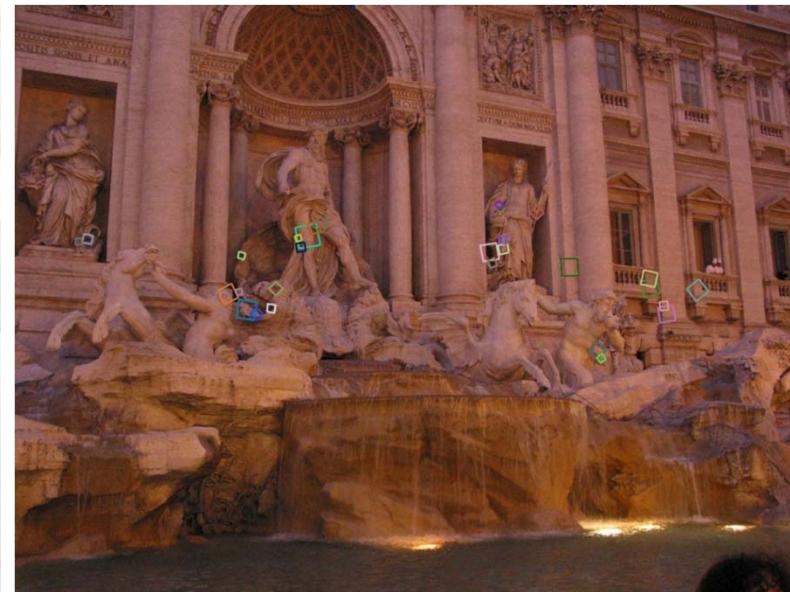
- Scale Invariant Feature Transform
- Descriptor computation:
 - Divide patch into 4×4 sub-patches: 16 cells
 - Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
 - Resulting descriptor: $4 \times 4 \times 8 = 128$ dimensions



David G. Lowe. "[Distinctive image features from scale-invariant keypoints.](#)" IJCV 60 (2), pp. 91-110, 2004.

Overview: SIFT

- Extraordinarily robust matching technique
 - Can handle changes in viewpoint up to ~60 deg. out-of-plane rotation
 - Can handle significant changes in illumination
 - Sometimes even day vs. night (below)
 - Fast and efficient—can run in real time
 - Lots of code available
 - http://people.csail.mit.edu/albert/ladypack/wiki/index.php/Known_implementations_of_SIFT



Slide credit: Steve Seitz

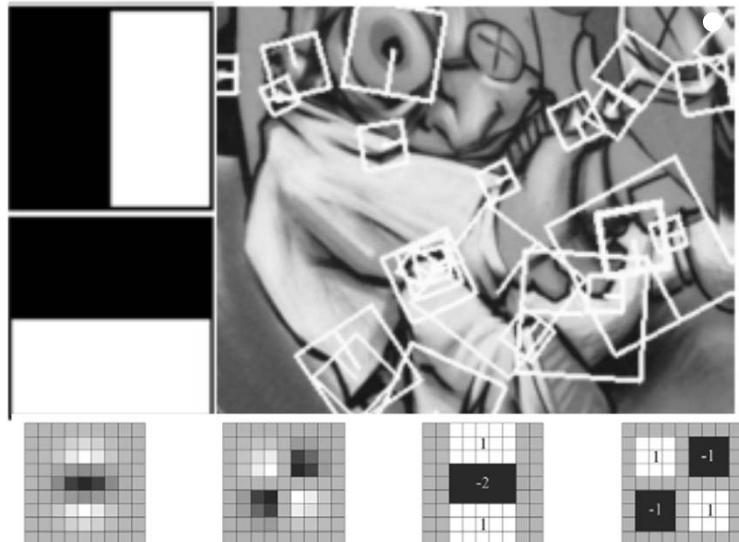
Working with SIFT Descriptors

- One image yields:
 - n 128-dimensional descriptors: each one is a histogram of the gradient orientations within a patch
 - $[n \times 128$ matrix]
 - n scale parameters specifying the size of each patch
 - $[n \times 1$ vector]
 - n orientation parameters specifying the angle of the patch
 - $[n \times 1$ vector]
 - n 2D points giving positions of the patches
 - $[n \times 2$ matrix]



Slide credit: Steve Seitz

Local Descriptors: SURF



- Fast approximation of SIFT idea

Efficient computation by 2D box filters & integral images
⇒ 6 times faster than SIFT

Equivalent quality for object identification

<http://www.vision.ee.ethz.ch/~surf>

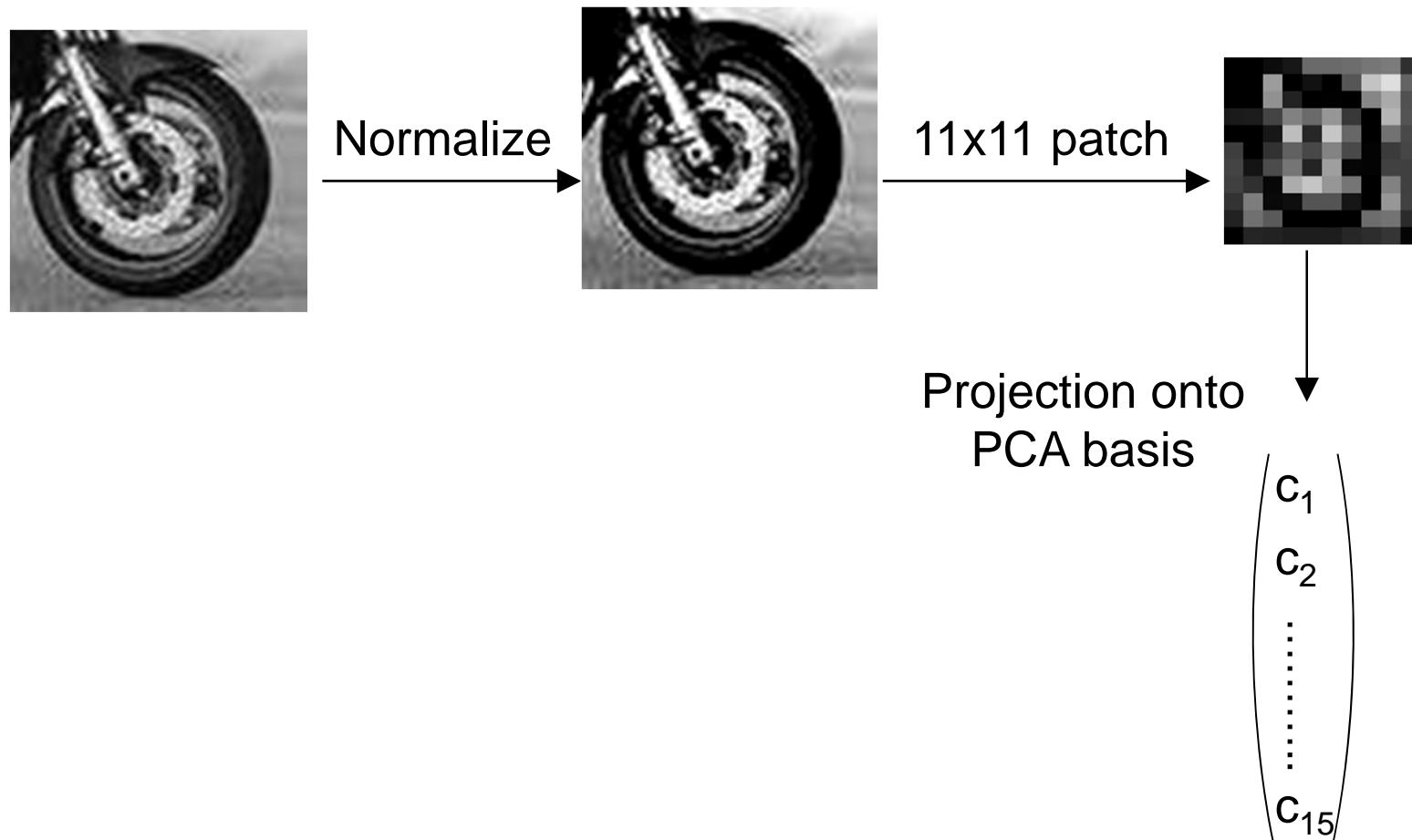
GPU implementation available

Feature extraction @ 100Hz
(detector + descriptor, 640×480 img)

<http://homes.esat.kuleuven.be/~ncorneli/gpusurf/>

[Bay, ECCV'06], [Cornelis, CVGPU'08]

Other local descriptors: Gray-scale intensity



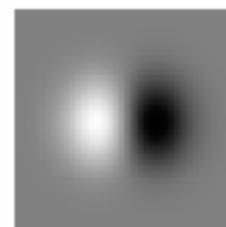
Other local descriptors: Steerable filters

$$R_1^{0^\circ} = G_1^0 * I$$

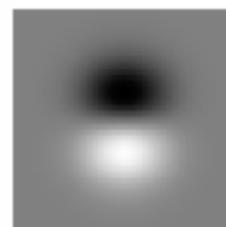
$$R_1^{90^\circ} = G_1^{90} * I$$

then

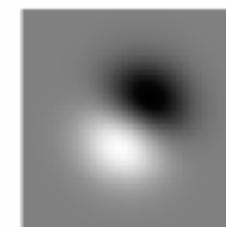
$$R_1^\theta = \cos(\theta)R_1^{0^\circ} + \sin(\theta)R_1^{90^\circ}$$



a



b



c



d



e



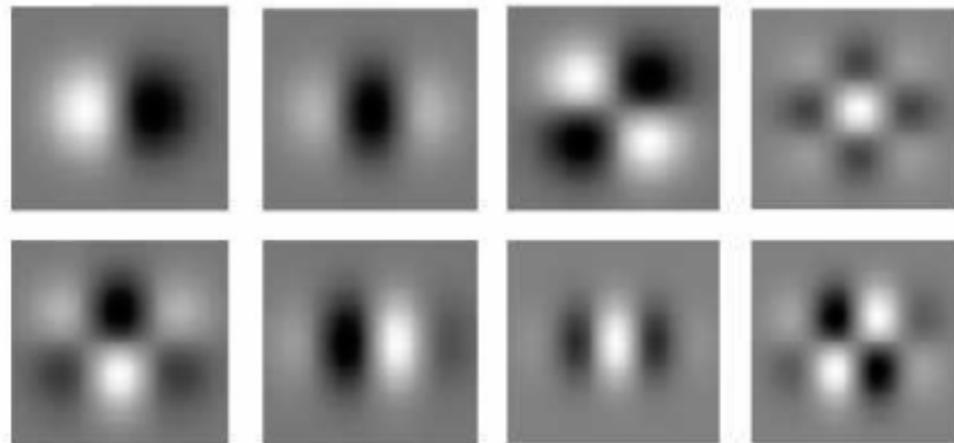
f



g

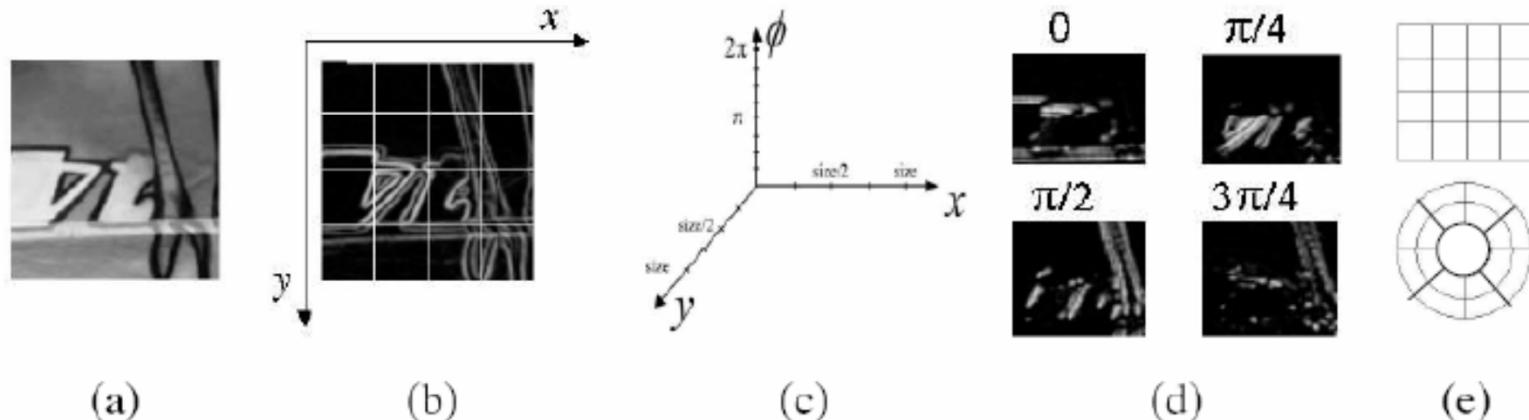
Other local descriptors: Gray-scale intensity

Gaussian derivatives up to 4th order. The remaining derivatives can be computed by rotation of 90 degrees.



Other local descriptors: GLOH

- GLOH: Gradient Location and Orientation Histogram(Miko04)
 - Very similar to SIFT.
 - Log-polar location grid:
 - 3 bins in radial direction;
 - 8 bins in angular direction
 - Gradient orientation quantized in 16 bins.
 - Total: $(2 \times 8 + 1) \times 16 = 272$ bins \rightarrow PCA dimension reduction.



Other local descriptors: Shape context

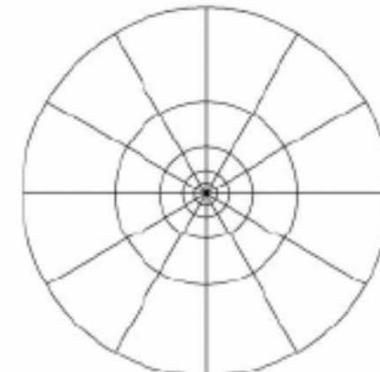
[Belongie et al. 2002]



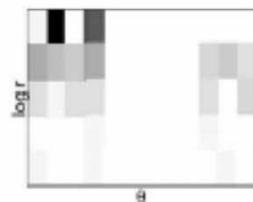
(a)



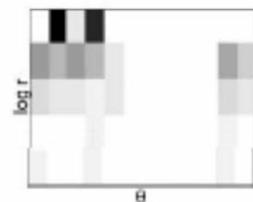
(b)



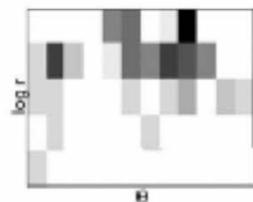
(c)



(d)



(e)



(f)

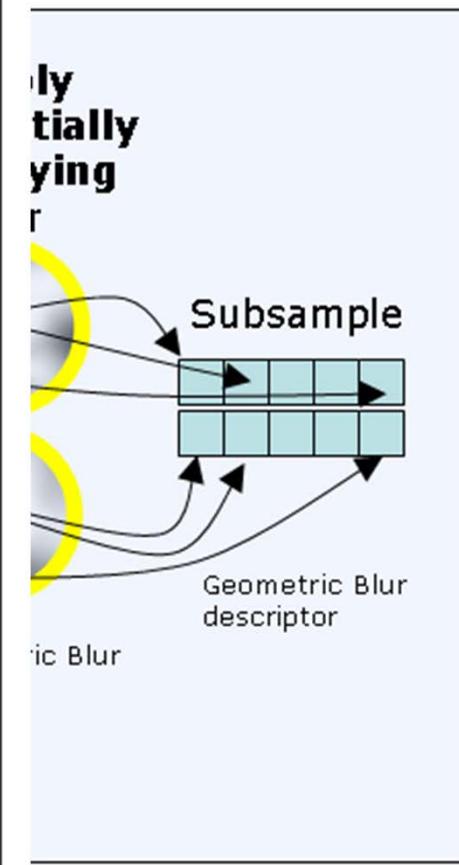
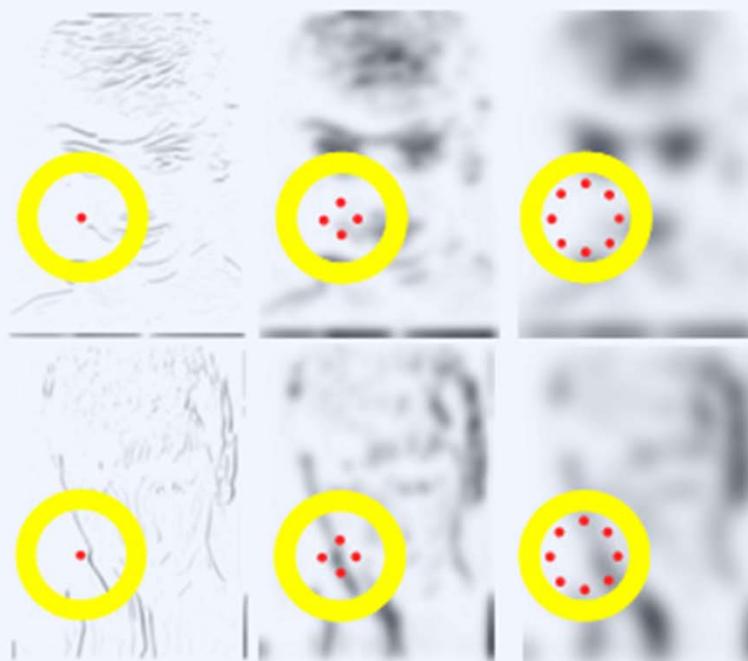


