Perception VI: Homography. SIFT

Dr. Ilya Afanasyev, i.afanasyev@innopolis.ru

“Intelligent Mobile Robotics” course, Innopolis University, Russia, Spring 2017

Perception VI: Affine & Projective Transformations (Homography)

Dr. Ilya Afanasyev, i.afanasyev@innopolis.ru

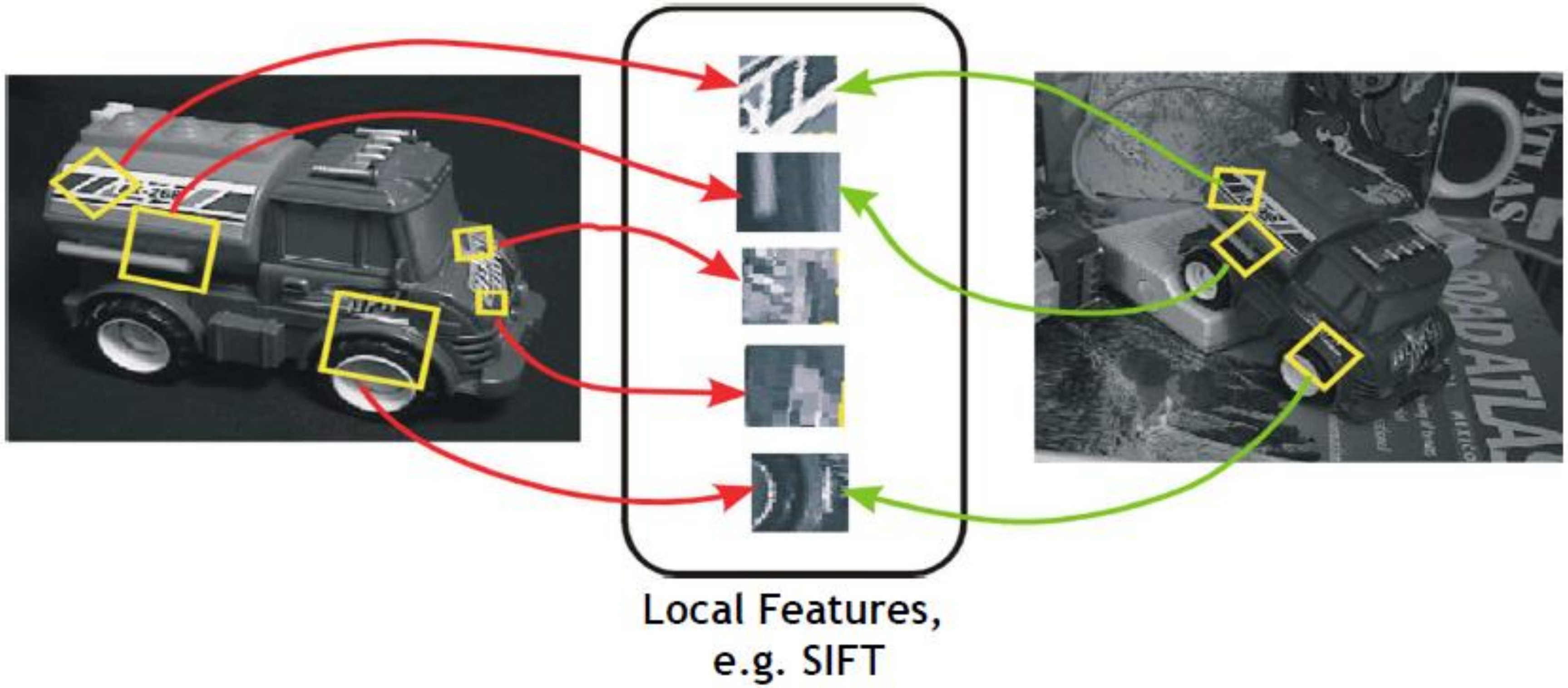
“Intelligent Mobile Robotics” course, Innopolis University, Russia, Spring 2017

Contents

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

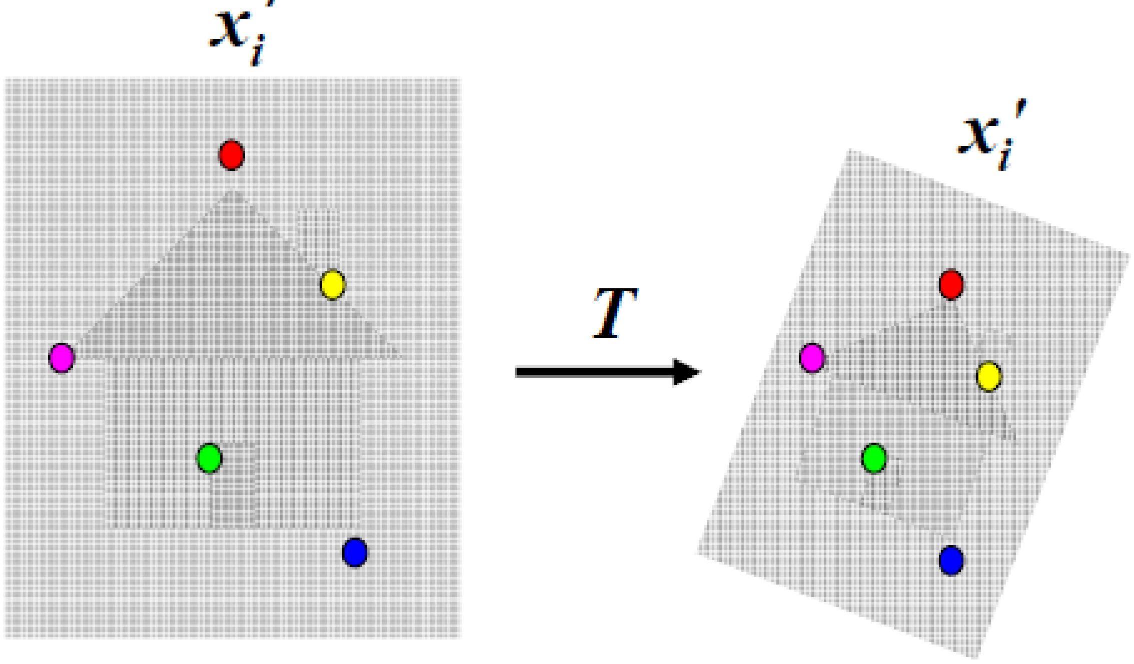
Recognition with Local Features

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



Alignment Problem

- 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 2x2 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 2x2 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 Transform

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

- 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?

$$x' = x + t_x$$

$$y' = y + t_y$$

- A: Using the rightmost column:

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

Spatial transformations

2D Affine Transformation

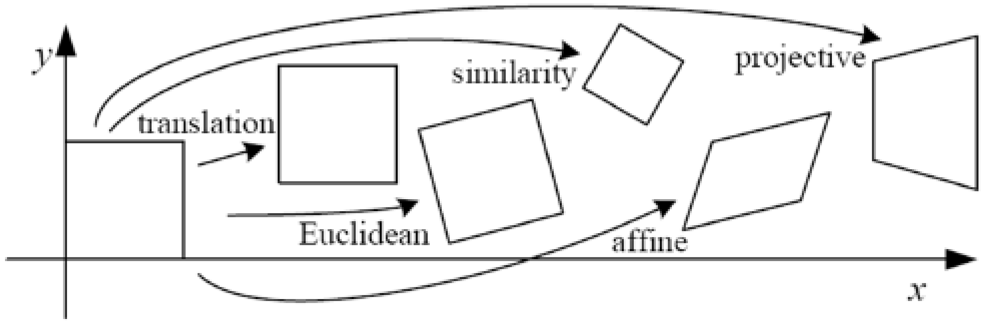
$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

- Affine transformations:
  - Linear transformations
  - Translations
- Parallel lines remain parallel

Projective Transformation

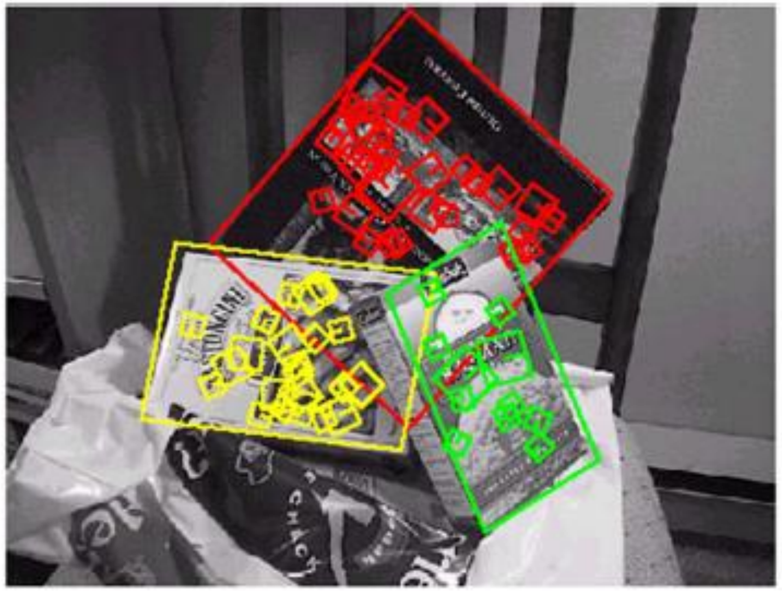
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

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



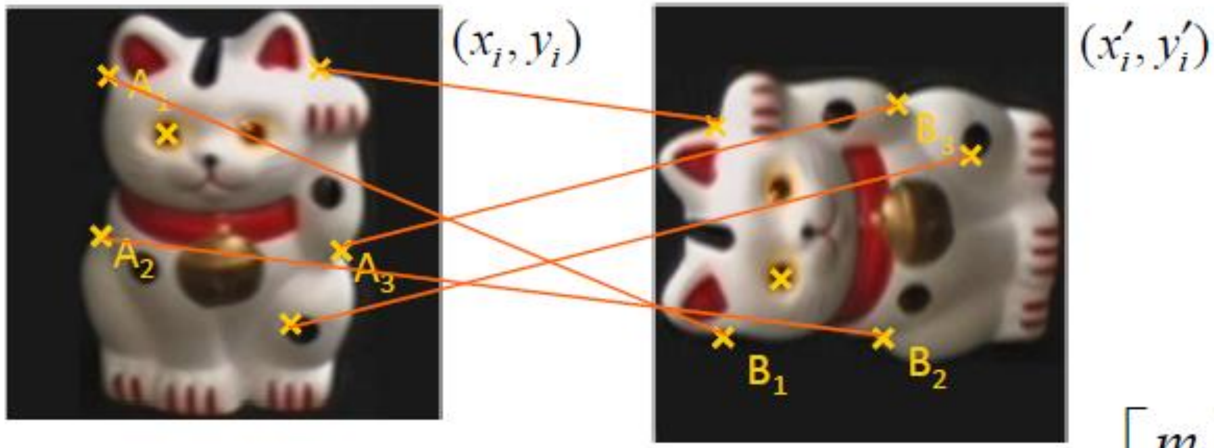
Let's Start with Affine Transformations

- 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



Fitting an Affine Transformation

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



$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \Rightarrow \begin{bmatrix} \quad & \quad \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \quad \end{bmatrix}$$

