Correspondence
Outline

- Motivation
- Interest point detection
- Descriptors
Core visual understanding task: finding correspondences between images
Example: image matching of landmarks

Correspondence + geometry estimation
Object recognition by matching

Sparse correspondence

Dense correspondence
Example: license plate recognition
Example: product recognition

Google Glass
Abstract

This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of distortion, change in 3D viewpoint, addition of noise, and change in illumination.

The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from many images. This paper also describes an approach to using these features for object recognition. The recognition proceeds by matching individual features to a database of features from known objects using a fast nearest-neighbor algorithm, followed by a Hough transform to identify clusters belonging to a single object, and finally performing verification through least-squares solution for consistent pose parameters. This approach to recognition can robustly identify objects among clutter and occlusion while achieving near real-time performance.

Accepted for publication in the International Journal of Computer Vision, 2004.

IJCV 04
Motivation

Which of these patches are easier to match?

Why? How can we mathematically operationalize this?
Corner Detector: Basic Idea

“flat” region: no change in any direction

“edge”: no change along the edge direction

“corner”: significant change in all directions

Defn: points are “matchable” if small shifts always produce a large SSD error
The math

Defn: points are “matchable” if small shifts always produce a large SSD error

\[
\text{cornerness}(x_0, y_0) = \min_{u,v} E_{x_0,y_0}(u, v)
\]

where

\[
E_{x_0,y_0}(u, v) = \sum_{(x,y) \in W(x_0,y_0)} [I(x + u, y + v) - I(x, y)]^2
\]

Why can’t this be right?
The math

Defn: points are “matchable” if small shifts always produce a large SSD error

$$\text{cornerness}(x_0, y_0) = \min_{u, v} E_{x_0, y_0}(u, v)$$

where

$$E_{x_0, y_0}(u, v) = \sum_{(x, y) \in W(x_0, y_0)} [I(x + u, y + v) - I(x, y)]^2$$

$$u^2 + v^2 = 1$$
General mathematical tool: nonlinear least squares

\[
\text{cornerness}(x_0, y_0) = \min_{u^2 + v^2 = 1} E_{x_0, y_0}(u, v)
\]

where

\[
E_{x_0, y_0}(u, v) = \sum_{(x, y) \in W(x_0, y_0)} [I(x + u, y + v) - I(x, y)]^2
\]

https://en.wikipedia.org/wiki/Non-linear_least_squares

We’ll apply a “standard technique”: Gauss-Newton optimization
Background: Taylor series expansion

\[ f(x + u) = f(x) + \frac{\partial f(x)}{\partial x}u + \frac{1}{2} \frac{\partial f(x)}{\partial xx}u^2 + \text{Higher Order Terms} \]

Approximation of \( f(x) = e^x \) at \( x=0 \)

Why are low-order expansions reasonable?
Underlying smoothness of real-world signals
Multivariate Taylor series

\[ I(x + u, y + v) = I(x, y) + \left[ \frac{\partial I(x, y)}{\partial x} \; \frac{\partial I(x, y)}{\partial y} \right] \begin{bmatrix} u \\ v \end{bmatrix} + \]

\[ \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} \frac{\partial I(x, y)}{\partial xx} & \frac{\partial I(x, y)}{\partial xy} \\ \frac{\partial I(x, y)}{\partial xy} & \frac{\partial I(x, y)}{\partial yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \text{Higher Order Terms} \]

what’s this vector called?

what’s this matrix called?

\[ I(x + u, y + v) \approx I + I_x u + I_y v \]

where

\[ I_x = \frac{\partial I(x, y)}{\partial x} \]
Multivariate taylor series

\[ I(x + u, y + v) = I(x, y) + \left[ \frac{\partial I(x, y)}{\partial x} \quad \frac{\partial I(x, y)}{\partial y} \right] \begin{bmatrix} u \\ v \end{bmatrix} + \text{gradient} \]

\[
\frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} \frac{\partial I(x, y)}{\partial xx} & \frac{\partial I(x, y)}{\partial xy} \\ \frac{\partial I(x, y)}{\partial xy} & \frac{\partial I(x, y)}{\partial yy} \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} + \text{Higher Order Terms}
\]

\[ I(x + u, y + v) \approx I + I_x u + I_y v \]

where

\[ I_x = \frac{\partial I(x, y)}{\partial x} \]
Consider shifting the window $W$ by $(u,v)$

- how do the pixels in $W$ change?
- compare each pixel before and after by summing up the squared differences
- this defines an “error” of $E(u,v)$:

$$E(u, v) = \sum_{(x,y) \in W} \left[ I(x + u, y + u) - I(x, y) \right]^2$$

$$\approx \sum_{(x,y) \in W} \left[ I + I_x u + I_y v - I \right]^2$$

$$= \sum_{(x,y) \in W} \left[ I_x^2 u^2 + I_y^2 v^2 + 2I_x I_y uv \right]$$

$$= [u \ v] \ A \ [u \ v]^T, \quad A = \sum_{(x,y) \in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix}$$
Defn: points are “matchable” if small shifts always produce a large SSD error

\[ \text{Corner}(x_0, y_0) = \min_{u^2 + v^2 = 1} E(u, v) \]

where

\[ E(u, v) = \begin{bmatrix} u & v \end{bmatrix} A \begin{bmatrix} u \\ v \end{bmatrix}, \quad A = \sum_{(x, y) \in W(x_0, y_0)} \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} \]