(g)

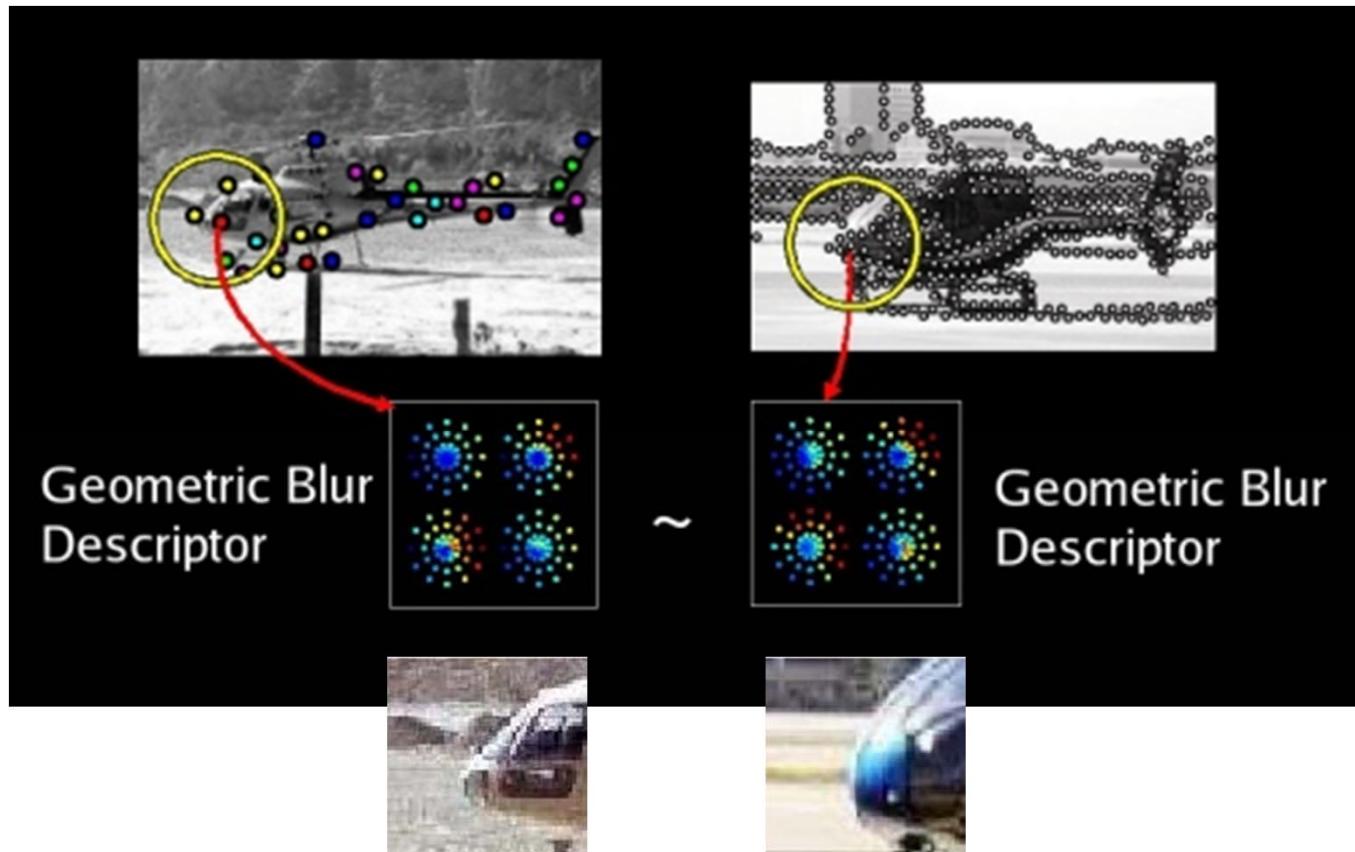
Other local descriptors: Geometric Blur

[Berg et al. 2001]

In practice compute discrete blur levels for whole image and sample as needed for each feature location.



Other local descriptors: Geometric Blur



Affine Covariant Features



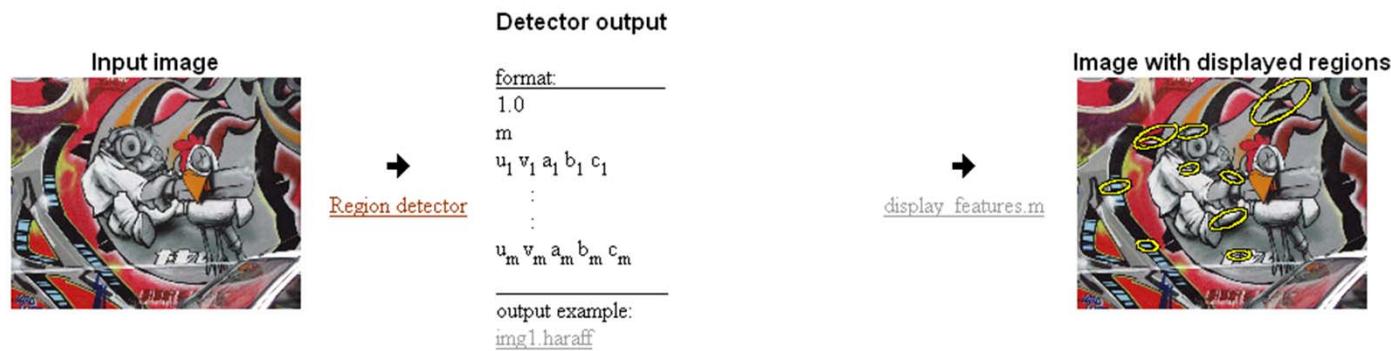
KATHOLIEKE UNIVERSITEIT
LEUVEN

 INRIA
RHÔNE ALPES



Collaborative work between the Visual Geometry Group, Katholieke Universiteit Leuven, Inria Rhône-Alpes and the Center for Machine Perception.

Affine Covariant Region Detectors



Parameters defining an affine region

u, v, a, b, c in $a(x-u)(x-u) + 2b(x-u)(y-v) + c(y-v)(y-v) = 1$
with $(0,0)$ at image top left corner

Code

- provided by the authors, see [publications](#) for details and links to authors web sites.

Linux binaries

[Harris-Affine & Hessian-Affine](#)

[MSER](#) - Maximally stable extremal regions (also Windows)

[IBR](#) - Intensity extrema based detector

[EBR](#) - Edge based detector

[Salient](#) region detector

Example of use

prompt>./h_affine.ln -haraff -i [img1.ppm](#) -o img1.haraff -thres 1000 matlab>> [d](#)

prompt>./h_affine.ln -hesaff -i [img1.ppm](#) -o img1.hesaff -thres 500 matlab>> [d](#)

prompt>./mser.ln -t 2 -es 2 -i [img1.ppm](#) -o img1.mser matlab>> [d](#)

prompt>./ibr.ln [img1.ppm](#) img1.ibr -scalefactor 1.0 matlab>> [d](#)

prompt> ./ebr.ln [img1.ppm](#) img1.ebr matlab>> [d](#)

prompt>./salient.ln [img1.ppm](#) img1.sal matlab>> [d](#)

Displaying results

<http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries>

Value of Local Features

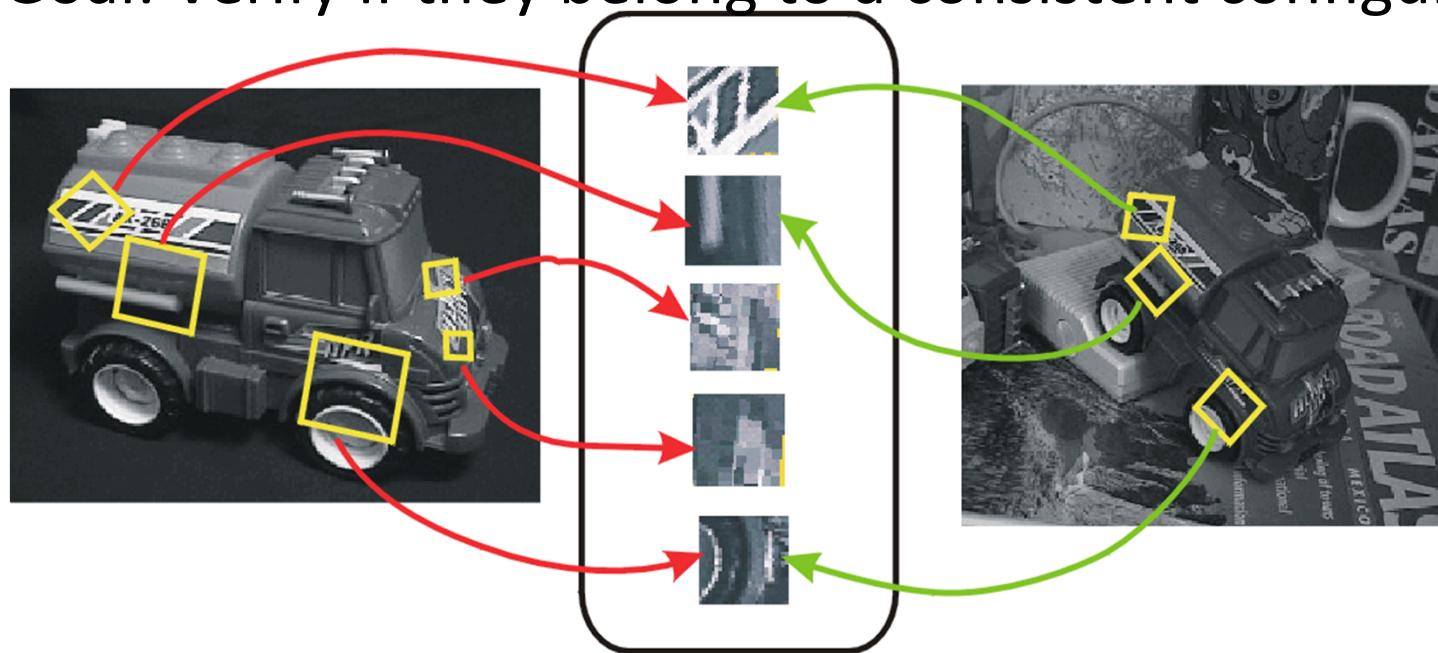
- Advantages
 - Critical to find distinctive and repeatable local regions for multi-view matching.
 - Complexity reduction via selection of distinctive points.
 - Describe images, objects, parts without requiring segmentation; robustness to clutter & occlusion.
 - Robustness: similar descriptors in spite of moderate view changes, noise, blur, etc.

What we will learn today

- Local descriptors
 - SIFT
 - An assortment of other descriptors
- Recognition and matching with local features
(Problem Set 3 (Q3))
 - Matching objects with local features: David Lowe
 - Panorama stitching: Brown & Lowe
 - Applications

Recognition with Local Features

- Image content is transformed into local features that are invariant to translation, rotation, and scale
- Goal: Verify if they belong to a consistent configuration



Local Features,
e.g. SIFT

Slide credit: David Lowe

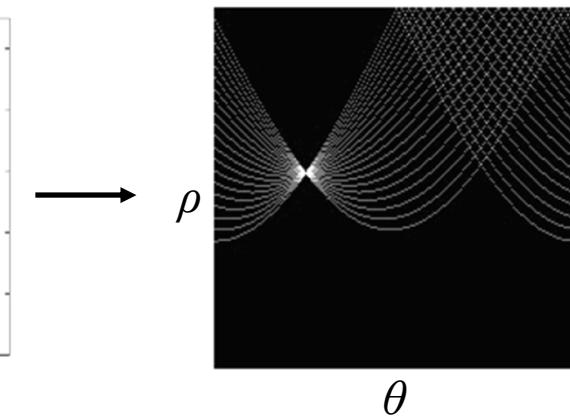
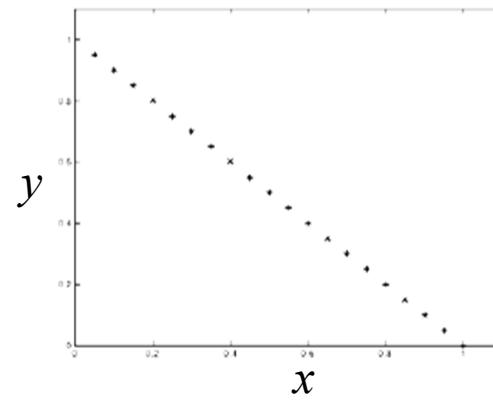
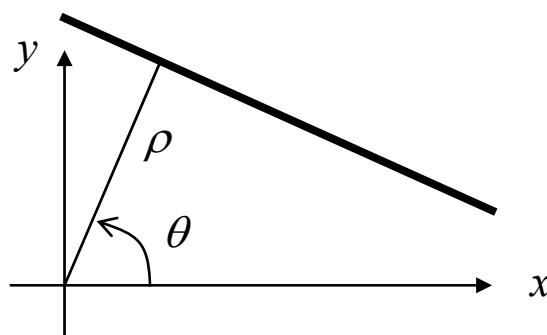
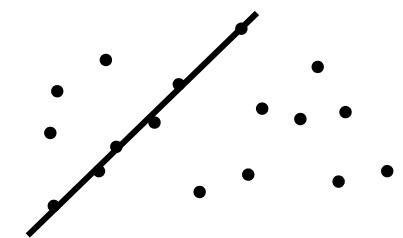
Finding Consistent Configurations

- Global spatial models
 - Generalized Hough Transform [Lowe99]
 - RANSAC [Obdrzalek02, Chum05, Nister06]
 - Basic assumption: object is planar
- Assumption is often justified in practice
 - Valid for many structures on buildings
 - Sufficient for small viewpoint variations on 3D objects



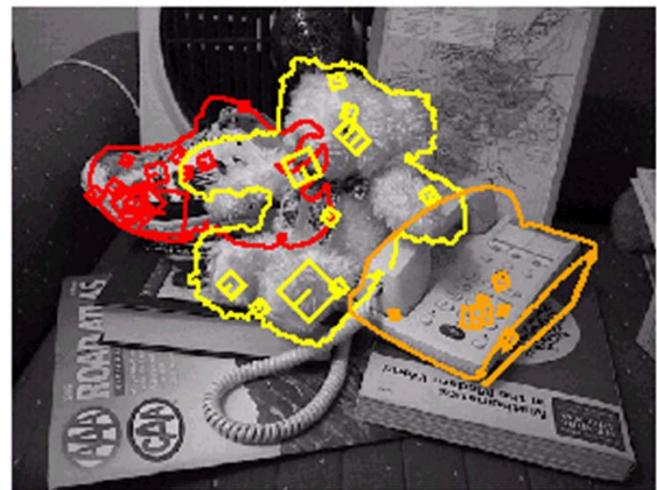
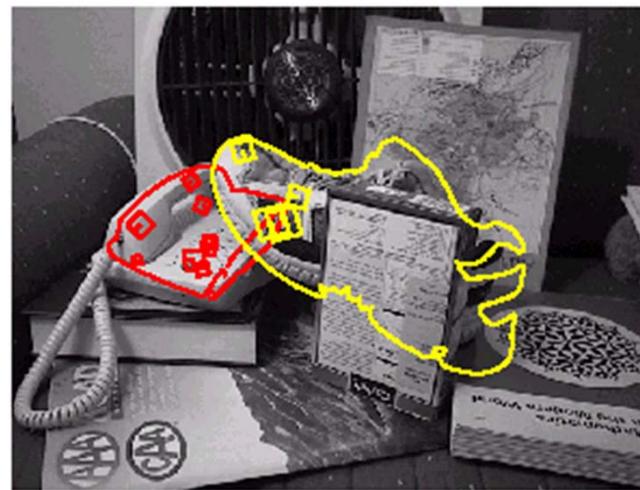
Reminder: Hough Transform