Fitting an Affine Transformation

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

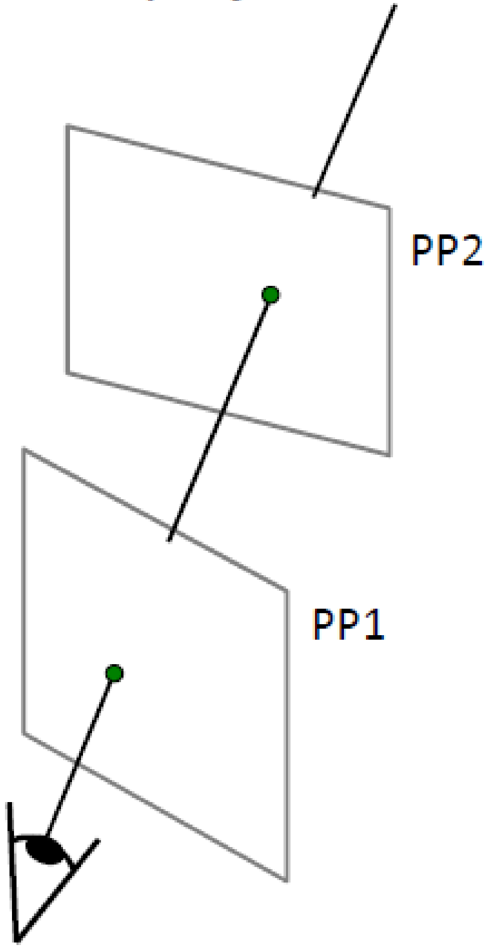
$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix} \Rightarrow \begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

- 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

- 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

$$\underbrace{\begin{bmatrix} wx' \\ wy' \\ w \end{bmatrix}}_{p'} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}}_{p}$$



Fitting a Projective Transformation

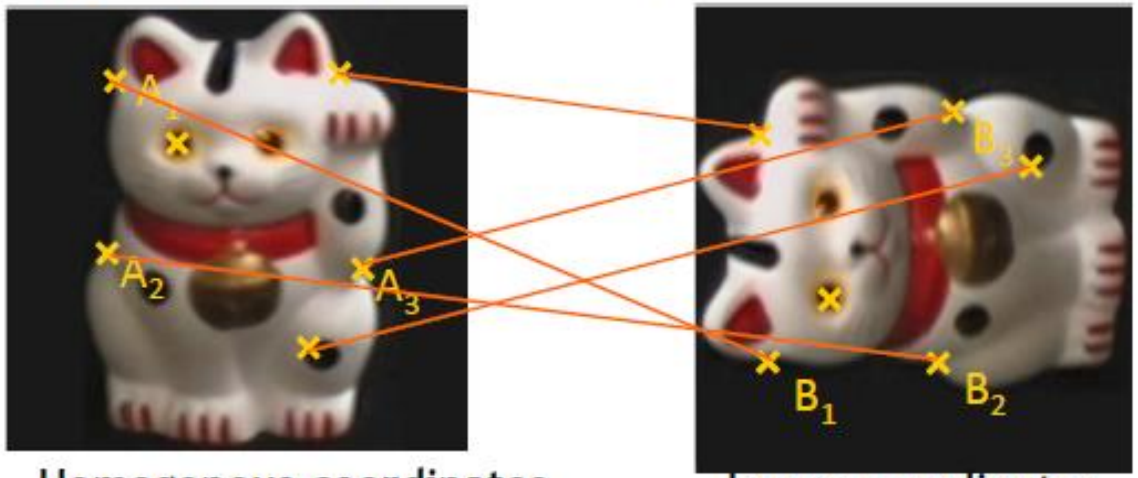
- A projective transform is a mapping between any two perspective projections with the same center of projection.
  - I.e. two planes in 3D along the same sight ray
- Properties
  - Rectangle should map to arbitrary quadrilateral
  - Parallel lines aren't
  - but must preserve straight lines
- This is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

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

Set scale factor to 1
$\Rightarrow$ 8 parameters left.

Projective Transformation (Homography)

- Estimating the transformation



Homogenous coordinates　　　Image coordinates

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

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

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

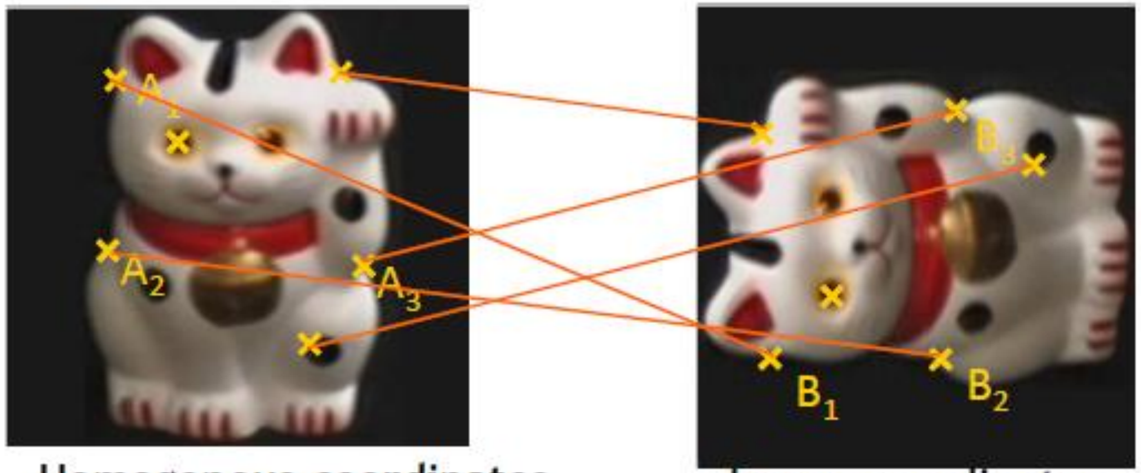
Matrix notation

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

© Slide credit: Krystian Mikolajczyk

Projective Transformation (Homography)

- Estimating the transformation



Homogenous coordinates

Image coordinates

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

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

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

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

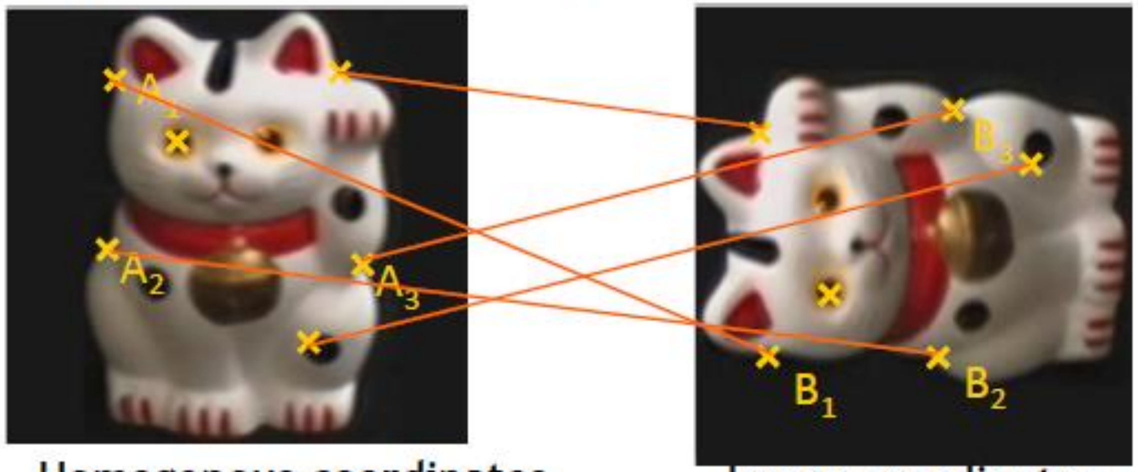
Matrix notation

$$x' = Hx$$

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

Projective Transformation (Homography)

- Estimating the transformation



Homogenous coordinates   Image coordinates

$$\mathbf{X}_{A_1} \leftrightarrow \mathbf{X}_{B_1}$$
$$\mathbf{X}_{A_2} \leftrightarrow \mathbf{X}_{B_2}$$
$$\mathbf{X}_{A_3} \leftrightarrow \mathbf{X}_{B_3}$$
$$\vdots$$

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

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

Matrix notation

$$x' = Hx$$

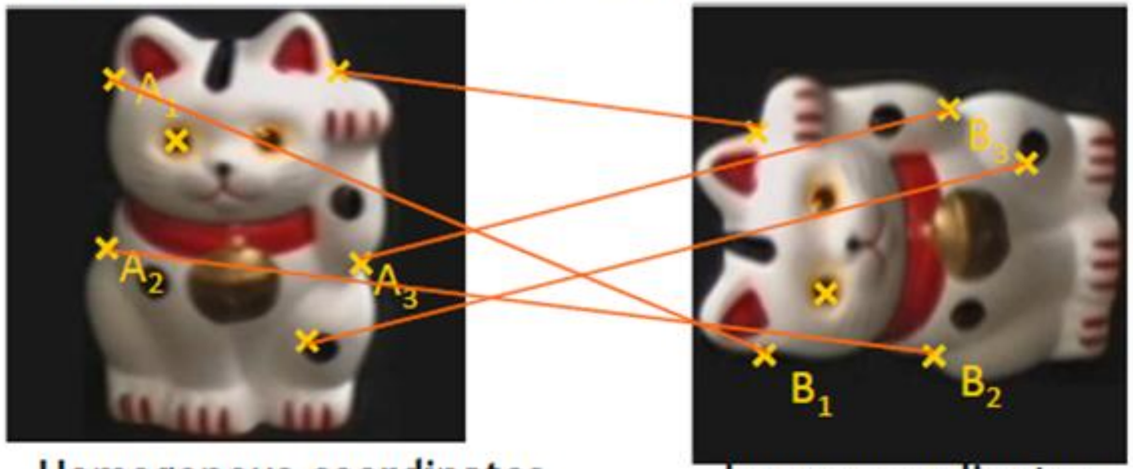
$$x'' = \frac{1}{z'} x'$$

$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1} \qquad y_{A_1} = \frac{h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