Claim 1: ‘A’ is symmetric (\( A^T = A \)) and PSD

Claim 2: Corner-ness is given by min eigenvalue of ‘A’

Question: Is ‘A’ a Hessian matrix?
Recall: spectral decompositions

\[ A = V \Lambda V^T \]

Defn: a symmetric matrix A is PSD if \( x^T A x \geq 0 \) for all \( x \)

A is PSD \( \iff \) eigenvalues are all positive
A is PSD \( \iff \) \( A = XX^T \), where \( X = [\sqrt{\lambda_1}v_1 \ \sqrt{\lambda_2}v_2 \ldots] = \Lambda^{\frac{1}{2}}V \)

\[ X = U \Sigma V^T \rightarrow A = XX^T = U \Sigma^2 U^T \]

Eigenvectors of \( A \) = left singular vectors of \( X \)
Eigenvalues of \( A \) = squared singular values of \( X \)
Aside: turns out spectral decomposition holds for any symmetric matrix

\[ A = V \Lambda V^T \]

\[ \Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \ldots \\ 0 & \lambda_2 & 0 & \ldots \\ 0 & 0 & \lambda_3 & \ldots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix} \quad V = [v_1, v_2, \ldots, v_n] \quad V^T V = I \]

In the general case, eigenvalues can be negative
Alternative visualization of PSD matrices

$$A = V \Lambda V^T$$

Consider set of \((x_1,x_2)\) points for which:

$$\begin{bmatrix} x_1 & x_2 \end{bmatrix} A \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 1$$

- \(A = I\)
  - \(x_1^2 + x_2^2 = 1 \rightarrow (1, 0)(0, 1)\)
- \(A = \Lambda\)
  - \(4x_1^2 + x_2^2 = 1 \rightarrow (.5, 0)(0, 1)\)
- \(A = V \Lambda V^T\)
Defn: points are “matchable” if small shifts always produce a large SSD error

Corner\((x_0, y_0)\) = \(\min_{u^2 + v^2 = 1} E(u, v)\)

where

\[ E(u, v) = \begin{bmatrix} u & v \end{bmatrix} A \begin{bmatrix} u \\ v \end{bmatrix}, \quad A = \sum_{(x,y) \in W(x_0,y_0)} \begin{bmatrix} I_x^2 & I_x I_y \\ I_y I_x & I_y^2 \end{bmatrix} \]

Solution is given by minimum eigenvalue

Implies \((x_0, y_0)\) is a good corner if minimum eigenvalue is large

(or alternatively, if both eigenvalues of ‘A’ are large)
What will eigenvalues (and eigenvectors) look like?

let’s think about ‘A’ matrix…
Classification of image points using eigenvalues of $A$:

- **Corner**
  - $\lambda_1$ and $\lambda_2$ are large,
  - $\lambda_1 \sim \lambda_2$;
  - $E$ increases in all directions

- **Edge**
  - $\lambda_2 >> \lambda_1$

- **Flat** region
  - $\lambda_1$ and $\lambda_2$ are small;
  - $E$ is almost constant in all directions

- **Edge**
  - $\lambda_1 >> \lambda_2$
Efficient computation

Computing eigenvalues (and eigenvectors) is expensive.

Turns out that it’s easy to compute their sum (trace) and product (determinant)

- $\text{Det}(A) = \lambda_{\text{min}} \lambda_{\text{max}}$
- $\text{Trace}(A) = \lambda_{\text{min}} + \lambda_{\text{max}}$ (trace = sum of diagonal entries)

$$R = 4 \frac{\text{Det}(A)}{\text{Trace}(A)^2}$$ (is proportional to the ratio of eigvenvalues and is 1 if they are equal)

$$R = \text{Det}(A) - \alpha \text{Trace}(A)^2$$ (also favors large eigenvalues)
Harris detector example
corner value (red high, blue low)

Question: can we compute these heat maps with convolutions?
Threshold ($f > \text{value}$)
Harris features (in red)

The tops of the horns are detected in both images
Scale and rotation invariance

Will interest point detector still fire on rotated & scaled images?
Rotation invariance (?)