- Origin: Detection of straight lines in clutter
 - Basic idea: each candidate point votes for all lines that it is consistent with.
 - Votes are accumulated in quantized array
 - Local maxima correspond to candidate lines
- Representation of a line
 - Usual form $y = a x + b$ has a singularity around 90° .
 - Better parameterization: $x \cos(\theta) + y \sin(\theta) = \rho$



3D Object Recognition

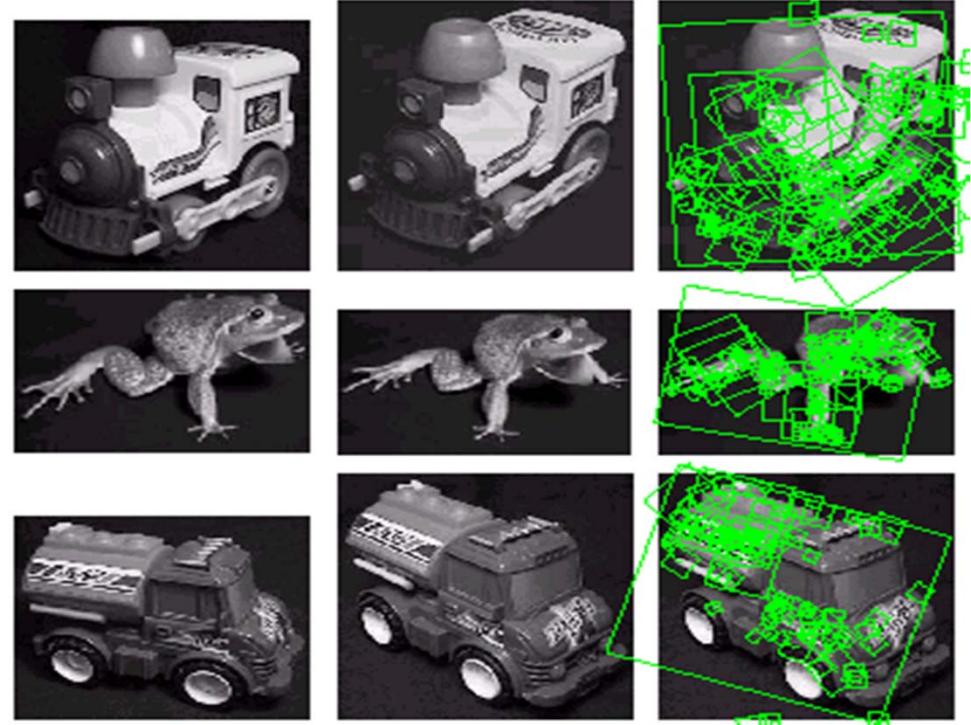
- Gen. Hough T. for Recognition
 - Typically only 3 feature matches needed for recognition
 - Extra matches provide robustness
 - Affine model can be used for planar objects



View Interpolation

- **Training**

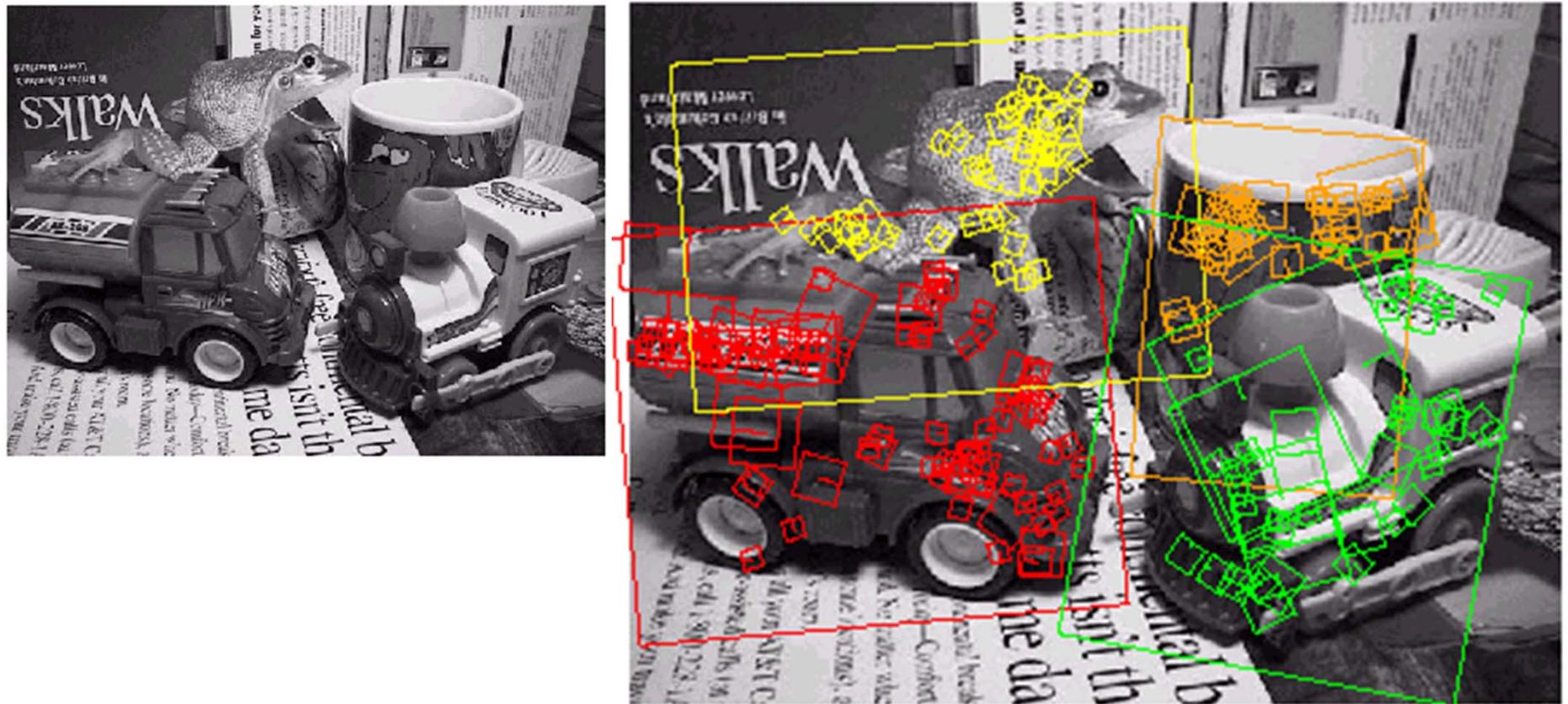
- Training views from similar viewpoints are clustered based on feature matches.
- Matching features between adjacent views are linked.



- **Recognition**

- Feature matches may be spread over several training viewpoints.
⇒ Use the known links to “transfer votes” to other viewpoints.

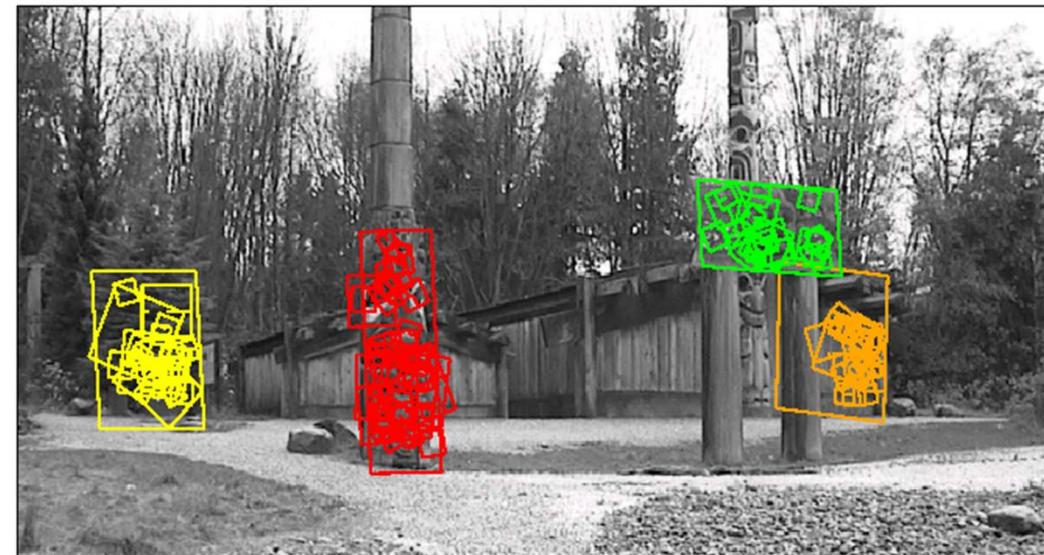
Recognition Using View Interpolation



Location Recognition



Training



What we will learn today

- Local descriptors
 - SIFT
 - An assortment of other descriptors
- Recognition and matching with local features
(Problem Set 3 (Q3))
 - Matching objects with local features: David Lowe
 - Panorama stitching: Brown & Lowe
 - Applications

Introduction

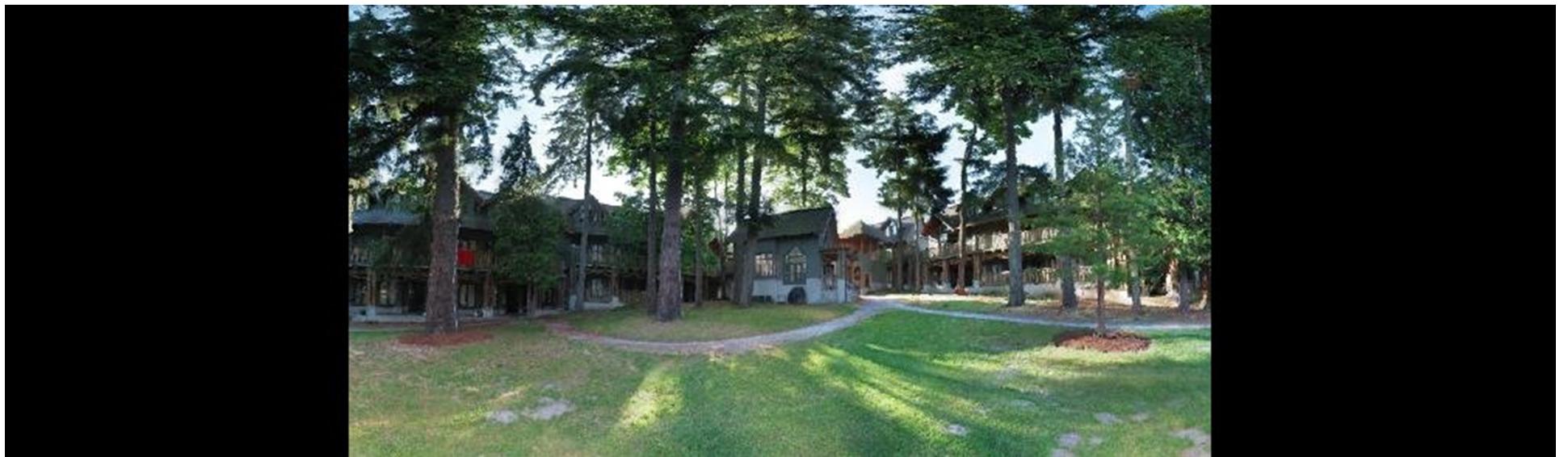
- Are you getting the whole picture?
 - Compact Camera FOV = $50 \times 35^\circ$



Slide credit: Matthew Brown

Introduction

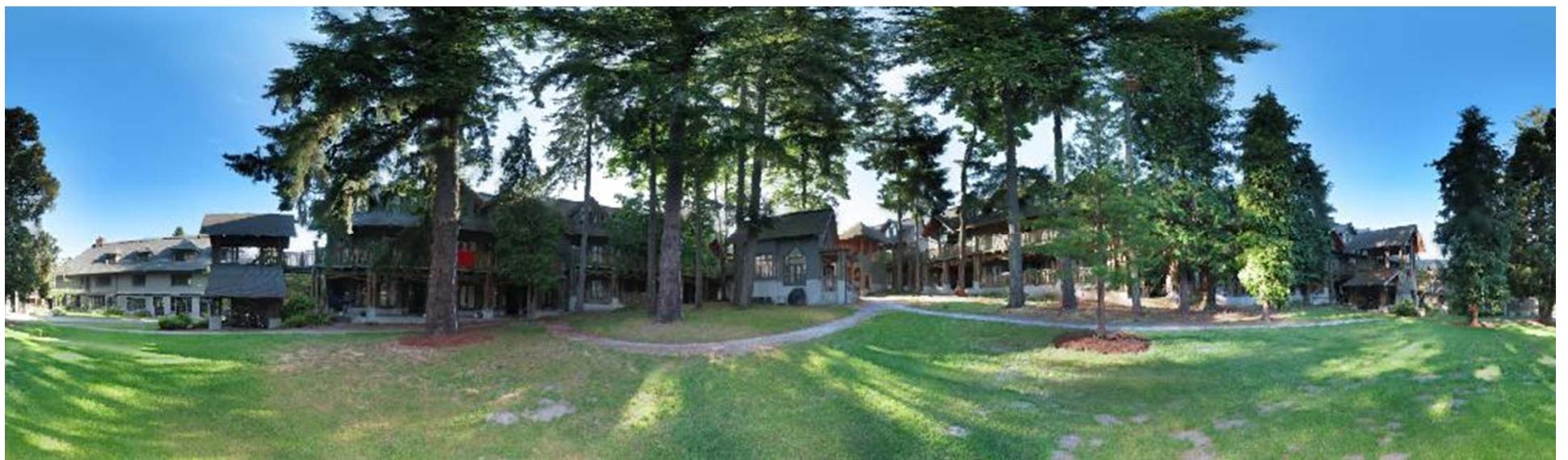
- Are you getting the whole picture?
 - Compact Camera FOV = $50 \times 35^\circ$
 - Human FOV = $200 \times 135^\circ$



Slide credit: Matthew Brown

Introduction

- Are you getting the whole picture?
 - Compact Camera FOV = $50 \times 35^\circ$
 - Human FOV = $200 \times 135^\circ$



Slide credit: Matthew Brown

Why “Recognising Panoramas”?

- 1D Rotations (θ)
 - Ordering \Rightarrow matching images

Slide credit: Matthew Brown

Why “Recognising Panoramas”?

- 1D Rotations (θ)
 - Ordering \Rightarrow matching images



Slide credit: Matthew Brown

Why “Recognising Panoramas”?

- 1D Rotations (θ)
 - Ordering \Rightarrow matching images



Slide credit: Matthew Brown

Why “Recognising Panoramas”?

- 1D Rotations (θ)
 - Ordering \Rightarrow matching images



- 2D Rotations (θ, ϕ)
 - Ordering $\not\Rightarrow$ matching images

Slide credit: Matthew Brown

Why “Recognising Panoramas”?

- 1D Rotations (θ)
 - Ordering \Rightarrow matching images



- 2D Rotations (θ, ϕ)
 - Ordering $\not\Rightarrow$ matching images



Why “Recognising Panoramas”?