© Slide credit: Krystian Mikolajczyk

Projective Transformation (Homography)

- Estimating the transformation



Homogenous coordinates | Image coordinates

$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

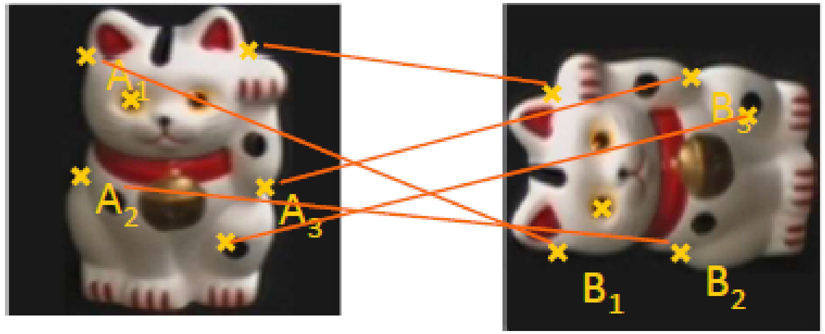
$$x_{A_1} = \frac{h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1} \qquad y_{A_1} = \frac{h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23}}{h_{31}\, x_{B_1} + h_{32} y_{B_1} + 1}$$

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

$$\boxed{\begin{aligned} h_{11}\, x_{B_1} + h_{12} y_{B_1} + h_{13} - x_{A_1} h_{31}\, x_{B_1} - x_{A_1} h_{32} y_{B_1} - x_{A_1} &= 0 \\ h_{21}\, x_{B_1} + h_{22} y_{B_1} + h_{23} - y_{A_1} h_{31}\, x_{B_1} - y_{A_1} h_{32} y_{B_1} - y_{A_1} &= 0 \end{aligned}}$$

Projective Transformation (Homography)

- Estimating the transformation



$$\boxed{\begin{array}{l} h_{11}\,x_{B_1} + h_{12}y_{B_1} + h_{13} - x_{A_1}h_{31}\,x_{B_1} - x_{A_1}h_{32}y_{B_1} - x_{A_1} = 0 \\ h_{21}\,x_{B_1} + h_{22}y_{B_1} + h_{23} - y_{A_1}h_{31}\,x_{B_1} - y_{A_1}h_{32}y_{B_1} - y_{A_1} = 0 \end{array}}$$

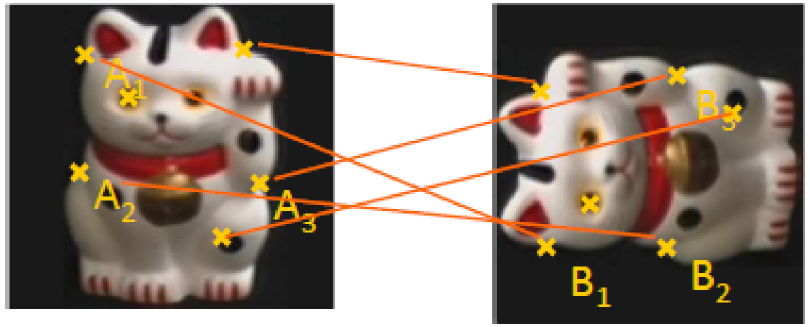
$$\begin{array}{l} \mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1} \\ \mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2} \\ \mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3} \\ \vdots \end{array} \qquad \begin{bmatrix} x_{B_1} & y_{B_1} & 1 & 0 & 0 & 0 & -x_{A_1}x_{B_1} & -x_{A_1}y_{B_1} & -x_{A_1}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

$$Ah = 0$$

© Slide credit: Krystian Mikolajczyk

Projective Transformation (Homography)

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



$$Ah = 0$$

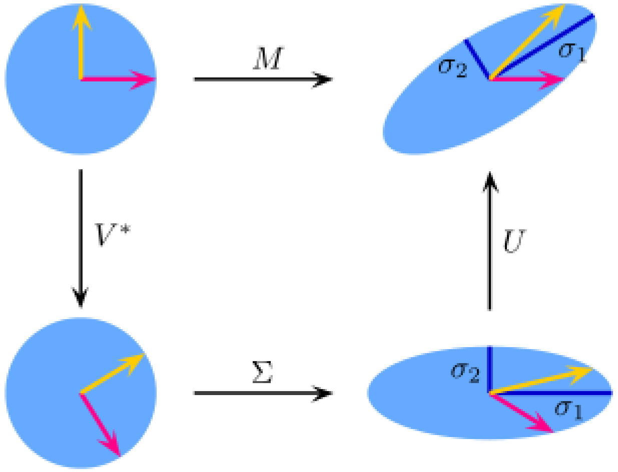
$$\mathbf{x}_{A_1} \leftrightarrow \mathbf{x}_{B_1}$$
$$\mathbf{x}_{A_2} \leftrightarrow \mathbf{x}_{B_2}$$
$$\mathbf{x}_{A_3} \leftrightarrow \mathbf{x}_{B_3}$$
$$\vdots$$