Are eigenvector stable under rotations? No
Are eigenvalues stable under rotations? Yes
Are eigenvector stable under scalings? Yes
Are eigenvalues stable under scalings? No
A solution to scale

search over image pyramid scales

\[ A(x, y, \sigma) = \sum_{x, y} \begin{bmatrix} I_x(\sigma)^2 & I_x I_y(\sigma) \\ I_y I_x(\sigma) & I_y^2(\sigma) \end{bmatrix} \]
A solution to scale

\[
\text{cornerness}(x, y, \sigma) = \det(A(x, y, \sigma)) - \alpha \text{Trace}^2(A(x, y, \sigma))
\]

Look for local maxima in (x,y,sigma)
Annoying “details”

1. Positions across scales don’t align

Soln: construct blurred versions of image

2. Gradients across scales aren’t comparable (gradients always smaller on blurred images)

Soln: multiply gradients by scale factor

Scale-space theory: A basic tool for analysing structures at different scales

Tony Lindeberg

Computational Vision and Active Perception Laboratory (CVAP)  
Department of Numerical Analysis and Computing Science
Putting it all together: Harris-Laplacian detector

\[ A(x, y, \sigma_I, \sigma_d) = \sigma_D^2 G(\sigma_I) \ast \begin{bmatrix} I_x(\sigma_D)^2 & I_x I_y(\sigma_D) \\ I_y I_x(\sigma_D) & I_y^2(\sigma_D) \end{bmatrix} \]

Relate Gaussian for integration with Gaussian for computing derivatives

Heuristic: \( \sigma_D = \cdot 7 \sigma_I \)

https://en.wikipedia.org/wiki/Harris_affine_region_detector
“Sub-pixel” accuracy across sigma (and $x,y$)

1. Optimize cornerness($x,y,sigma$) over discrete set of locations and scales

2. Fine-tune “sub-pixel” accuracy by iterating the following:
   
   i. Given $(x,y)$, we can find maximal sigma with finer search
   
   ii. Given sigma, find maximal $(x,y)$ of cornerness

Repeat (i,ii) over local neighborhoods of $(sigma,x,y)$ until convergence
Scale selection in 2D

Lindeberg et al., 1996

Function responses for increasing scale. Scale trace (signature).
Scale selection in 2D

Function responses for increasing scale
Scale trace (signature)
Scale selection in 2D
Scale selection in 2D

Function responses for increasing scale
Scale trace (signature)
Scale selection in 2D

Function responses for increasing scale
Scale trace (signature)
Scale selection in 2D

Function responses for increasing scale
Scale trace (signature)

\[ f(I_{\lambda_{\text{max}}} (\lambda, \sigma)) \]
Scale selection in 2D

Function responses for increasing scale
Scale trace (signature)
Extension 1: anisotropic scale

Need richer description of “neighborhood” or scale

Replace scalar $\sigma$ with $\sum$

(e.g., scale differently long x and y, or even a diagonal axis)

1. Optimize cornerness($x,y,\text{sigma}$) over discrete set of locations and scales

2. Fine-tune “sub-pixel” accuracy by iterating the following:
   i. Given $(x,y)$, find maximal $\Sigma$ with local search
   ii. Given $\Sigma$, find maximal $(x,y)$ of cornerness
Affine Invariance
Application: Finding correspondences
Final matches: 32 correct correspondences
Scale: 4.9
Rotation: 19°

Example from Mikolajczyk and Schmid 2004
Extension 2: directly work with scale-space features or “blobs”


\[ D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) \ast I(x, y) \]

\[ k = 2^{\frac{1}{s}} \text{ where } s = \# \text{ levels in an octave} \]
Look for “blob detections” that are
locally maximal, high confidence, and localizeable

Local maxima of $D(x,y,\sigma)$

$D(x,y,\sigma) > \text{thresh}$

$\begin{bmatrix}
D_{xx} & D_{xy} \\
D_{yx} & D_{yy}
\end{bmatrix}$

min eigenvalue of Hessian $> \text{thresh}$

Added benefit of Hessian: use second-order taylor expansion to get “subpixel” accuracy

Alternative approach for rotation invariance

(Lowe, SIFT)

Compute gradients for all pixels in patch. Histogram (bin) gradients by orientation

(I prefer this because you can look for multiple peaks)
Comparison

# correspondences
---
# possible correspondences
(points present)

![Graph showing comparison of repeatability rates for different methods: Harris-Laplacian, SIFT (Lowe), and Harris. The x-axis represents scale, and the y-axis represents repeatability rate. The graph indicates that Harris-Laplacian and SIFT (Lowe) perform better than Harris at lower scales.]
References


Software can be downloaded from Schmid’s and Lowe’s pages

Coordinate frames

Represent each patch in a canonical scale and orientation (or general affine coordinate frame)

\[ d(p_1, p_2) = \left\| \begin{bmatrix} \cdot \\ \cdot \end{bmatrix} - \begin{bmatrix} \cdot \\ \cdot \end{bmatrix} \right\| \]
Scale Invariant Feature Transform

Basic idea:

- Take 16x16 square window around detected feature
- Compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude)
- Create histogram of surviving edge orientations

Adapted from slide by David Lowe
SIFT descriptor

Full version

• Divide the 16x16 window into a 4x4 grid of cells (2x2 case shown below)
• Compute an orientation histogram for each cell
• 16 cells * 8 orientations = 128 dimensional descriptor

Adapted from slide by David Lowe
Properties of SIFT

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 60 degree 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
We’ll discuss many more on Thursday!

http://www.vlfeat.org/overview/sift.html