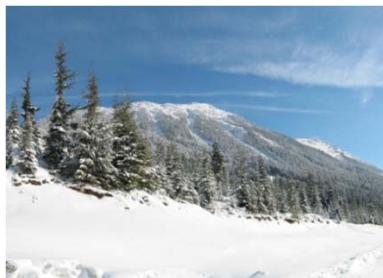
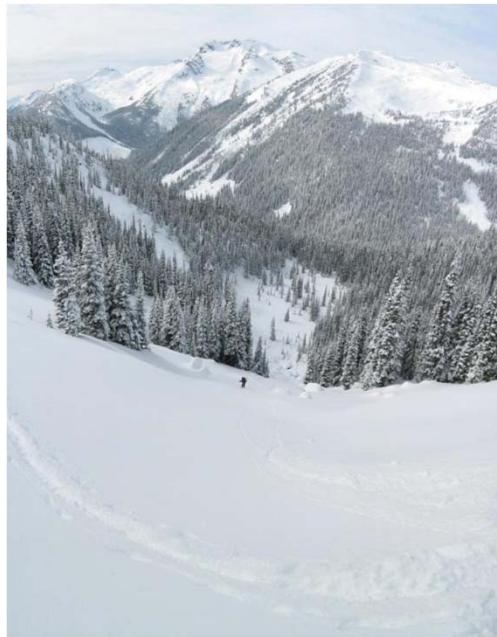
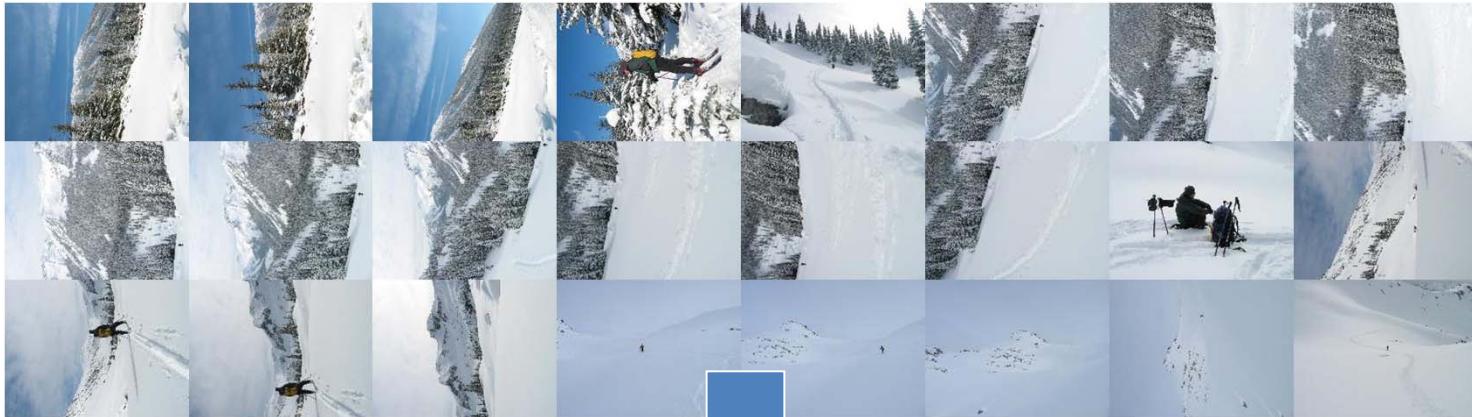
- 1D Rotations (θ)
 - Ordering \Rightarrow matching images



- 2D Rotations (θ, ϕ)
 - Ordering $\not\Rightarrow$ matching images



Why “Recognising Panoramas”?

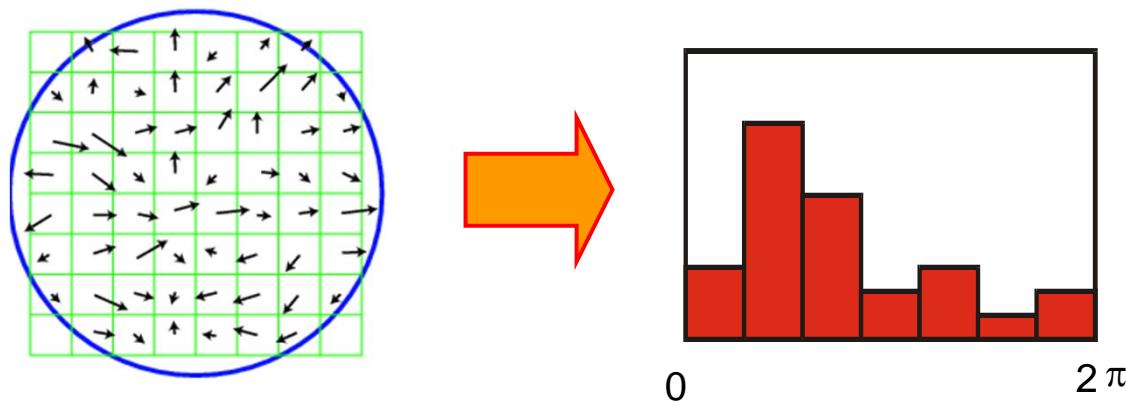


Slide credit: Matthew Brown

Algorithm (Brown & Lowe, 2003)

1. Feature Matching
 - a) SIFT Features
 - b) Nearest Neighbor Matching
2. Image Matching
3. Bundle Adjustment
4. Multi-band Blending

Nearest Neighbor Matching of SIFT Features



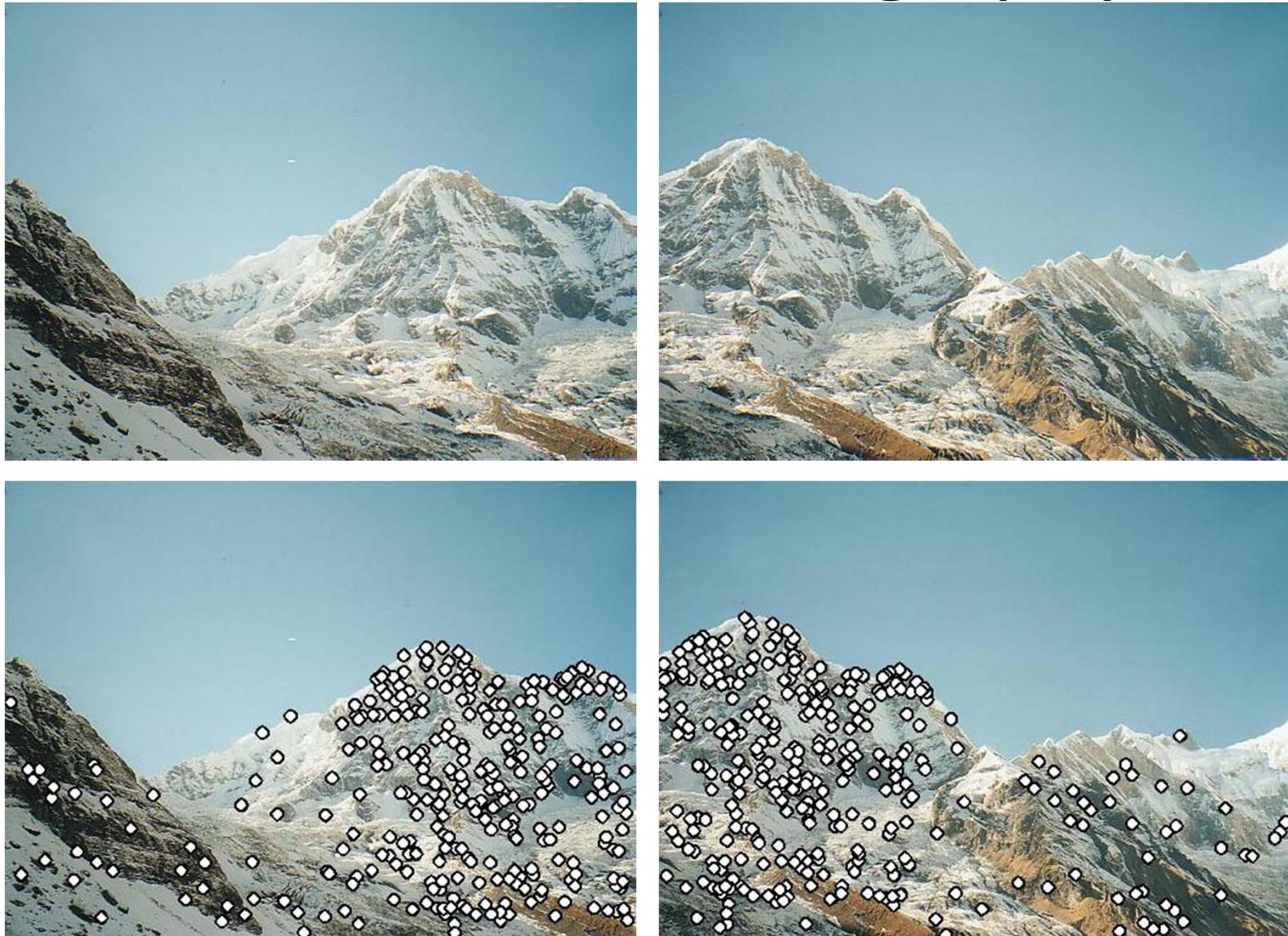
- Find k-NN for each feature
 - $k \approx$ number of overlapping images (we use $k = 4$)
- Use k-d tree
 - k-d tree recursively bi-partitions data at mean in the dimension of maximum variance
 - Approximate nearest neighbors found in $O(n \log n)$

Slide credit: Matthew Brown

Algorithm (Brown & Lowe, 2003)

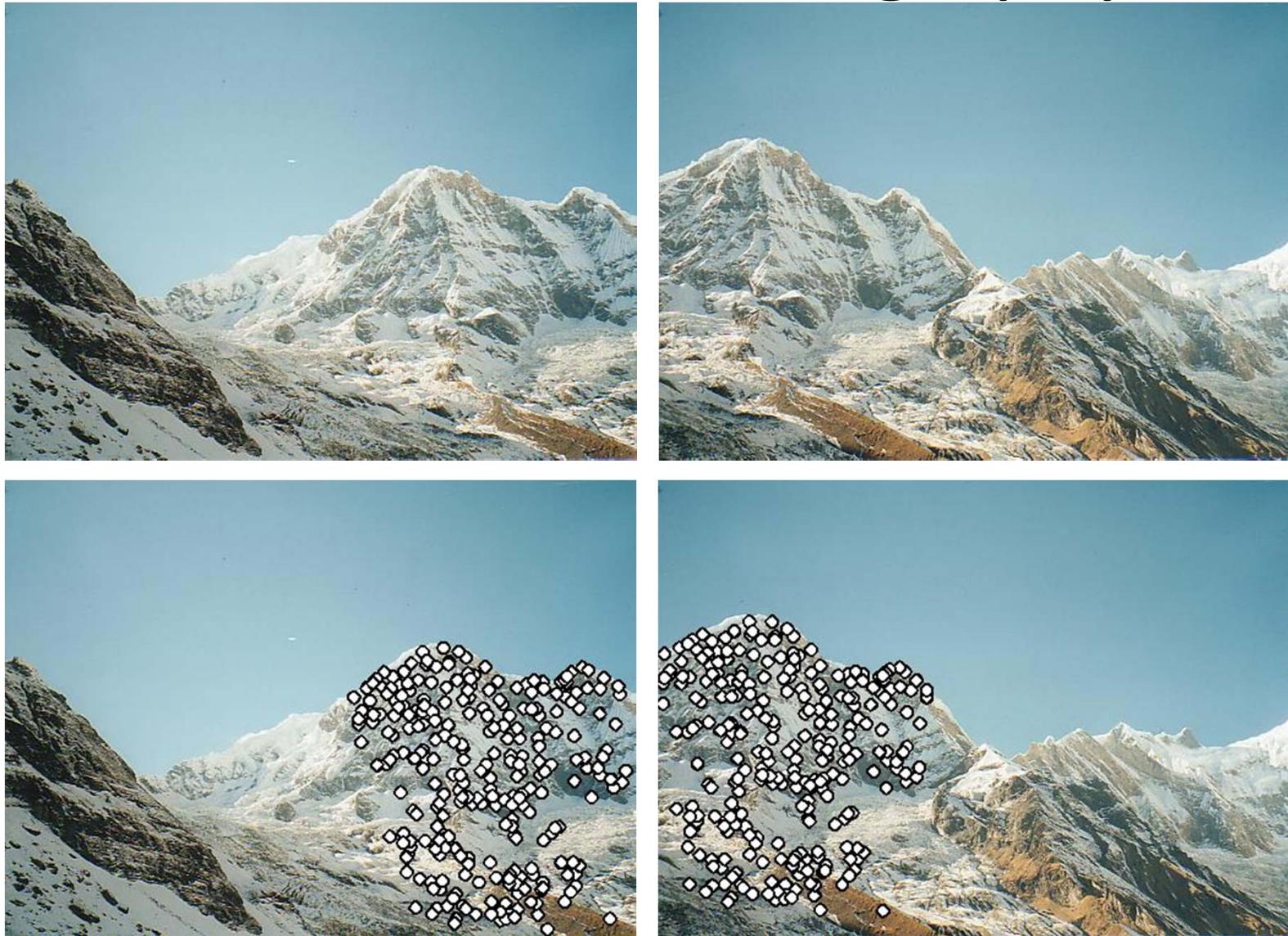
1. Feature Matching
2. Image Matching
 - a) RANSAC for Homography
 - b) Probabilistic model for verification
3. Bundle Adjustment
4. Multi-band Blending