SVD

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

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

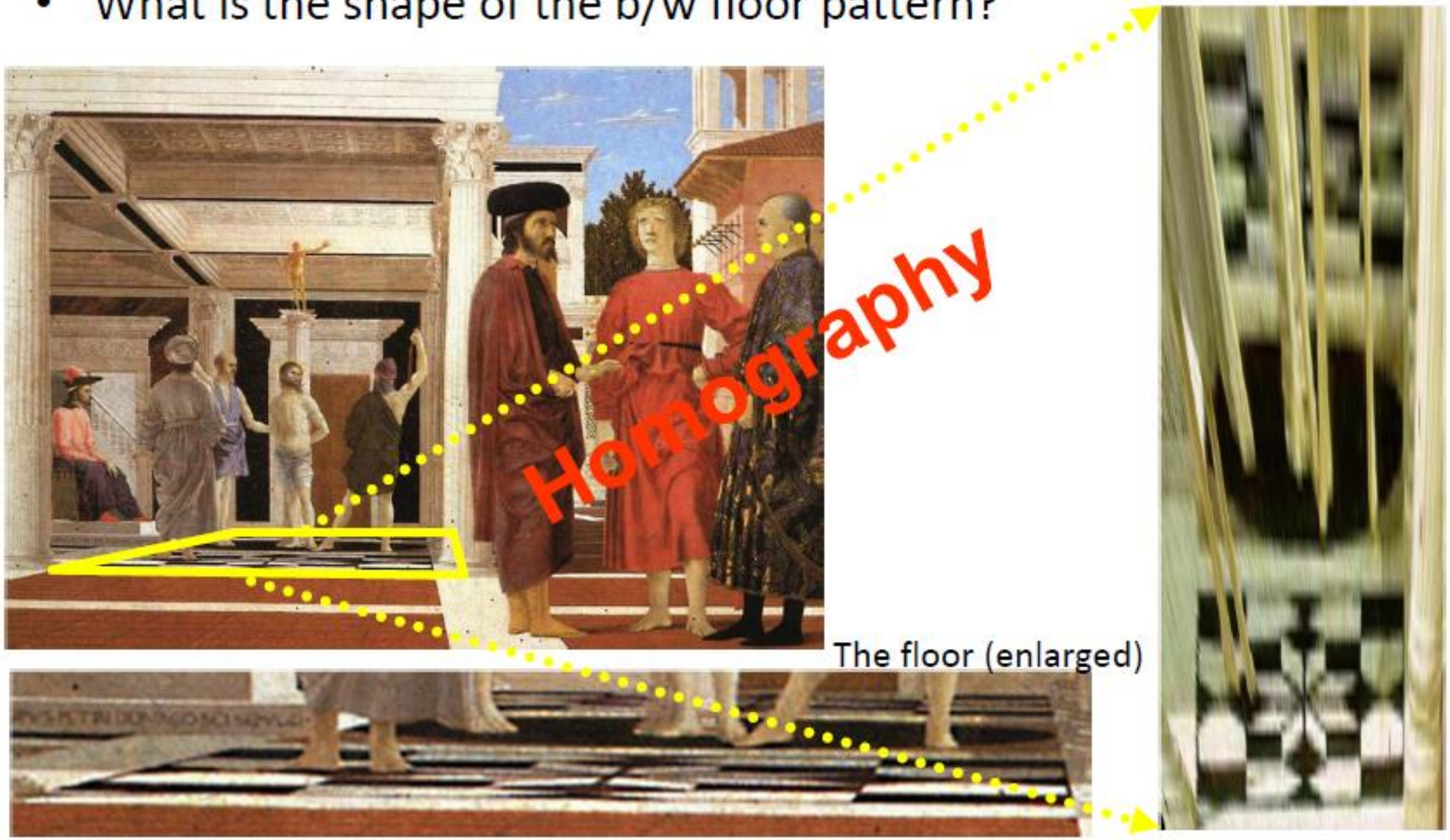
Minimizes least square error



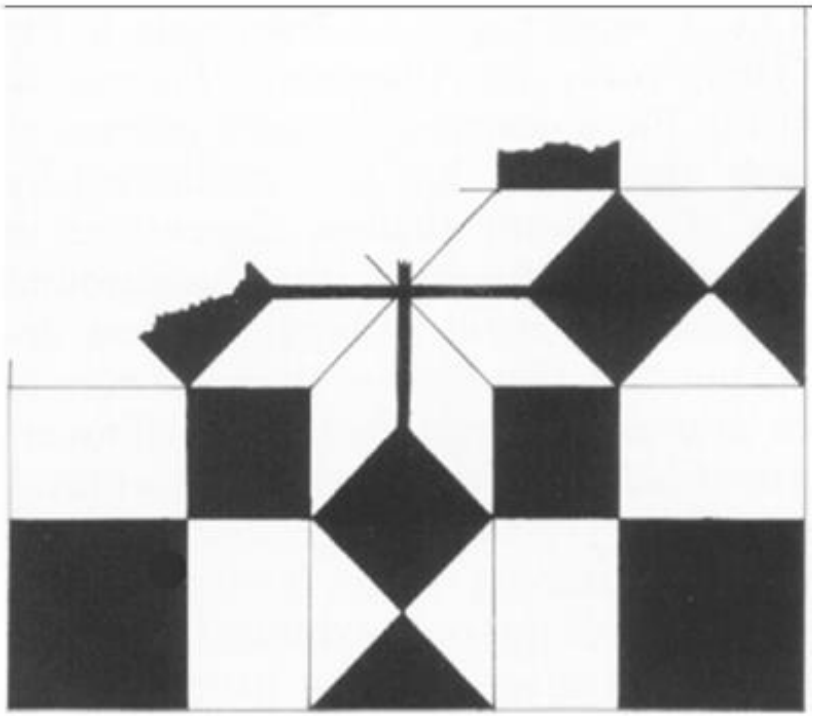
$$M = U \cdot \Sigma \cdot V^*$$

Uses: Analyzing Patterns and Shapes

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



Uses: Analyzing Patterns and Shapes



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

Automatic rectification

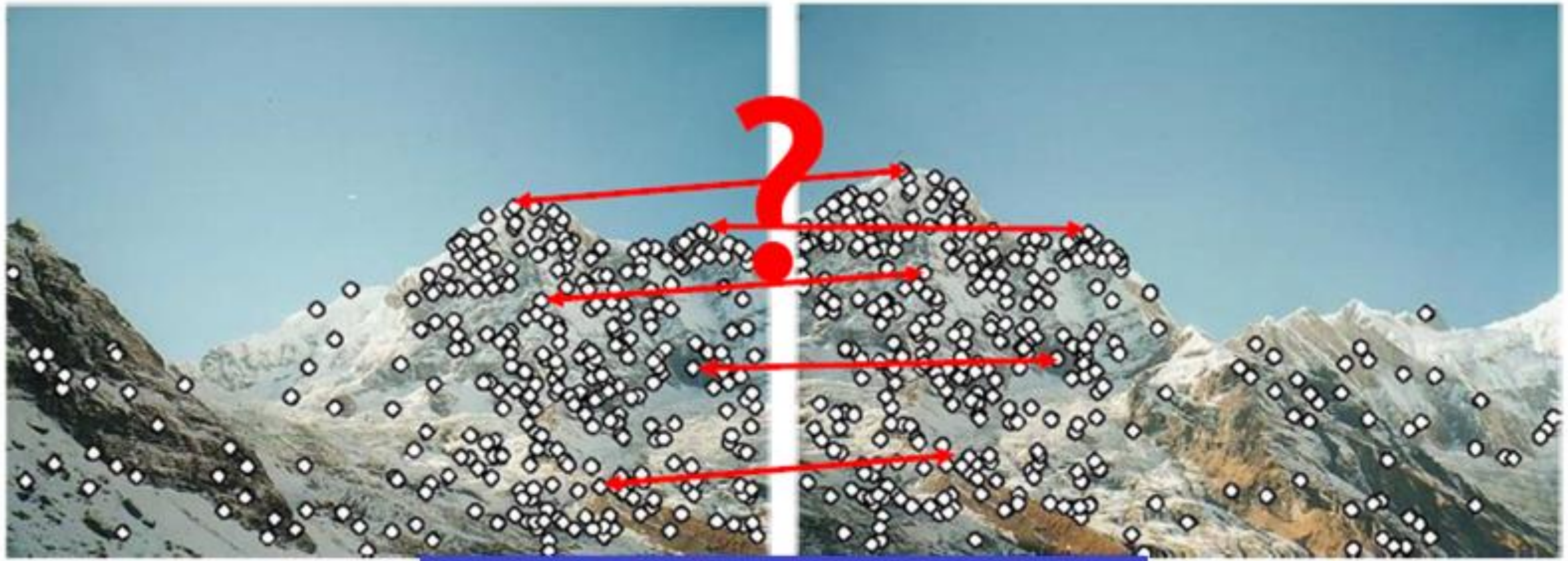
Perception VI: Descriptors. SIFT

Dr. Ilya Afanasyev, i.afanasyev@innopolis.ru

Local Descriptors

- We know how to detect points
- Next question:

How to *describe* them for matching?

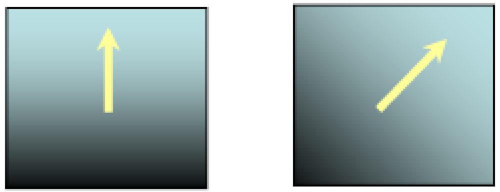


Point descriptor should be:
1. Invariant
2. Distinctive

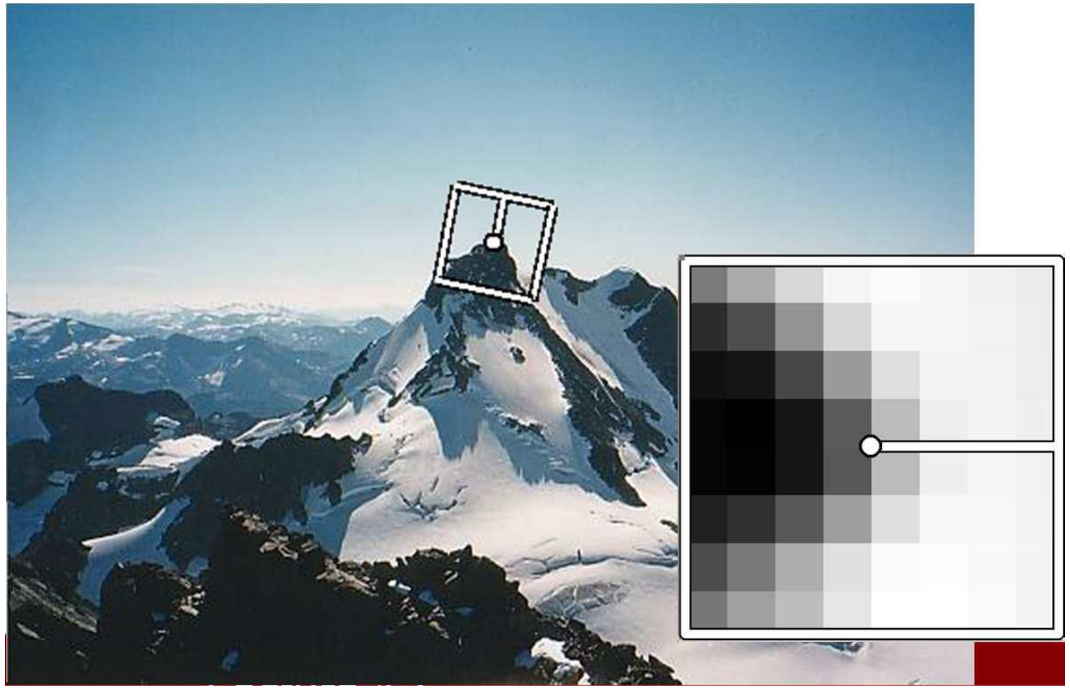
© Slide credit: Kristen Grauman

Rotation Invariant Descriptors

- Find local orientation
  - Dominant direction of gradient for the image patch

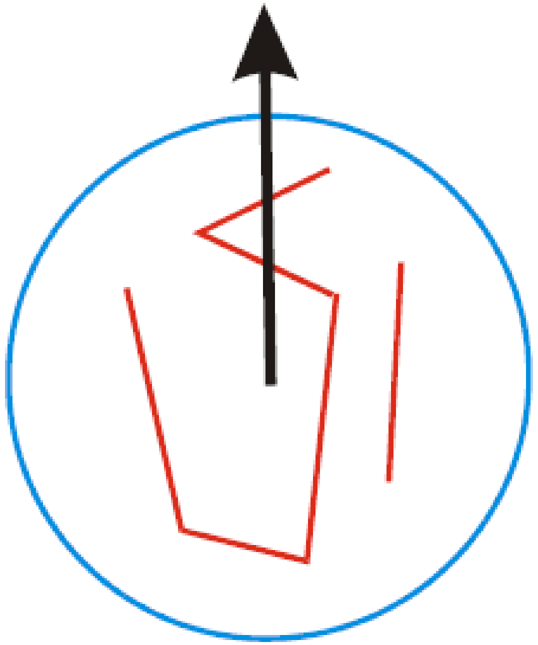


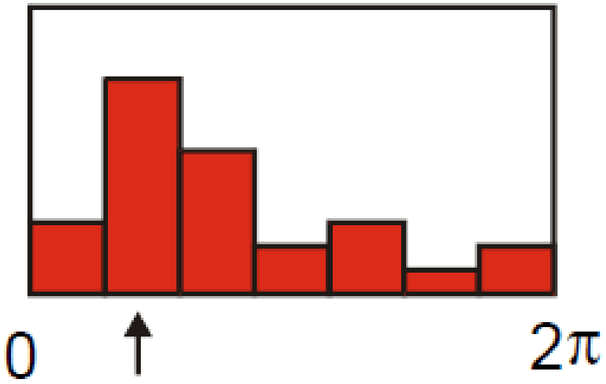
- Rotate patch according to this angle
  - This puts the patches into a canonical orientation.