RANSAC for Homography



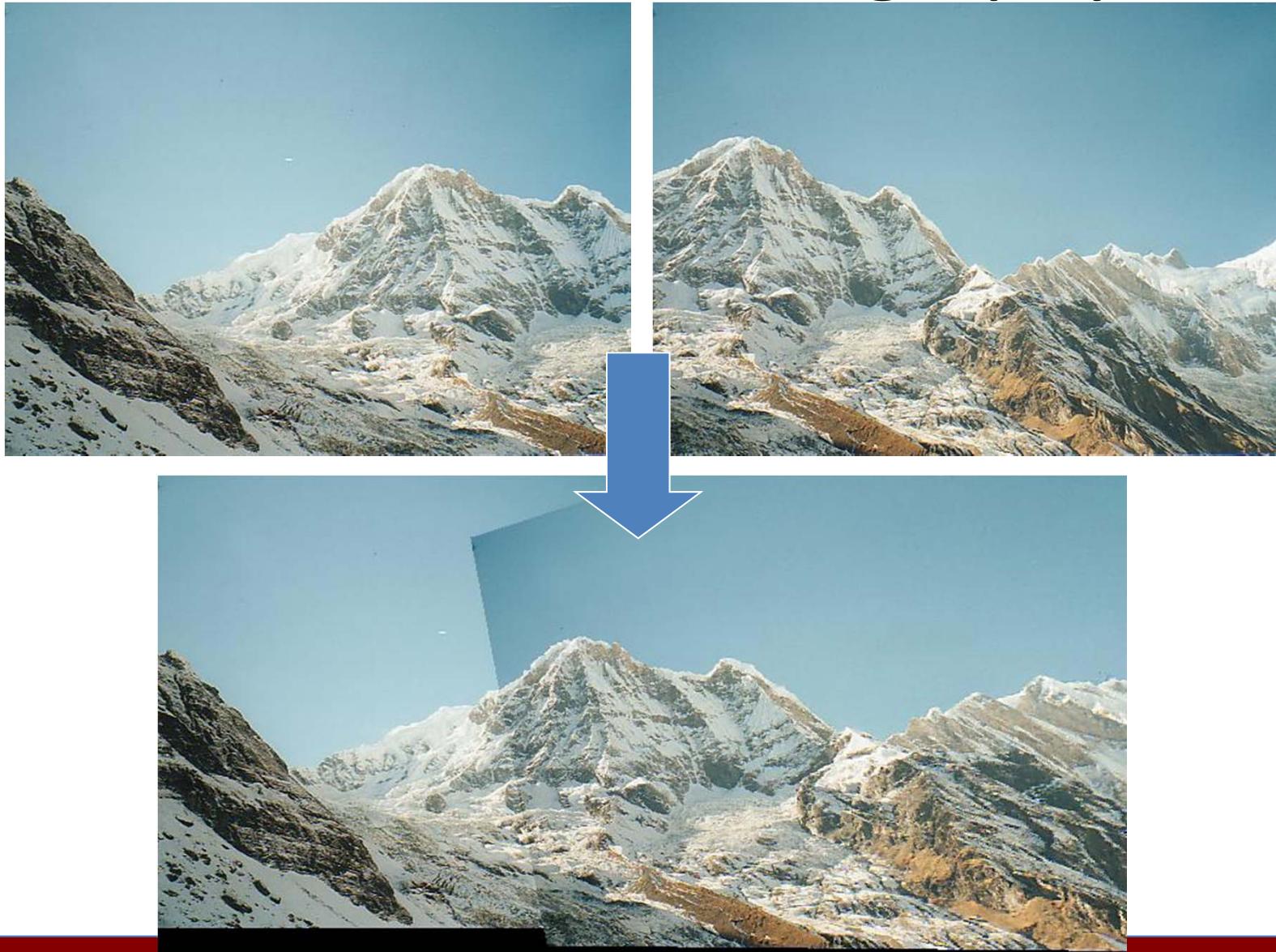
Slide credit: Matthew Brown

RANSAC for Homography



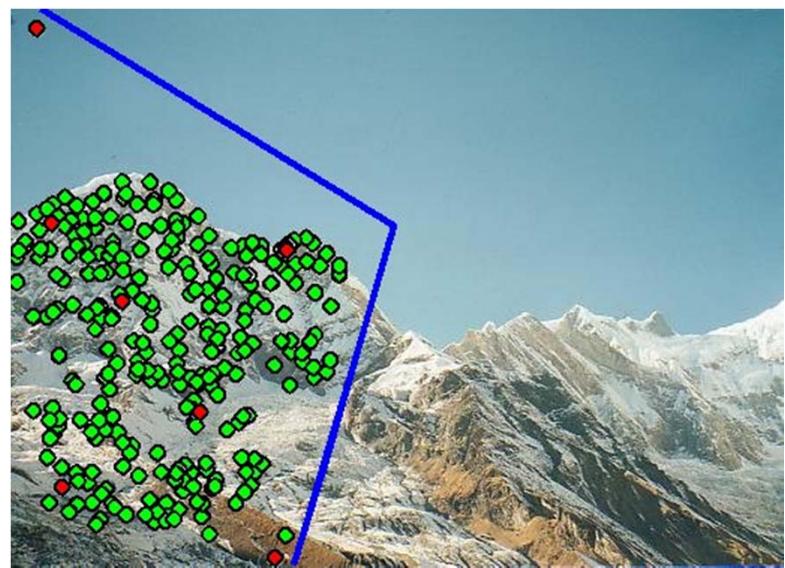
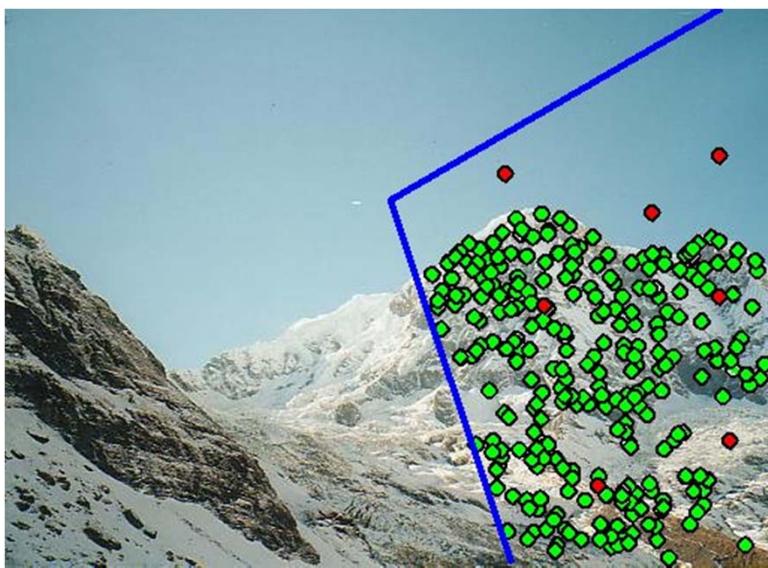
Slide credit: Matthew Brown

RANSAC for Homography



Slide credit: Matthew Brown

Probabilistic model for verification



Slide credit: Matthew Brown

Probabilistic model for verification

- Compare probability that this set of RANSAC inliers/outliers was generated by a correct/false image match

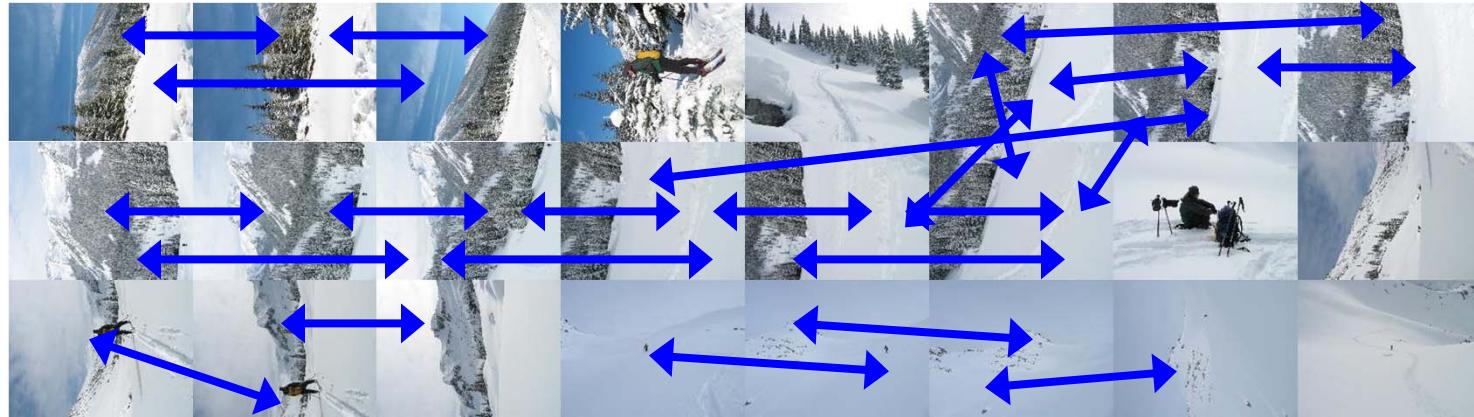
$$\frac{B(n_i; n_f, p_1)}{B(n_i; n_f, p_0)} \begin{matrix} \text{accept} \\ \gtrless \\ \text{reject} \end{matrix} \frac{1}{\frac{1}{p_{min}} - 1}$$

- n_i = #inliers, n_f = #features
 - p_1 = $p(\text{inlier} \mid \text{match})$, p_0 = $p(\text{inlier} \mid \sim\text{match})$
 - p_{min} = acceptance probability

- Choosing values for p_1 , p_0 and p_{min}

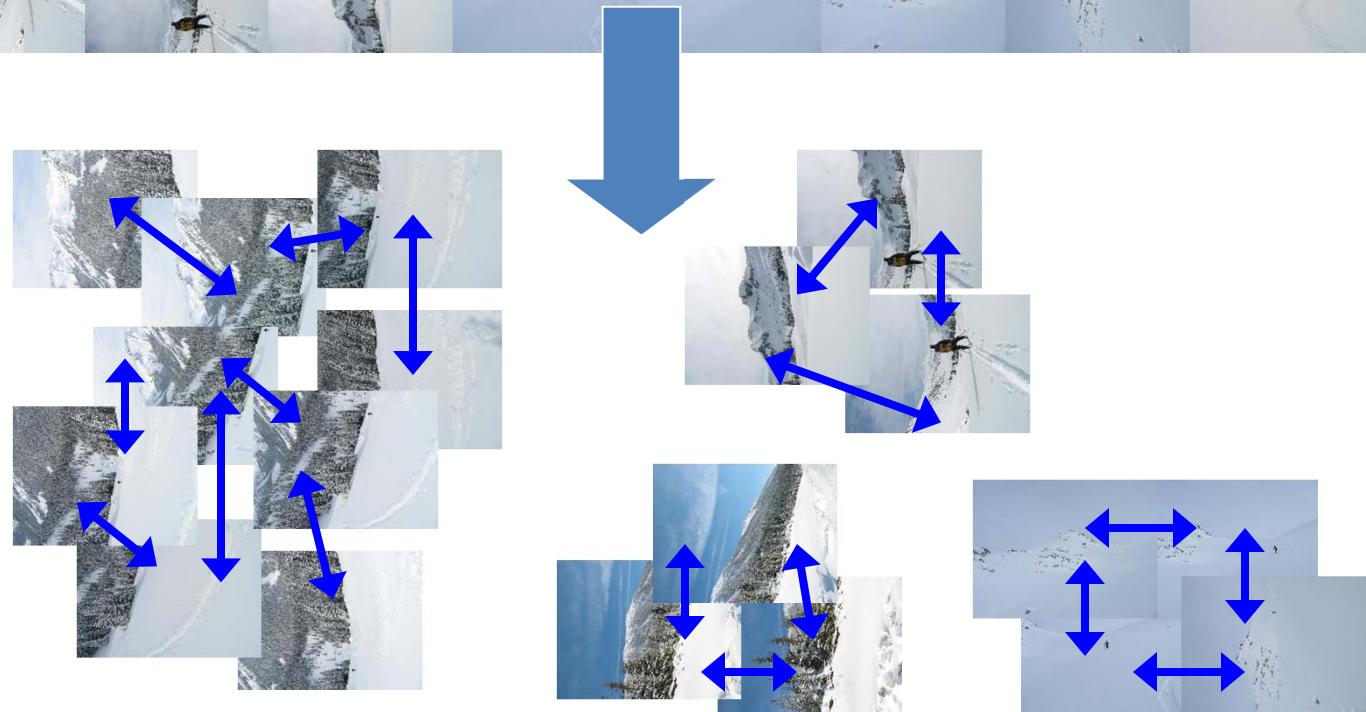
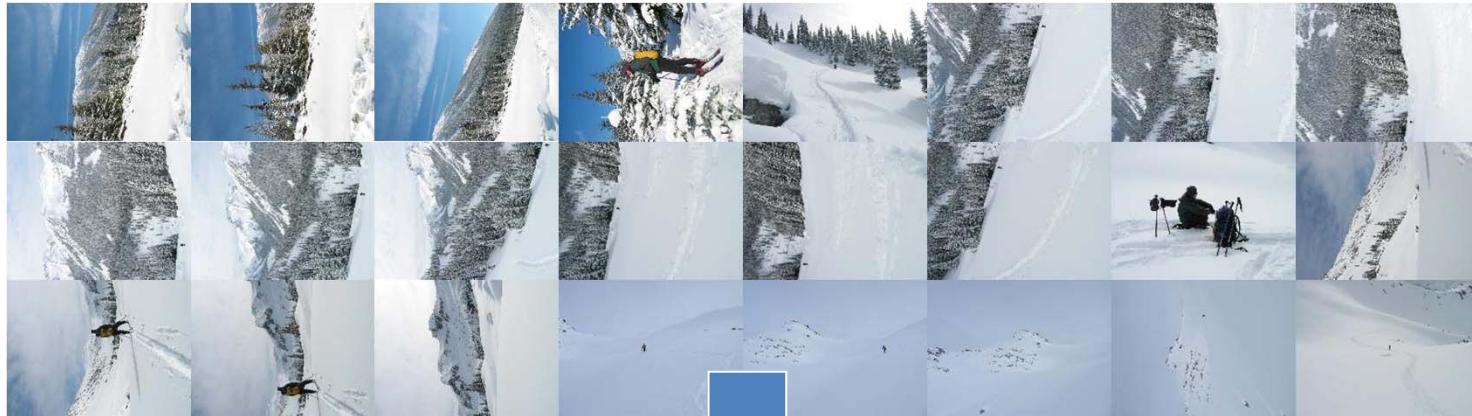
$$n_i > 5.9 + 0.22n_f$$

Finding the panoramas



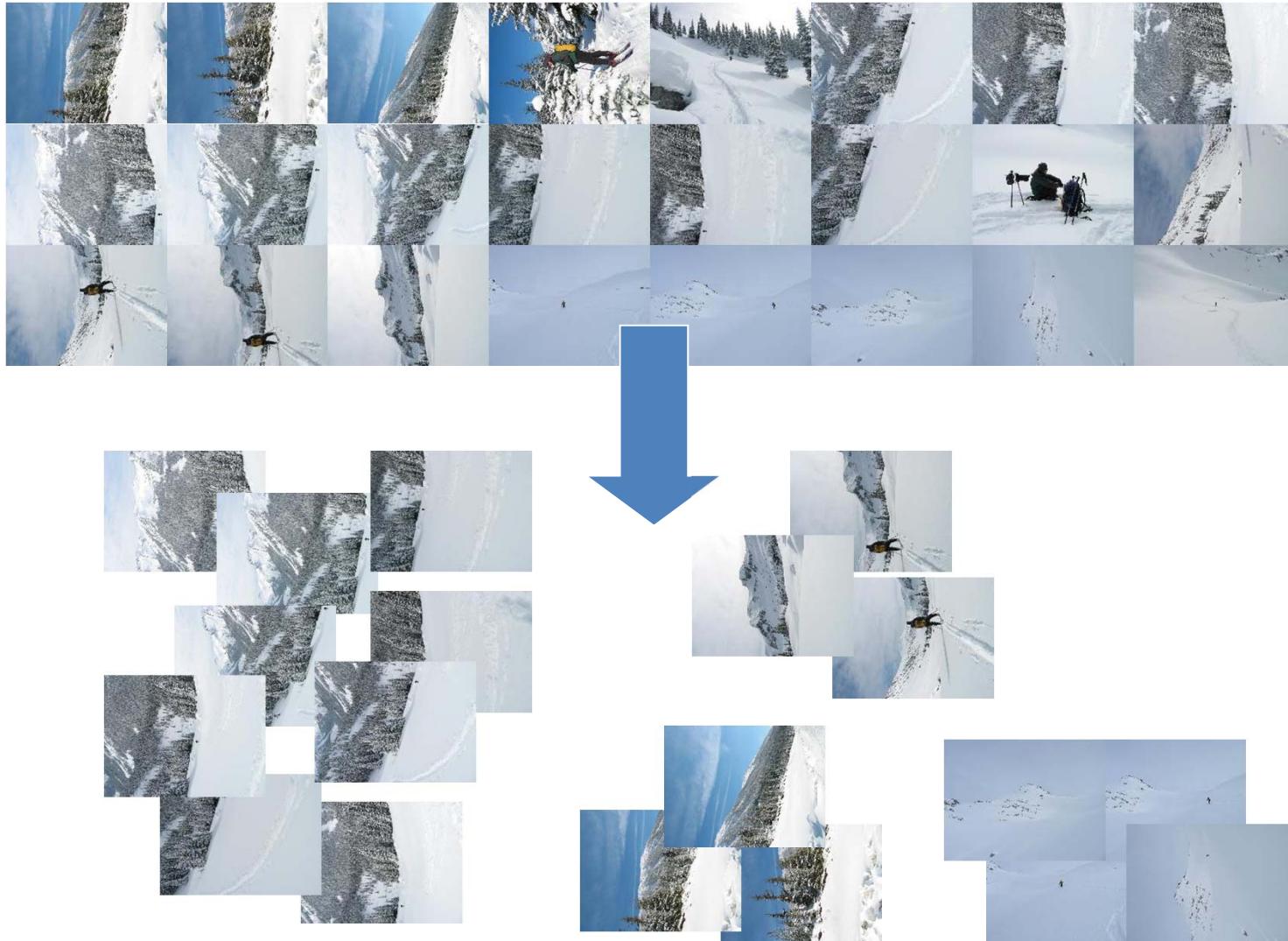
Slide credit: Matthew Brown

Finding the panoramas



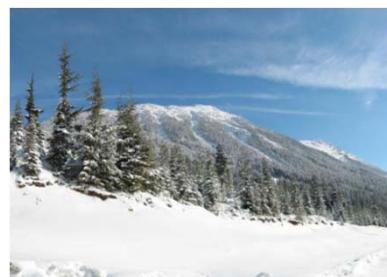
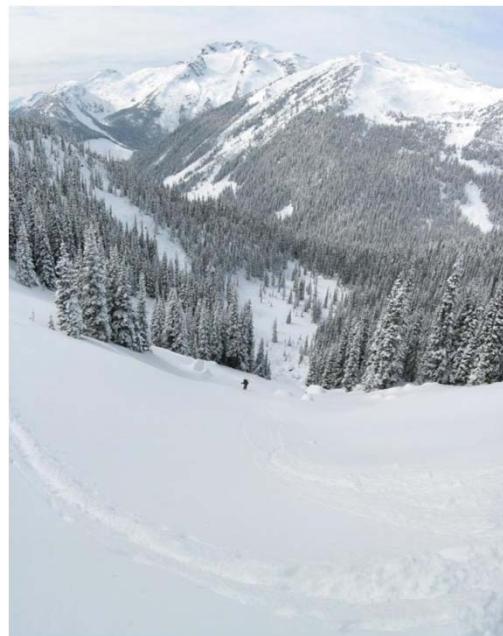
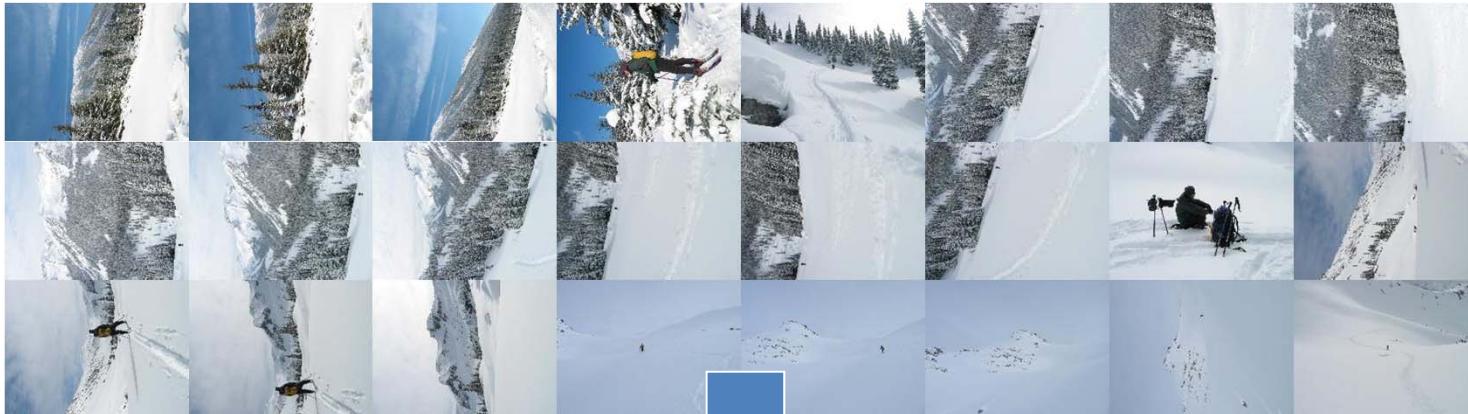
Slide credit: Matthew Brown

Finding the panoramas



Slide credit: Matthew Brown

Finding the panoramas



Slide credit: Matthew Brown

Algorithm (Brown & Lowe, 2003)

1. Feature Matching
2. Image Matching
3. Bundle Adjustment (extra reading: H & Z, Chp 18)
 - a) Error function
4. Multi-band Blending

Error function

- Sum of squared projection errors

$$e = \sum_{i=1}^n \sum_{j \in \mathcal{I}(i)} \sum_{k \in \mathcal{F}(i,j)} f(\mathbf{r}_{ij}^k)^2$$

- n = #images
- $\mathcal{I}(i)$ = set of image matches to image i
- $\mathcal{F}(i, j)$ = set of feature matches between images i,j
- \mathbf{r}_{ij}^k = residual of kth feature match between images i,j

- Robust error function

$$f(\mathbf{x}) = \begin{cases} |\mathbf{x}|, & \text{if } |\mathbf{x}| < x_{max} \\ x_{max}, & \text{if } |\mathbf{x}| \geq x_{max} \end{cases}$$

Bundle Adjustment

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



Slide credit: Matthew

Bundle Adjustment

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



Slide credit: Matthew Brown

Algorithm (Brown & Lowe, 2003)

1. Feature Matching
2. Image Matching
3. Bundle Adjustment
4. Multi-band Blending

Multi-band Blending

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



Slide credit: Matthew Brown

2-band Blending



Low frequency ($\lambda > 2$ pixels)



High frequency ($\lambda < 2$ pixels)

Slide credit:

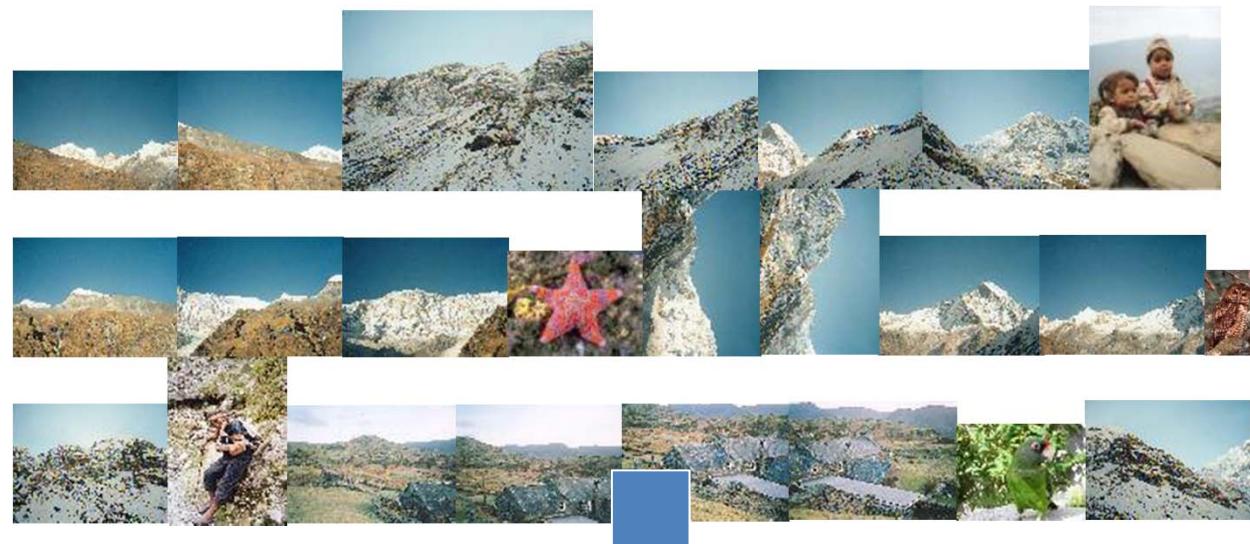
Linear Blending



2-band Blending



Results



Slide credit: Matthew Brown

Results



(a) Matier data set (7 images)



iPhone version
available



(b) Matier final stitch

[Brown, Szeliski, and Winder, 2005]

<http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html>

Applications of Local Invariant Features

- Wide baseline stereo
- Motion tracking
- Mobile robot navigation
- 3D reconstruction
- Recognition
 - Specific objects
 - Textures
 - Categories
- ...

Wide-Baseline Stereo



Image from T. Tuytelaars ECCV 2006 tutorial

Recognition of Specific Objects, Scenes



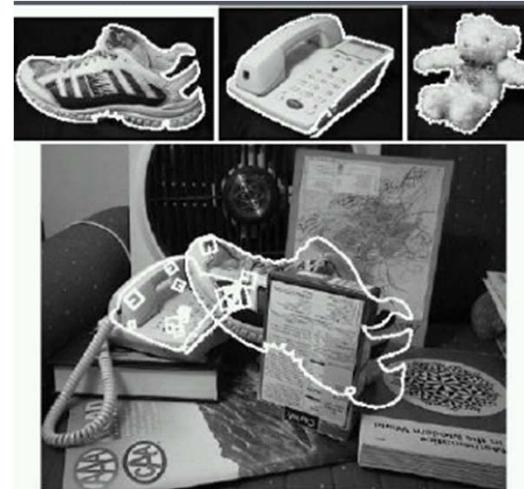
Schmid and Mohr 1997



Sivic and Zisserman, 2003



Rothganger et al. 2003

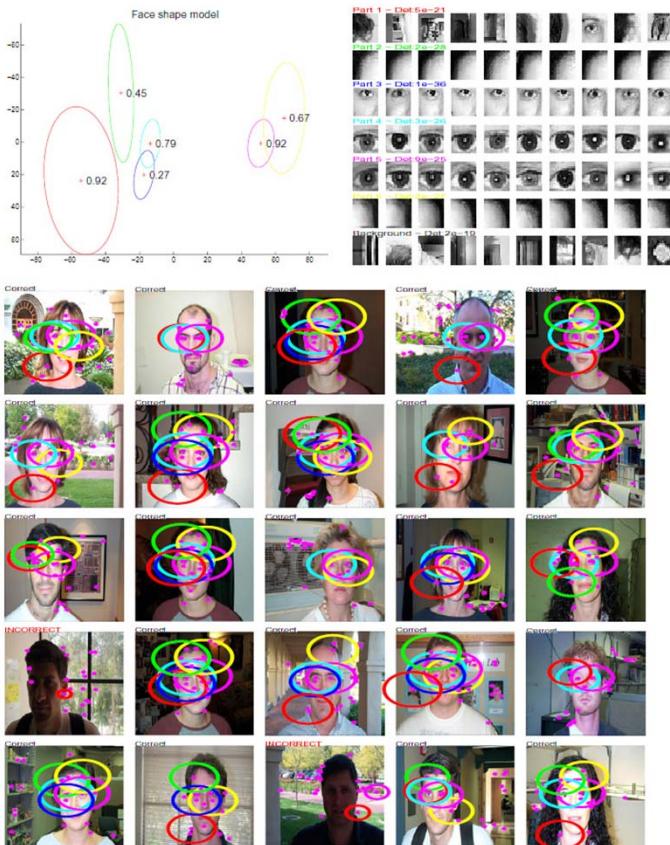


Lowe 2002

Slide credit: Kristen Grauman

Recognition of Categories

Constellation model



Weber et al. (2000)

Fergus et al. (2003)

Fei-Fei et al. (2003)

Bags of words

Database	Sample cluster #1	Sample cluster #2
Airplanes		
Motorbikes		
Leaves		
Wild Cats		
Faces		
Bicycles		
People		

Csurka et al. (2004)

Fei-Fei & Perona (2005)

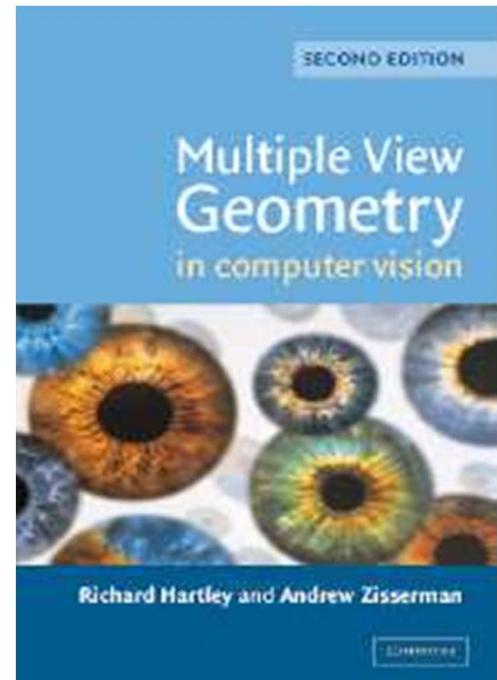
Dorko & Schmid (2005)

Sivic et al. (2005)

Lazebnik et al. (2006), ...

References and Further Reading

- More details on homography estimation can be found in Chapter 4.7 of
 - R. Hartley, A. Zisserman
Multiple View Geometry in Computer Vision
2nd Ed., Cambridge Univ. Press, 2004
- Details about the DoG detector and the SIFT descriptor can be found in
 - D. Lowe, [Distinctive image features from scale-invariant keypoints](#),
IJCV 60(2), pp. 91-110, 2004
- Try the available local feature detectors and descriptors
 - <http://www.robots.ox.ac.uk/~vgg/research/affine/detectors.html#binaries>



What we have learned today

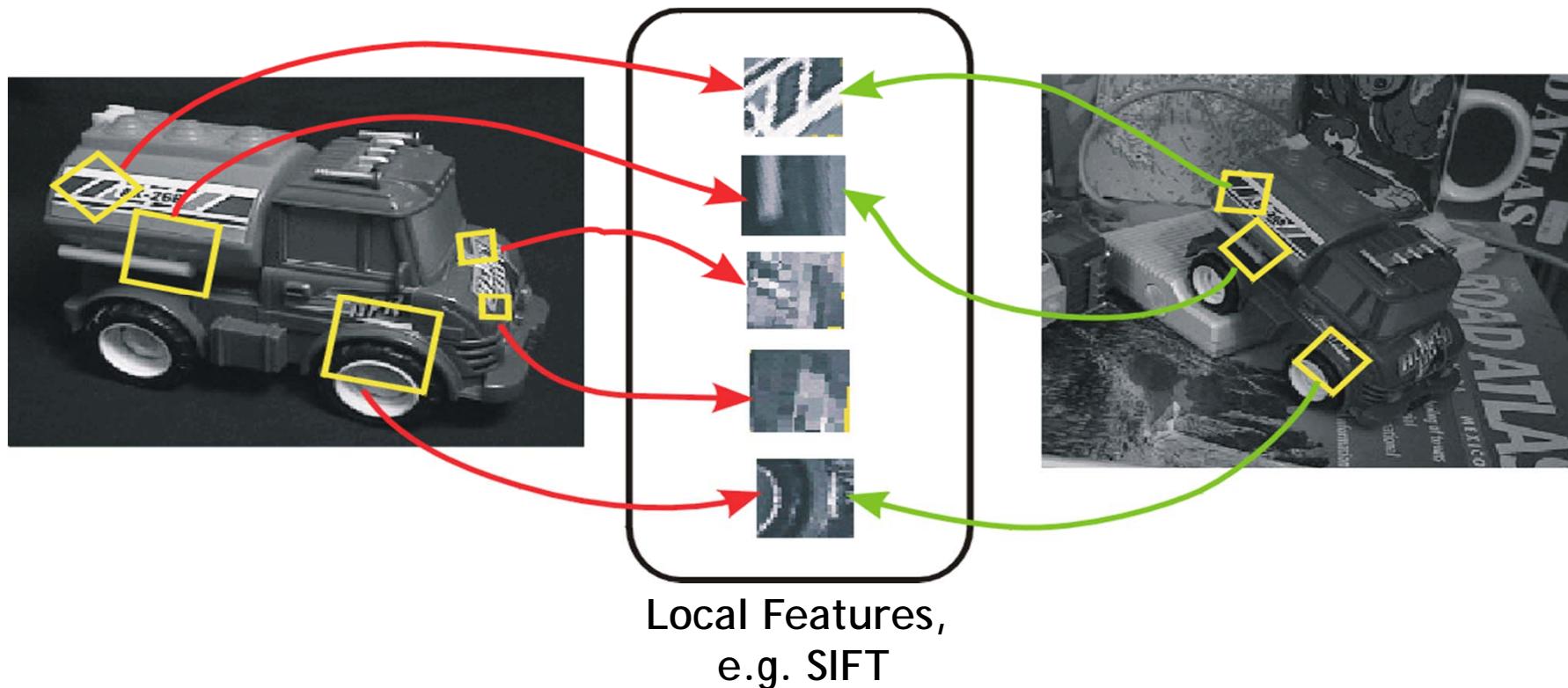
- Local descriptor
 - SIFT (**Problem Set 3 (Q2)**)
 - An assortment of other descriptors (**Problem Set 3 (Q4)**)
- Recognition and matching with local features
(Problem Set 3 (Q3))
 - Matching objects with local features: David Lowe
 - Panorama stitching: Brown & Lowe
 - Applications

Supplementary materials

- Matching images
 - Affine Transformation
 - Projective Transformation (Homography)

Recognition with Local Features

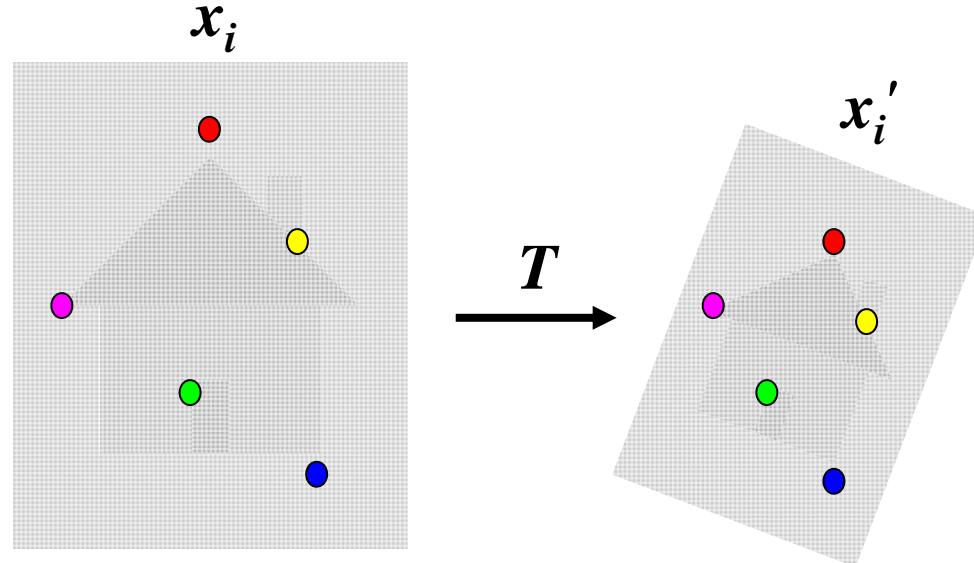
- Image content is transformed into local features that are invariant to translation, rotation, and scale
- Goal: Verify if they belong to a consistent configuration



Slide credit: David Lowe

Alignment Problem

- We have previously considered how to fit a model to image evidence
 - e.g., a line to edge points
- In alignment, we will fit the parameters of some transformation according to a set of matching feature pairs (“correspondences”).