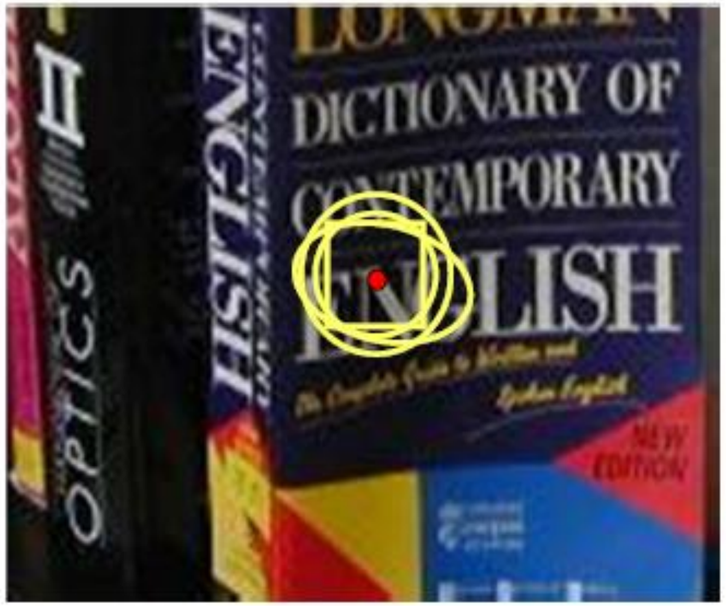
Orientation Normalization: Computation

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





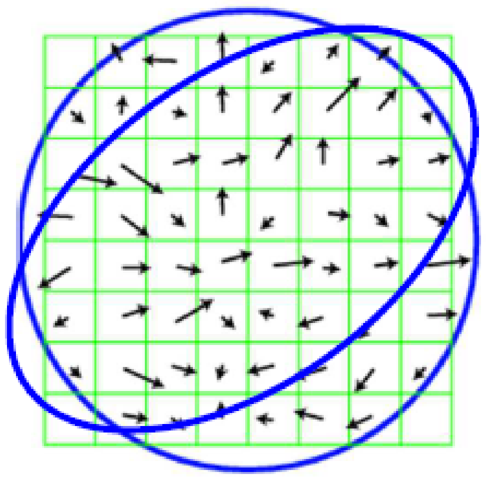
The Need for Invariance



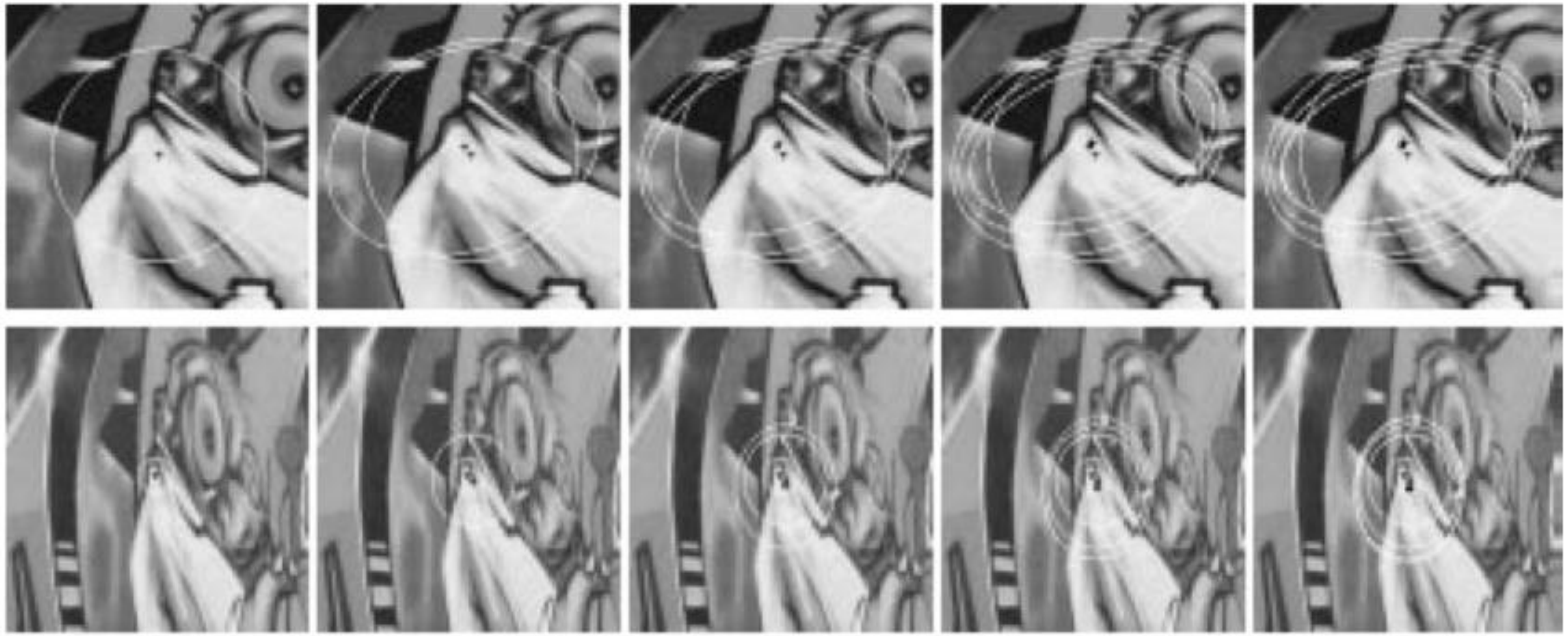
- Up to now, we had invariance to
  - Translation
  - Scale
  - Rotation
- Not sufficient to match regions under viewpoint changes
  - For this, we need also affine adaptation

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



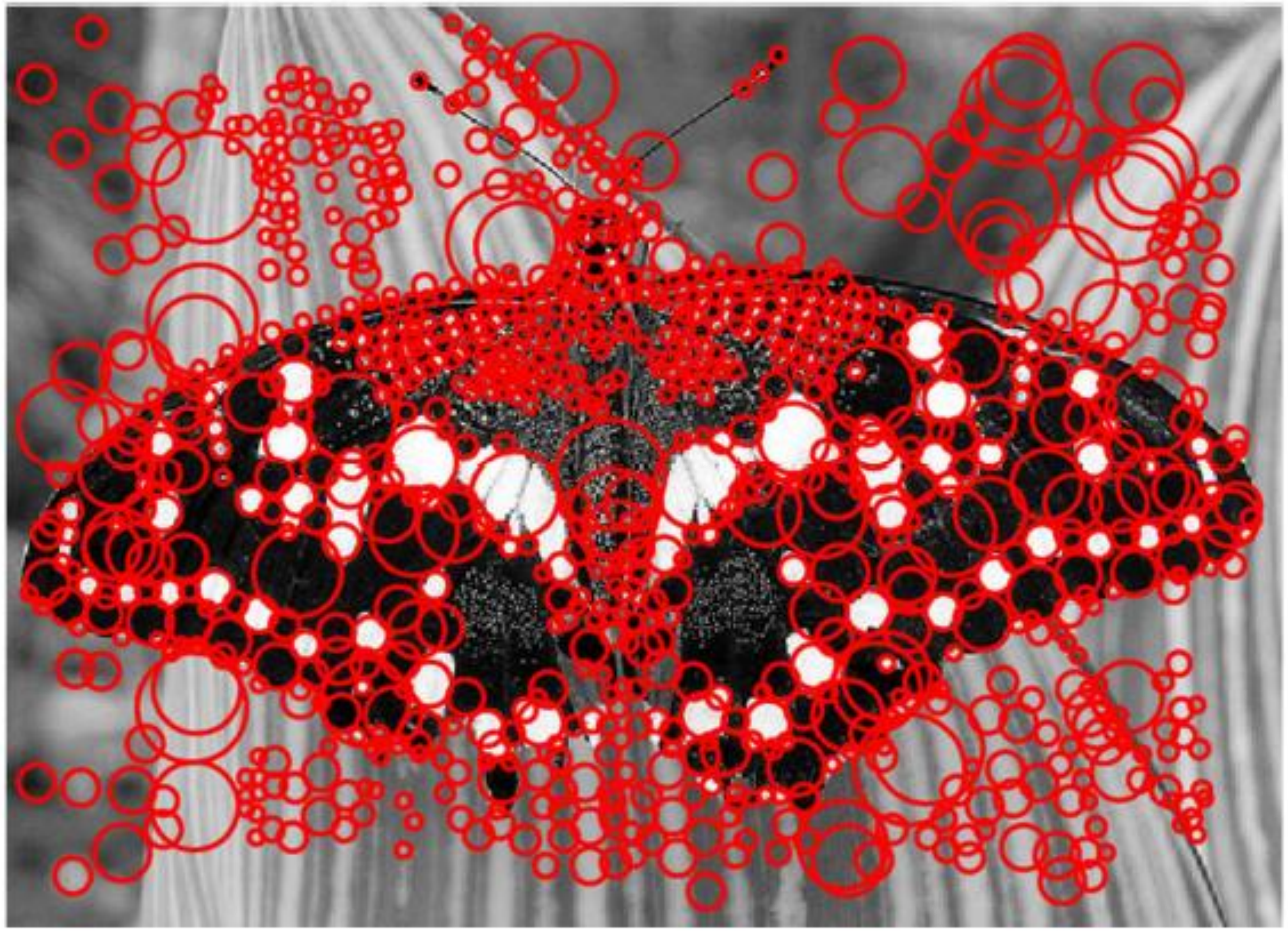
1. Detect keypoints, e.g. multi-scale Harris
2. Automatically select the scales
3. Adapt affine shape based on second order moment matrix
4. Refine point location

© Slide credit: Tinne Tuytelaars. K. Mikolajczyk and C. Schmid, Scale and affine invariant interest point detectors, IJCV 60(1):63-86, 2004.

Affine Normalization/Deskewing

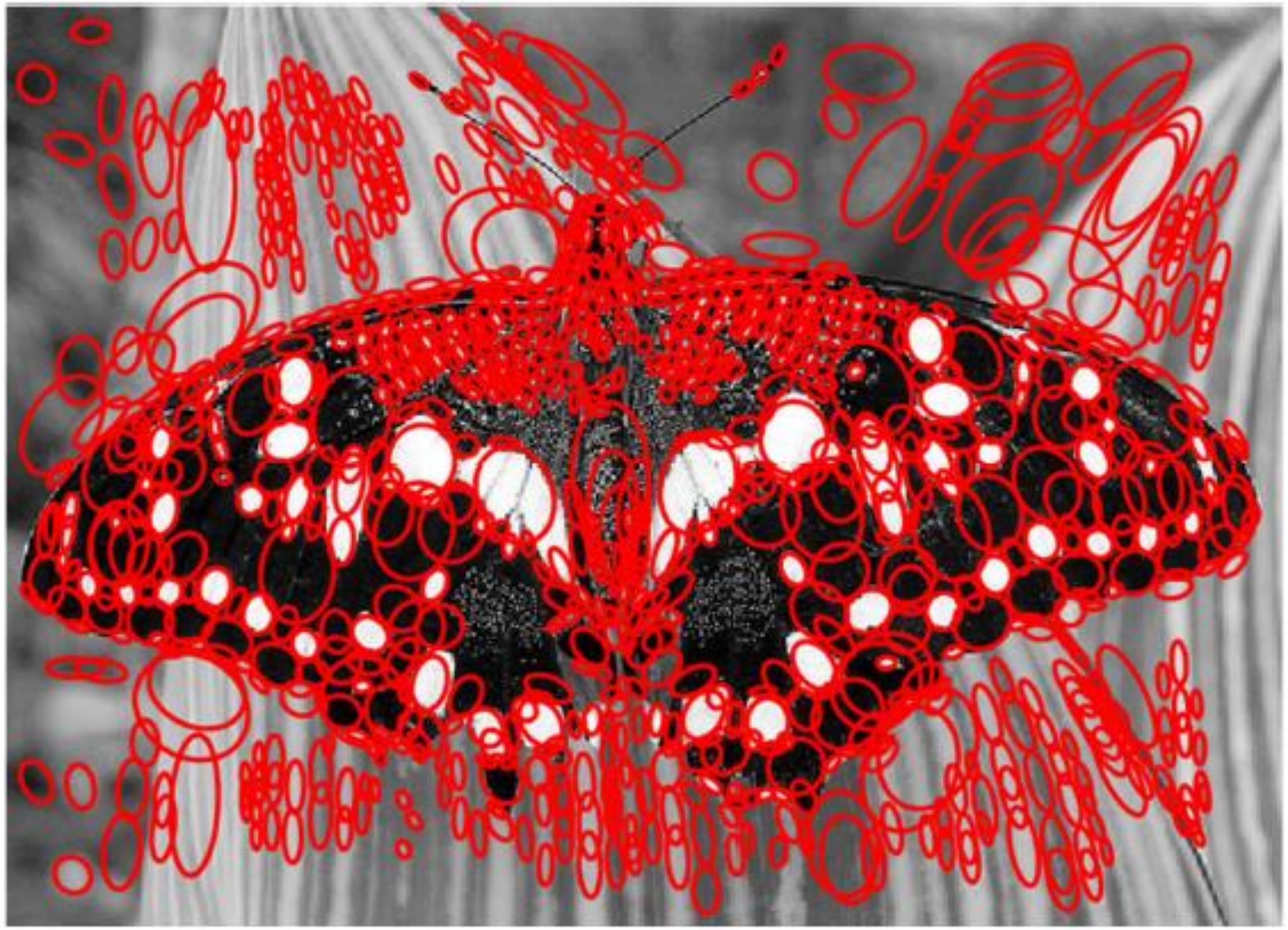
- Steps
  - Rotate the ellipse's main axis to horizontal
  - Scale the x axis, such that it forms a circle

Affine Adaptation Example



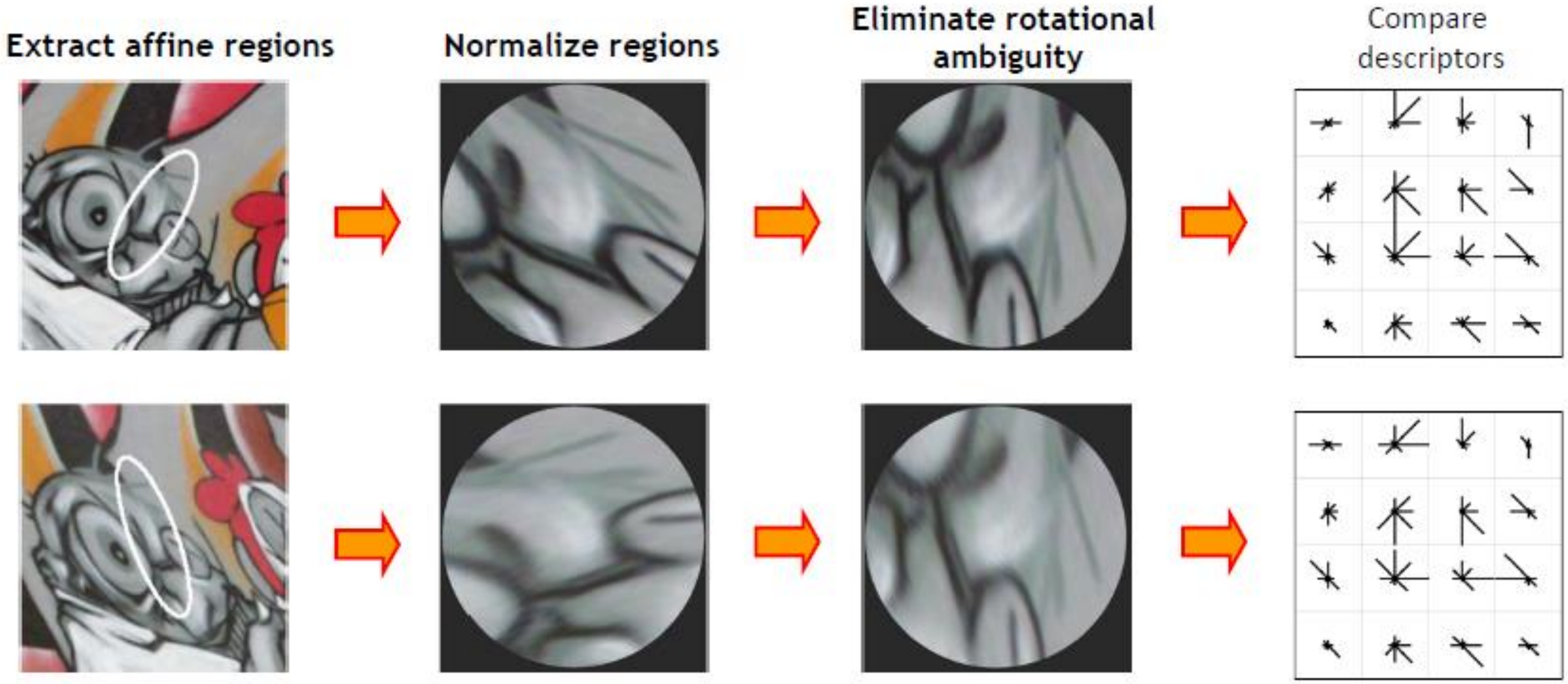
Scale-invariant regions (blobs)

Affine Adaptation Example



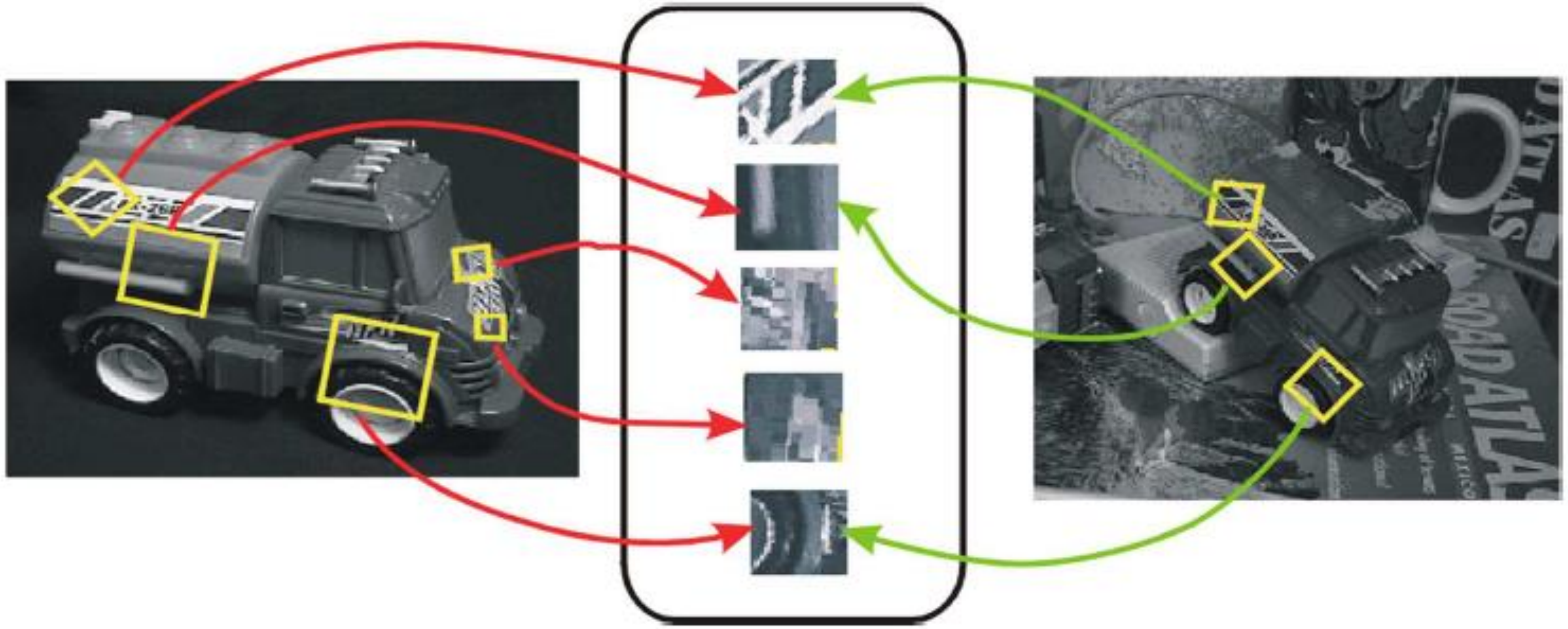
Affine-adapted blobs

Summary: Affine-Inv. Feature Extraction



Summary: Affine-Inv. Feature Extraction

- Invariance:
  - features(transform(image)) = features(image)
- Covariance:
  - features(transform(image)) = transform(features(image))