Basic 2D Transformations

- Basic 2D transformations as 3x3 matrices

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

Translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scaling

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Rotation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Shearing

What Can be Represented by a 2×2 Matrix?

- 2D Scaling?

$$x' = s_x * x$$

$$y' = s_y * y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- 2D Rotation around (0,0)?

$$x' = \cos \theta * x - \sin \theta * y$$

$$y' = \sin \theta * x + \cos \theta * y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- 2D Shearing?

$$x' = x + sh_x * y$$

$$y' = sh_y * x + y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

What Can be Represented by a 2×2 Matrix?

- 2D Mirror about y axis?

$$x' = -x$$

$$y' = y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- 2D Mirror over (0,0)?

$$x' = -x$$

$$y' = -y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- 2D Translation?

$$x' = x + t_x$$

$$y' = y + t_y$$

NO!

2D Linear Transforms

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Only linear 2D transformations can be represented with a 2x2 matrix.
- Linear transformations are combinations of ...
 - Scale,
 - Rotation,
 - Shear, and
 - Mirror

Homogeneous Coordinates

- Q: How can we represent translation as a 3x3 matrix using homogeneous coordinates?

$$\mathbf{x}' = \mathbf{x} + \mathbf{t}_x$$

$$\mathbf{y}' = \mathbf{y} + \mathbf{t}_y$$

- A: Using the rightmost column:

$$\text{Translation} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

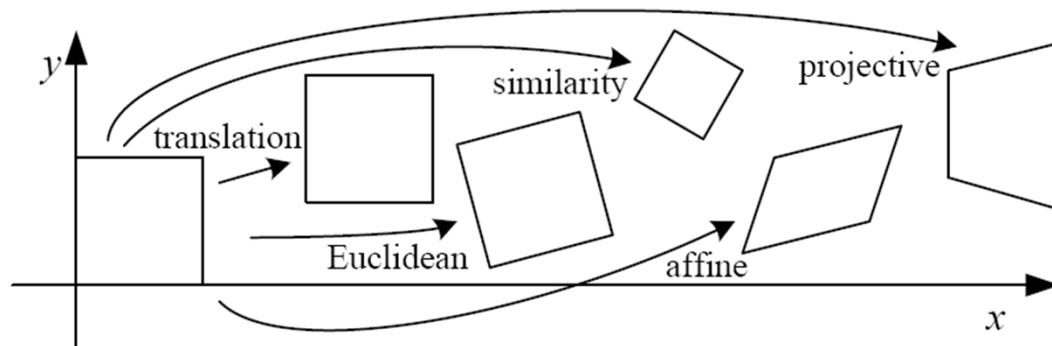
2D Affine Transformation

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Projective Transformation

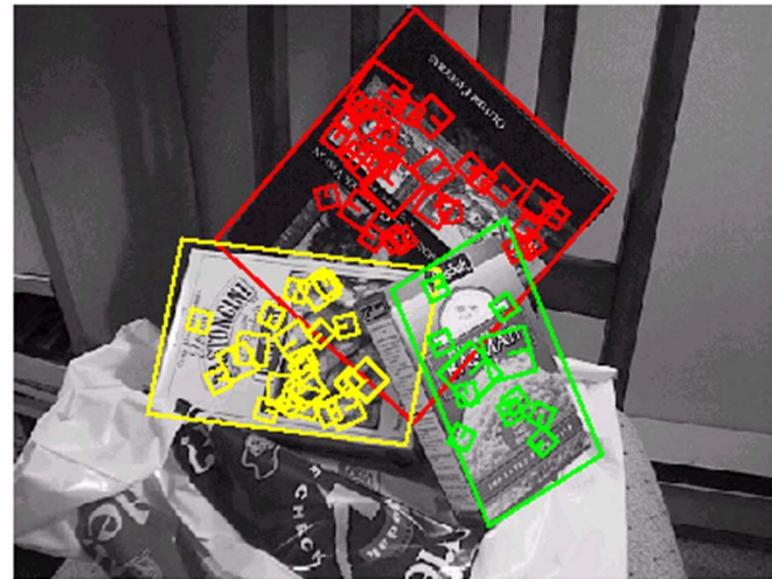
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- **Affine transformations:**
 - Linear transformations
 - Translations
- Parallel lines remain parallel
- **Projective transformations:**
 - Affine transformations, and
 - Projective warps
- Parallel lines do not necessarily remain parallel



Let's Start with Affine Transformations

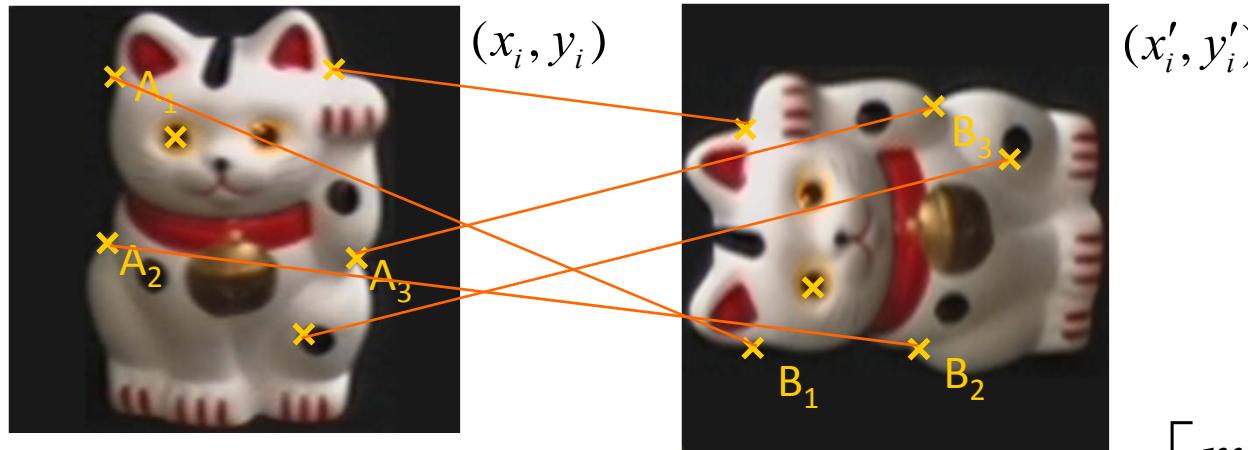
- Simple fitting procedure (linear least squares)
- Approximates viewpoint changes for roughly planar objects and roughly orthographic cameras
- Can be used to initialize fitting for more complex models



Slide credit: Svetlana Lazebnik, David Lowe

Fitting an Affine Transformation

- Assuming we know the correspondences, how do we get the transformation?



$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

➡

$$\begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \quad \\ \quad \\ \quad \\ \quad \\ \quad \\ \quad \end{bmatrix}$$

Slide credit: Bastian Leibe

Fitting an Affine Transformation

- Assuming we know the correspondences, how do we get the transformation?

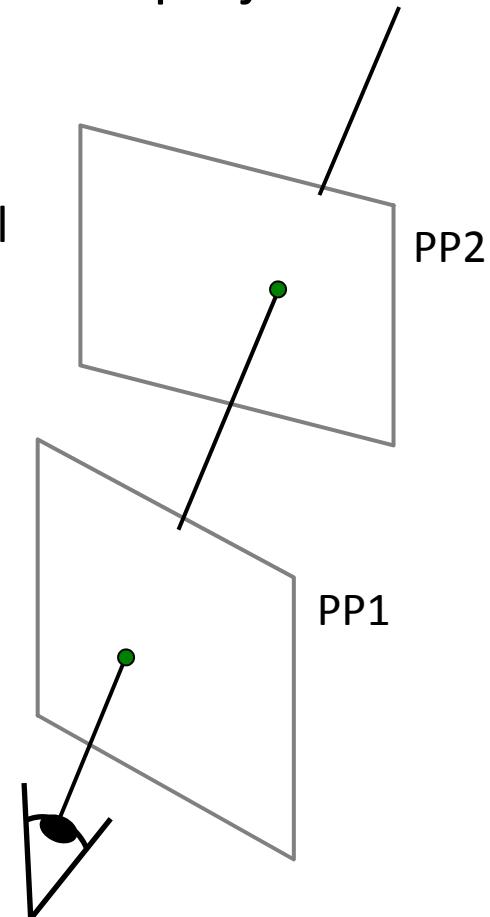
$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} \dots & & & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ \dots & & & & & \\ \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \dots \\ x'_i \\ y'_i \\ \dots \end{bmatrix}$$

- How many matches (correspondence pairs) do we need to solve for the transformation parameters?
- Once we have solved for the parameters, how do we compute the coordinates of the corresponding point for (x_{new}, y_{new}) ?

Fitting a Projective Transformation (i.e. Homography)

- A projective transform is a mapping between any two perspective projections with the same center of projection.
 - I.e. two planes in 3D along the same sight ray
- Properties
 - Rectangle should map to arbitrary quadrilateral
 - Parallel lines aren't
 - but must preserve straight lines
- This is called a **homography**

$$\begin{bmatrix} wx' \\ wy' \\ w \\ p' \end{bmatrix} = \begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \\ H & & \end{bmatrix} \begin{bmatrix} x \\ y \\ l \\ p \end{bmatrix}$$



Slide credit: Alyosha Efros

Fitting a Projective Transformation (i.e. Homography)

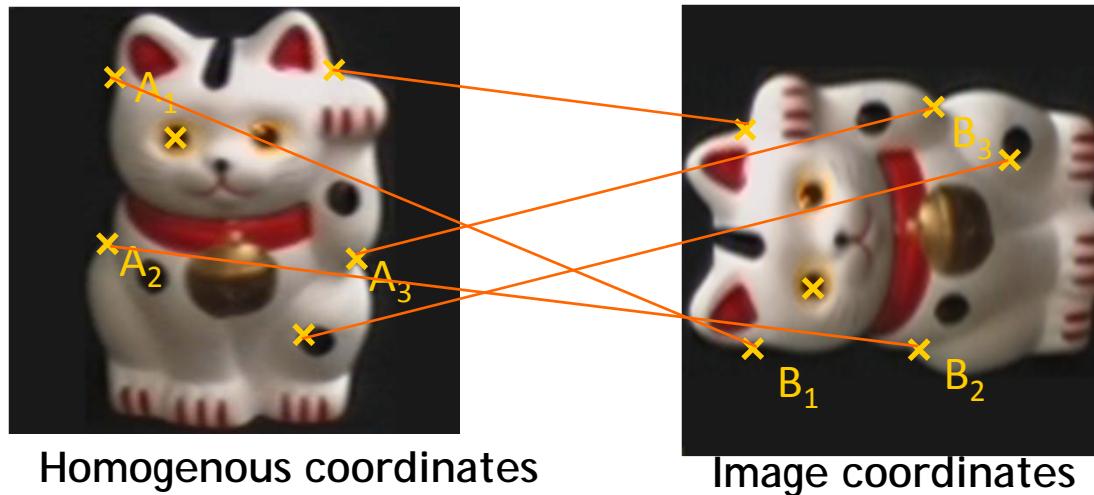
- A projective transform is a mapping between any two perspective projections with the same center of projection.
 - I.e. two planes in 3D along the same sight ray
- Properties
 - Rectangle should map to arbitrary quadrilateral
 - Parallel lines aren't
 - but must preserve straight lines
- This is called a **homography**

$$\begin{bmatrix} wx' \\ wy' \\ w \\ p \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & H \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \\ p \end{bmatrix}$$

Set scale factor to 1
→ 8 parameters left.

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$

$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

⋮

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

Matrix notation

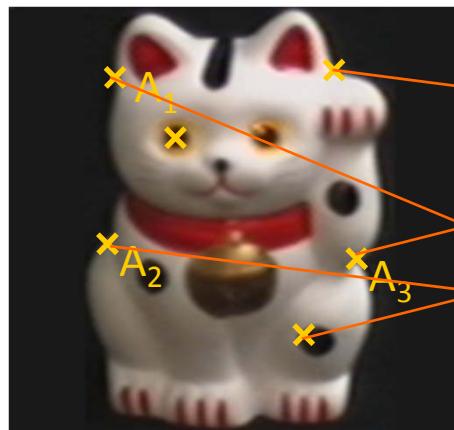
$$x' = Hx$$

$$x'' = \frac{1}{z'} x'$$

Slide credit: Krystian Mikolajczyk

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation



Homogenous coordinates

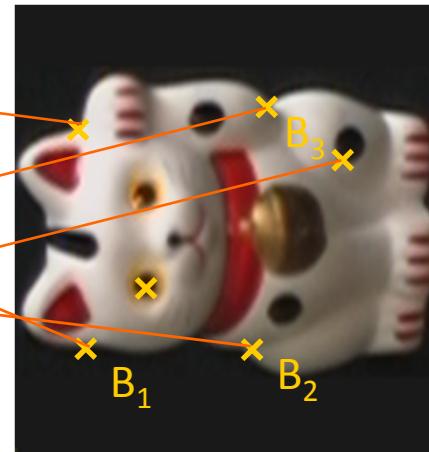


Image coordinates

$$\begin{aligned}\mathbf{x}_{A_1} &\leftrightarrow \mathbf{x}_{B_1} \\ \mathbf{x}_{A_2} &\leftrightarrow \mathbf{x}_{B_2} \\ \mathbf{x}_{A_3} &\leftrightarrow \mathbf{x}_{B_3} \\ &\vdots\end{aligned}$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