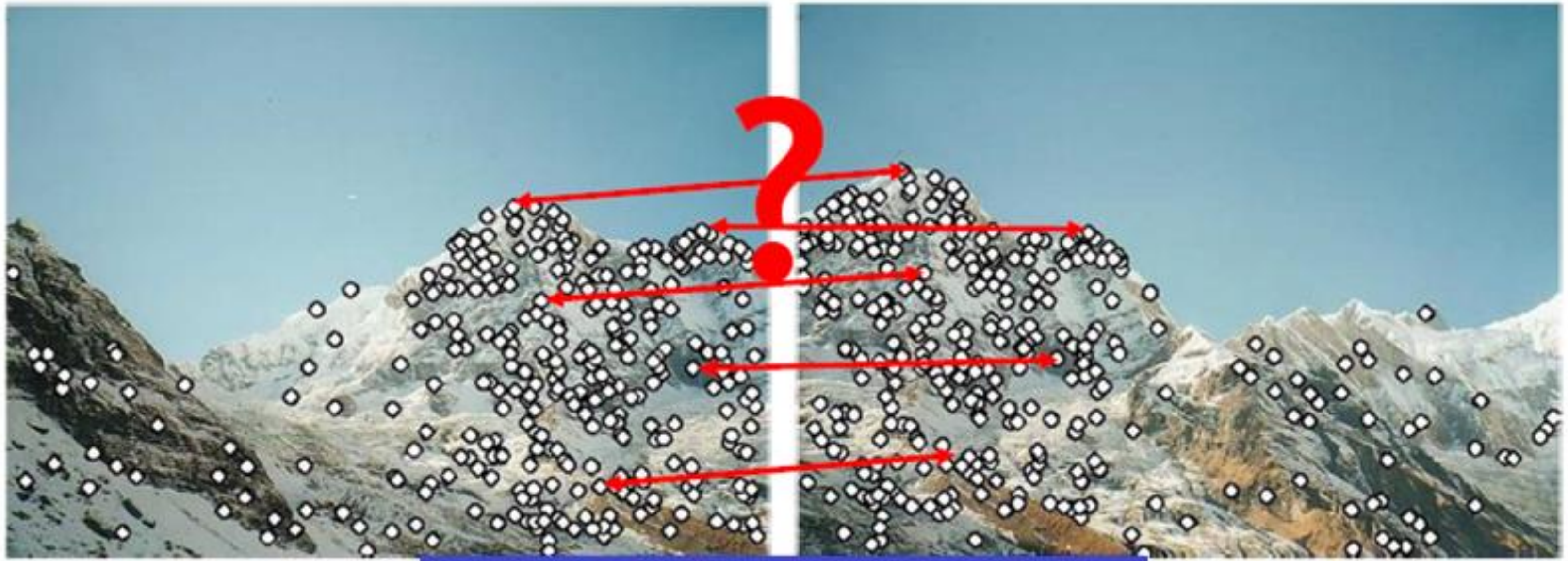


Covariant detection ⇒ invariant description

Local Descriptors

- We know how to detect points
- Next question:

How to *describe* them for matching?



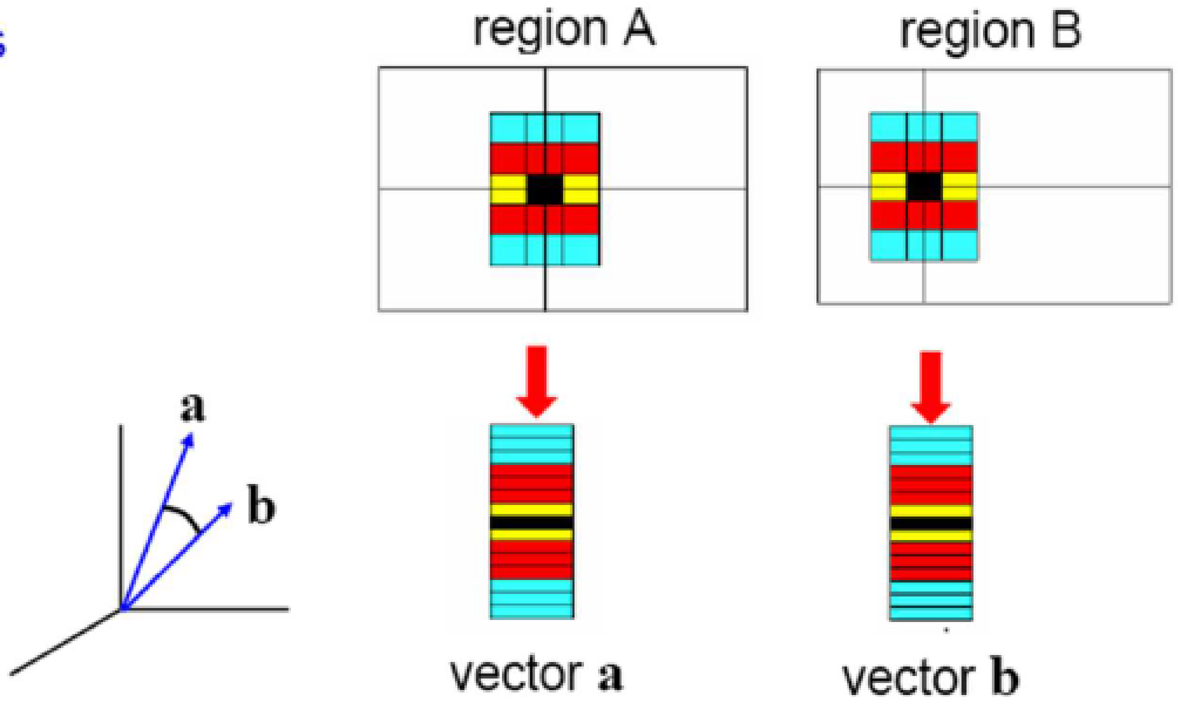
Point descriptor should be:
1. Invariant
2. Distinctive

Local Descriptors

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

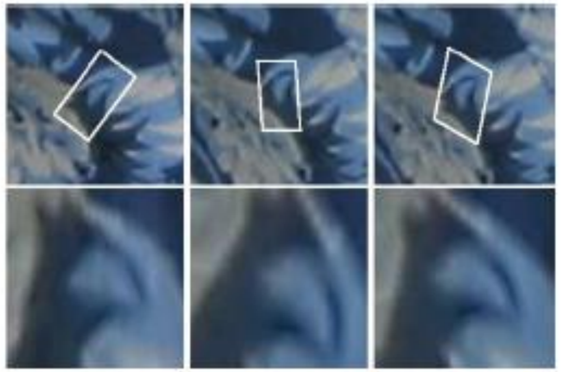
Write regions as vectors

$$\mathrm{A} \rightarrow \mathbf{a}, \;\; \mathrm{B} \rightarrow \mathbf{b}$$

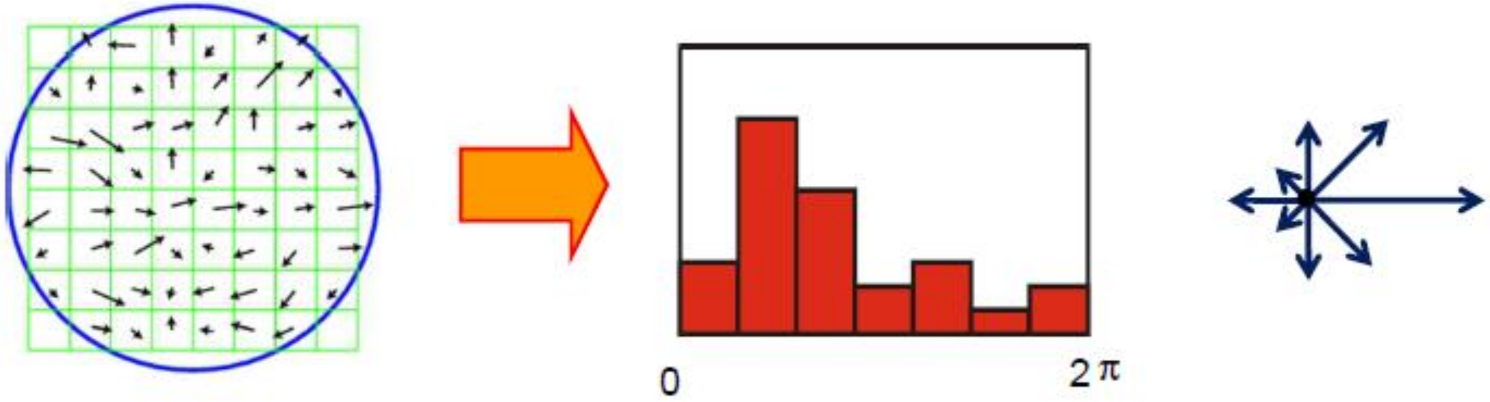


Feature Descriptors

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



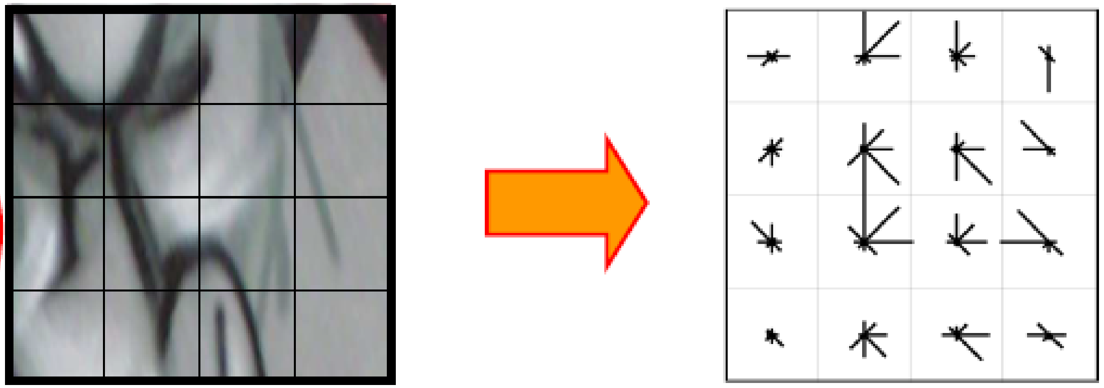
- Solution: histograms



Feature Descriptors: SIFT

- Scale Invariant Feature Transform
- Descriptor computation:
  – Divide patch into 4x4 sub-patches: 16 cells
  – Compute histogram of gradient orientations (8 reference angles) for all pixels inside each sub-patch
  – Resulting descriptor: 4x4x8 = 128 dimensions

(optimal) subtract from dominant orientation (normalize orient.)



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

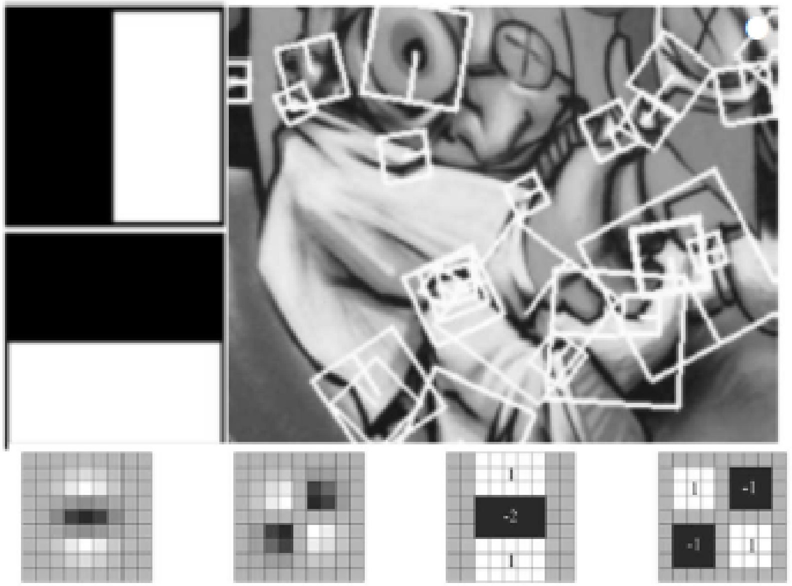
Overview: SIFT

- 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


Working with SIFT Descriptors

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

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

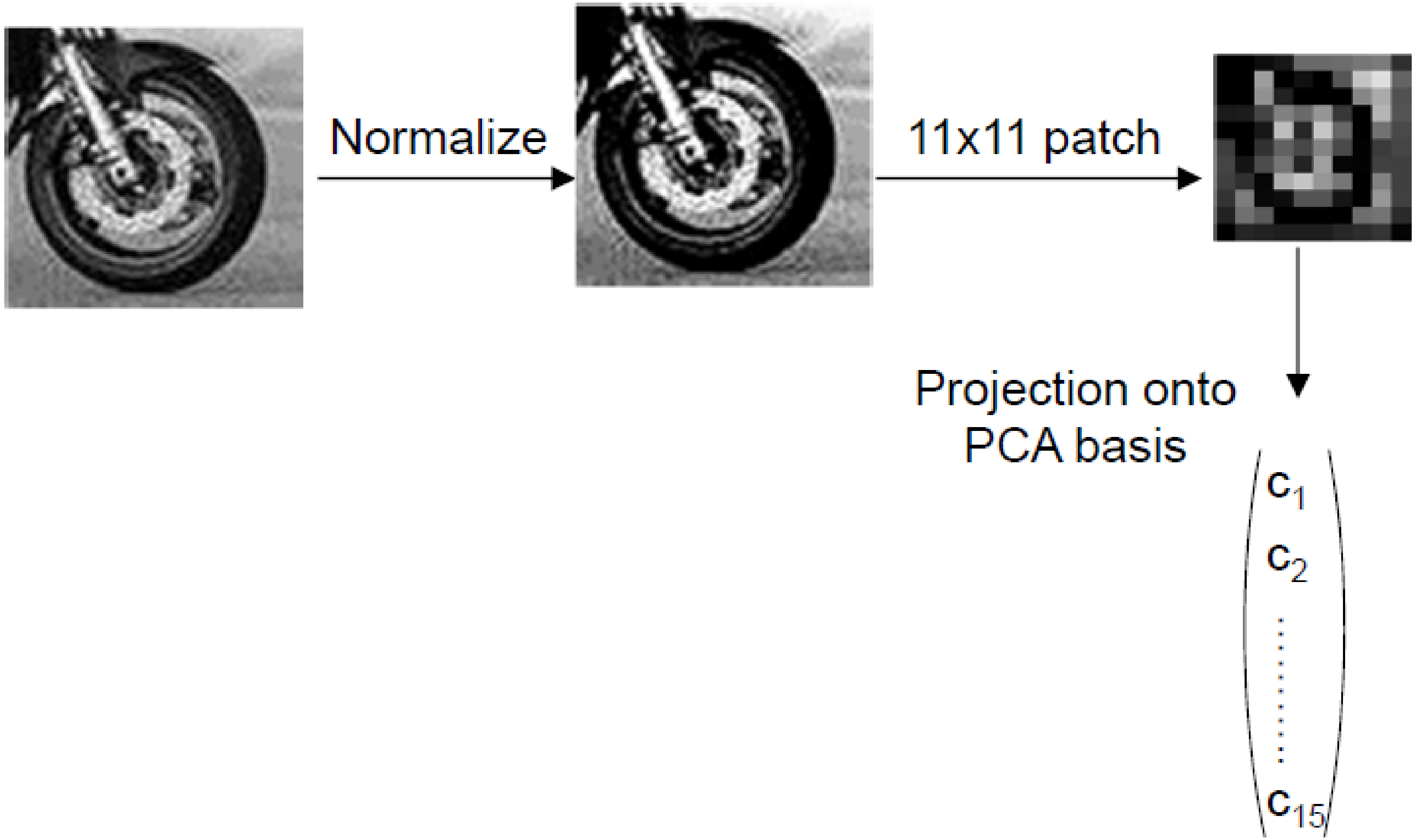
GPU implementation available

- Feature extraction @ 100Hz (detector + descriptor, 640×480 img)
- http://homes.esat.kuleuven.be/~ncorneli/gpusurf/

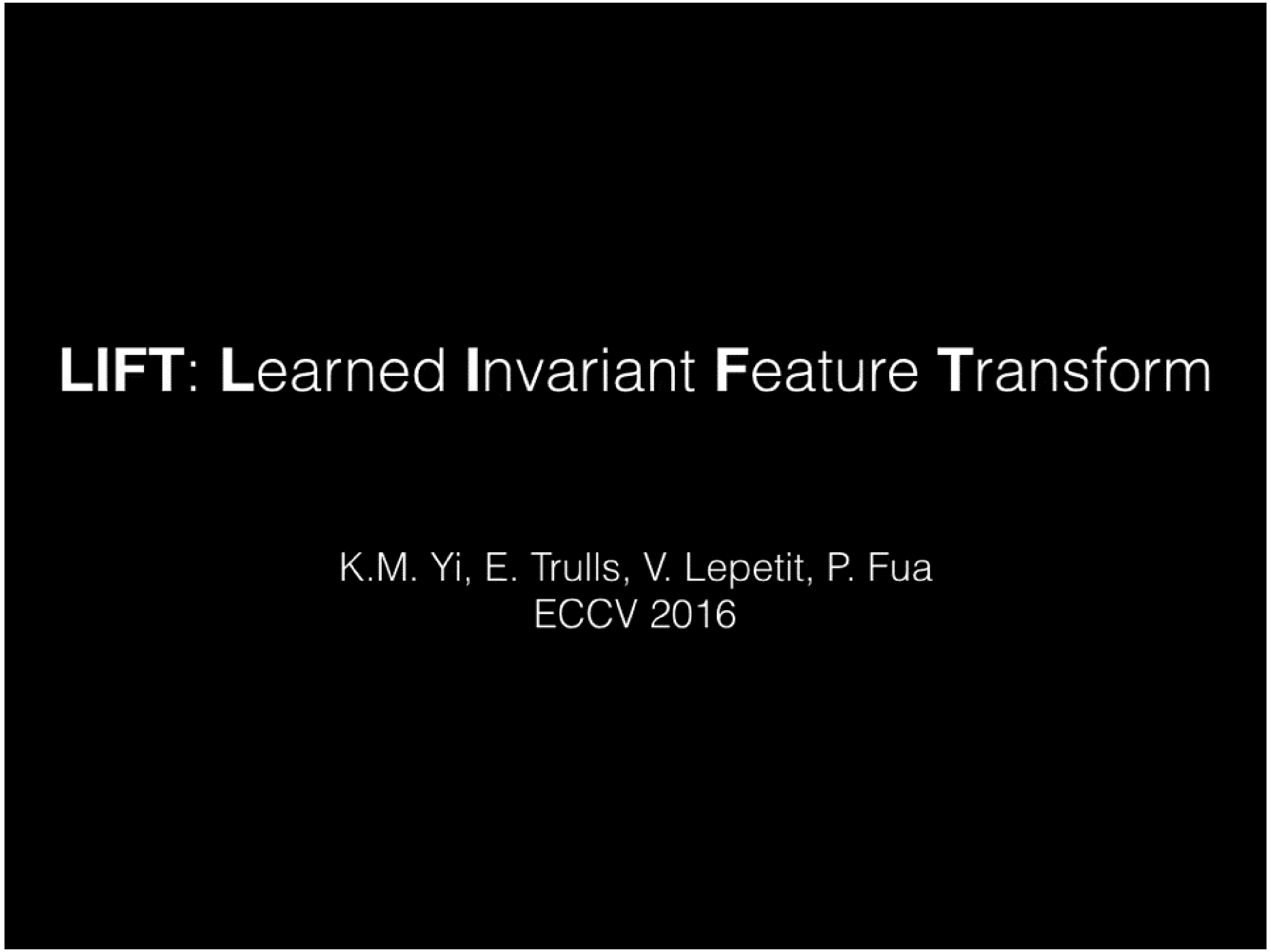
http://www.vision.ee.ethz.ch/~surf

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

Other local descriptors: Gray-scale intensity



Matching features: Learned Invariant Feature Transform (LIFT)



References

1) Roland Siegwart, Illah R. Nourbakhsh, and Davide Scaramuzza. Introduction to Autonomous Mobile Robots, MIT press, 2011
2) H. Choset, K. M. Lynch, et. al. “Principles of Robot Motion: Theory, Algorithms, and Implementations”, MIT press, 2005
3) Gregory Dudek and Michael Jenkin. Computational Principles of Mobile Robotics, 2nd ed., Cambridge University Press, 2010.
4) Alonzo Kelly. Mobile Robotics: Mathematics, Models, and Methods. Cambridge University Press, 2013
5) Horn, Berthold K. P. Robot Vision. Cambridge, MA: MIT Press /McGraw-Hill, 1986.
6) M. Chli, D. Scaramuzza, R. Siegwart, et al. Autonomous Mobile Robots, ETH Zurich, http://www.asl.ethz.ch/education/lectures/autonomous_mobile_robots.html
7) Fei-Fei Li, Stanford Vision Lab., CS231A, 2011 http://vision.stanford.edu/teaching/cs231a_autumn1112/

THANK YOU!

i.afanasyev@innopolis.ru