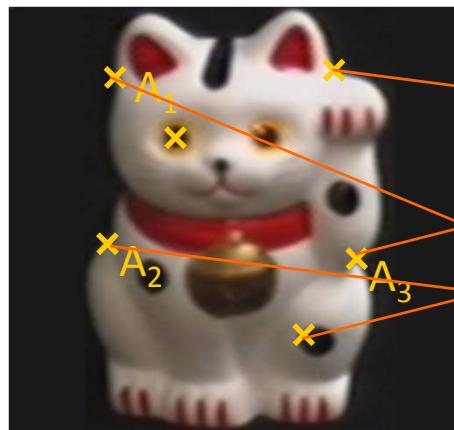
Matrix notation

$$x' = Hx$$
$$x'' = \frac{1}{z'} x'$$

Slide credit: Krystian Mikolajczyk

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation



Homogenous coordinates

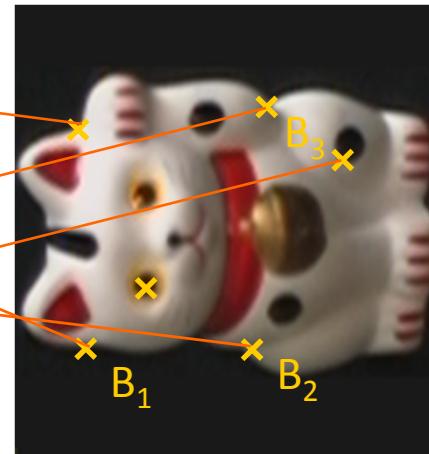


Image coordinates

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$

$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

⋮

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x_{A_1} = \frac{h_{11} x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31} x_{B_1} + h_{32} y_{B_1} + 1}$$

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

Matrix notation

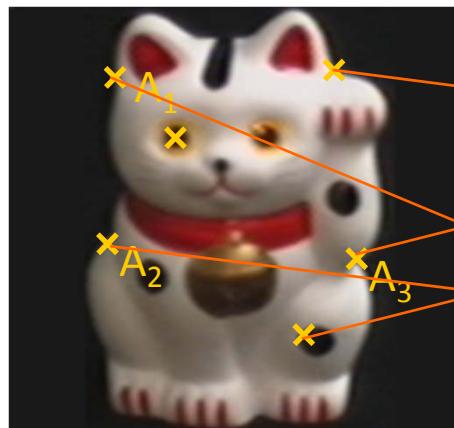
$$x' = Hx$$

$$x'' = \frac{1}{z'} x'$$

Slide credit: Krystian Mikolajczyk

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation



Homogenous coordinates

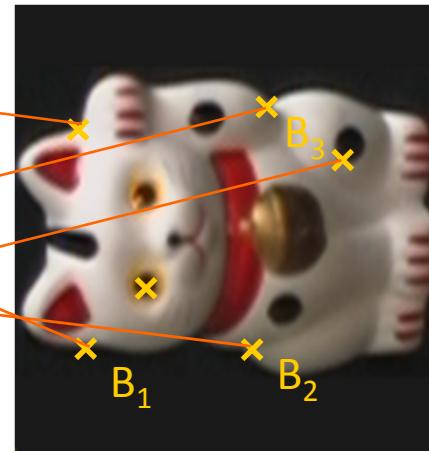


Image coordinates

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$

$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

\vdots

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x_{A_1} = \frac{h_{11} x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31} x_{B_1} + h_{32} y_{B_1} + 1}$$

$$\begin{bmatrix} x'' \\ y'' \\ 1 \end{bmatrix} = \frac{1}{z'} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

$$y_{A_1} = \frac{h_{21} x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31} x_{B_1} + h_{32} y_{B_1} + 1}$$

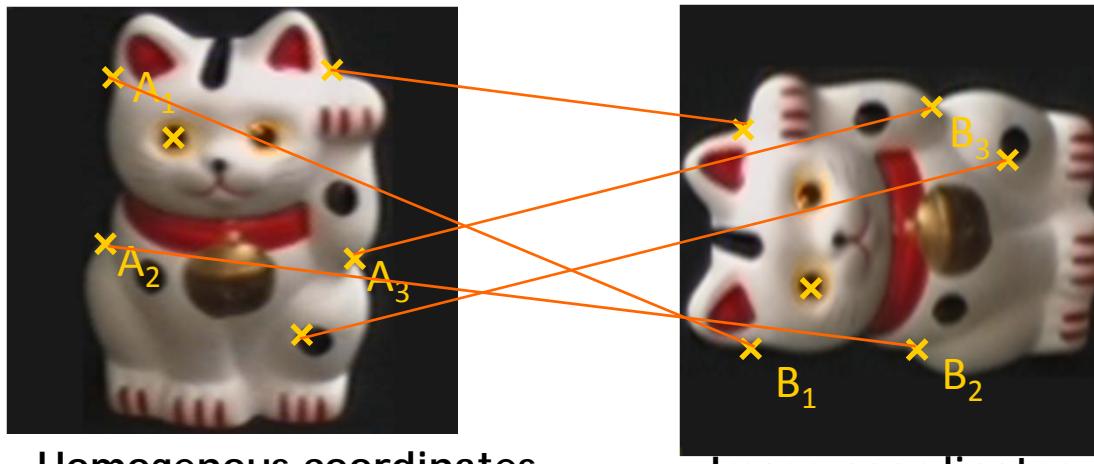
Matrix notation

$$x' = Hx$$

$$x'' = \frac{1}{z'} x'$$

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$

$$x_{A_1} = \frac{h_{11} x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31} x_{B_1} + h_{32} y_{B_1} + 1}$$

$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$y_{A_1} = \frac{h_{21} x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31} x_{B_1} + h_{32} y_{B_1} + 1}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

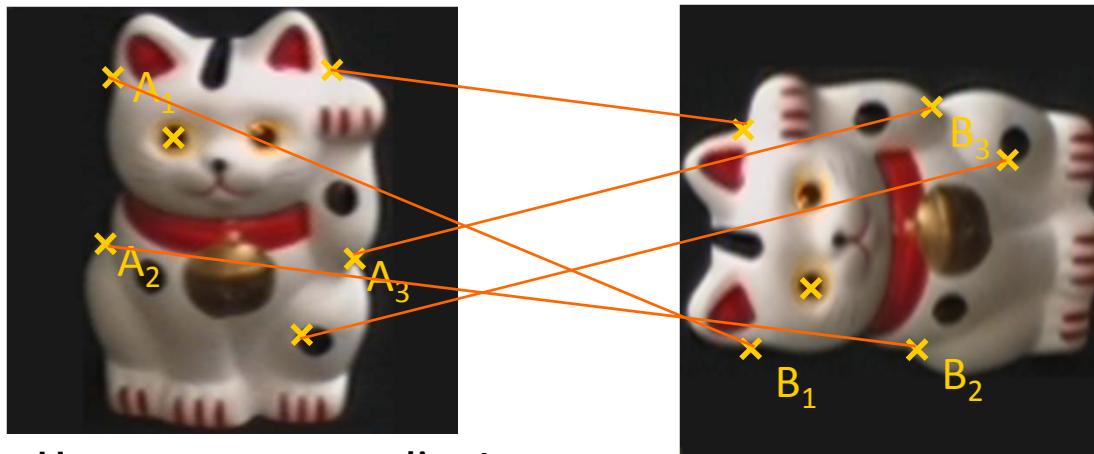
$$x_{A_1} h_{31} x_{B_1} + x_{A_1} h_{32} y_{B_1} + x_{A_1} = h_{11} x_{B_1} + h_{12} y_{B_1} + h_{13}$$

⋮

Slide credit: Krystian Mikolajczyk

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation



Homogenous coordinates

Image coordinates

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$

$$x_{A_1} = \frac{h_{11}x_{B_1} + h_{12}y_{B_1} + h_{13}}{h_{31}x_{B_1} + h_{32}y_{B_1} + 1}$$

$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$y_{A_1} = \frac{h_{21}x_{B_1} + h_{22}y_{B_1} + h_{23}}{h_{31}x_{B_1} + h_{32}y_{B_1} + 1}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

$$x_{A_1}h_{31}x_{B_1} + x_{A_1}h_{32}y_{B_1} + x_{A_1} = h_{11}x_{B_1} + h_{12}y_{B_1} + h_{13}$$

⋮

$$h_{11}x_{B_1} + h_{12}y_{B_1} + h_{13} - x_{A_1}h_{31}x_{B_1} - x_{A_1}h_{32}y_{B_1} - x_{A_1} = 0$$

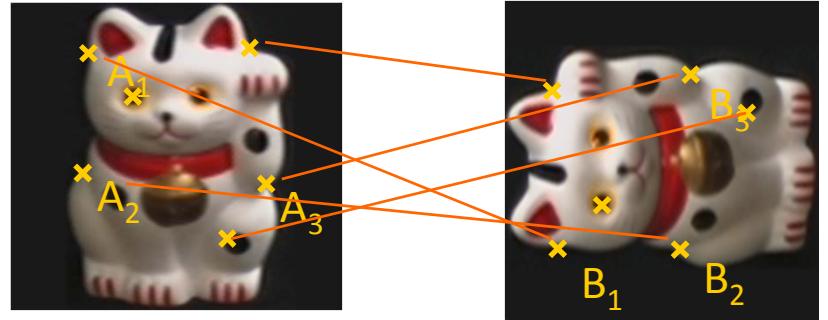
$$h_{21}x_{B_1} + h_{22}y_{B_1} + h_{23} - y_{A_1}h_{31}x_{B_1} - y_{A_1}h_{32}y_{B_1} - y_{A_1} = 0$$

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation

$$h_{11} x_{B_1} + h_{12} y_{B_1} + h_{13} - x_{A_1} h_{31} x_{B_1} - x_{A_1} h_{32} y_{B_1} - x_{A_1} = 0$$

$$h_{21} x_{B_1} + h_{22} y_{B_1} + h_{23} - y_{A_1} h_{31} x_{B_1} - y_{A_1} h_{32} y_{B_1} - y_{A_1} = 0$$



$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$

$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$

$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$

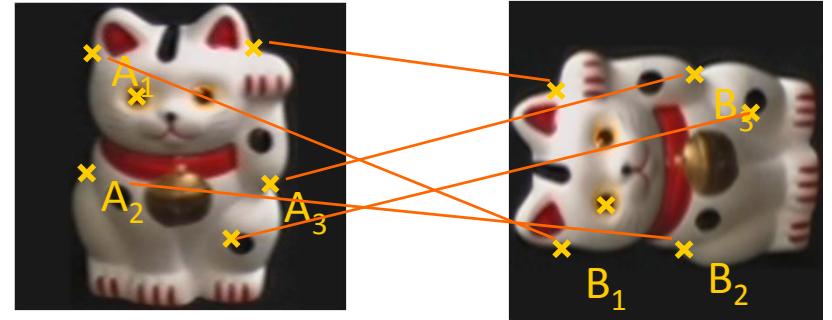
⋮

$$\begin{array}{l} \mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1} \\ \mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2} \\ \mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3} \\ \vdots \end{array} \left[\begin{array}{ccccccccc} x_{B_1} & y_{B_1} & 1 & 0 & 0 & 0 & -x_{A_1}x_{B_1} & -x_{A_1}y_{B_1} & -x_{A_1} \\ 0 & 0 & 0 & x_{B_1} & y_{B_1} & 1 & -y_{A_1}x_{B_1} & -y_{A_1}y_{B_1} & -y_{A_1} \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \end{array} \right] \cdot \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \end{bmatrix}$$

$$Ah = 0$$

Fitting a Projective Transformation (i.e. Homography)

- Estimating the transformation
- Solution:
 - Null-space vector of A
 - Corresponds to smallest eigenvector



SVD

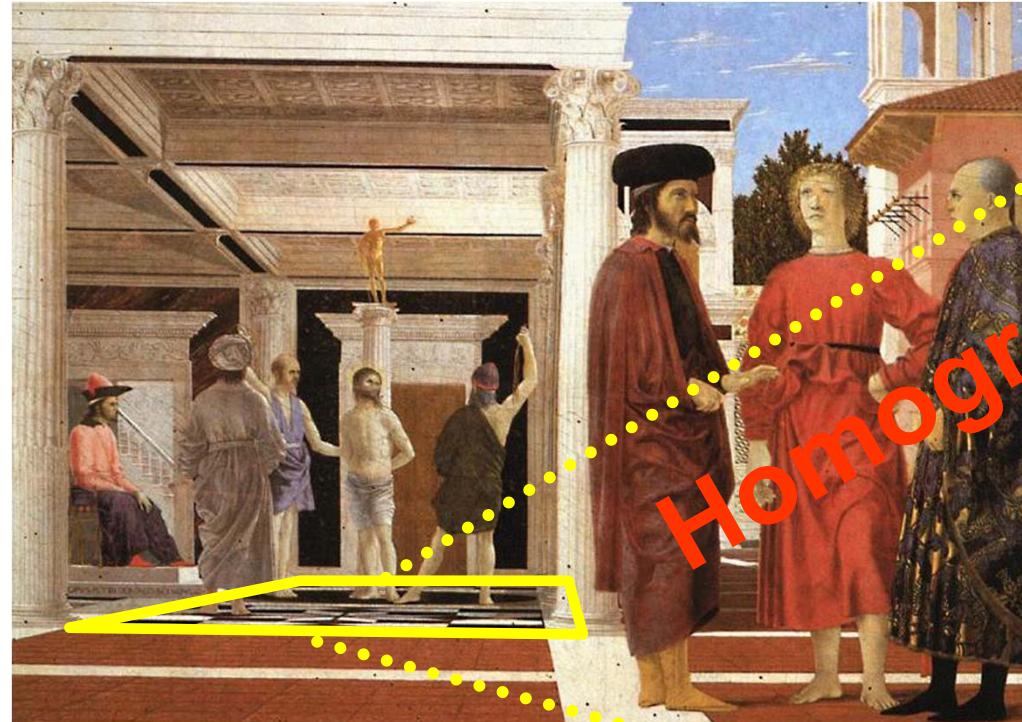
$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \mathbf{U} \begin{bmatrix} d_{11} & \cdots & d_{19} \\ \vdots & \ddots & \vdots \\ d_{91} & \cdots & d_{99} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{19} \\ \vdots & \ddots & \vdots \\ v_{91} & \cdots & v_{99} \end{bmatrix}^T$$

$\mathbf{h} = \frac{[v_{19}, \dots, v_{99}]}{v_{99}}$

Minimizes least square error

Uses: Analyzing Patterns and Shapes

- What is the shape of the b/w floor pattern?



Homography

The floor (enlarged)

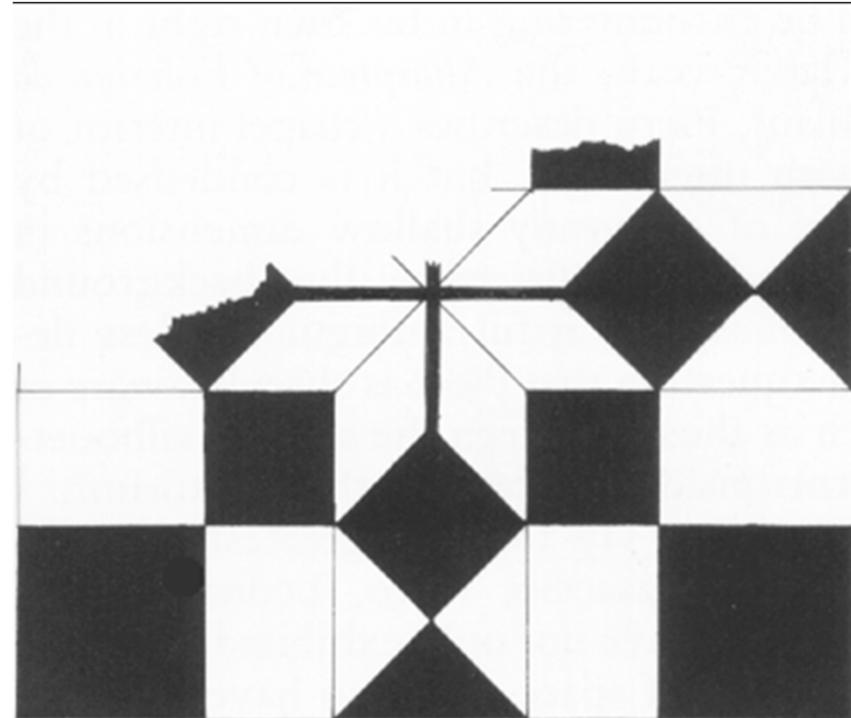


Slide credit: Antonio Criminisi

Analyzing Patterns and Shapes

Slide credit: Antonio Criminisi

Automatic rectification



From Martin Kemp *The Science of Art*
(manual reconstruction)