Features detectors and descriptors

Yann GAVET, Johan DEBAYLE

Ecole Nationale Supérieure des Mines de Saint-Etienne
Laboratoire Georges Friedel, UMR CNRS 5307

Outline

Section 1

Introduction: objectives

Applications

- Recognition, classification, database retrieval
- Image registration
- Scale detection
- Motion estimation, tracking
- Video stabilization
- Stereovision / 3D reconstruction

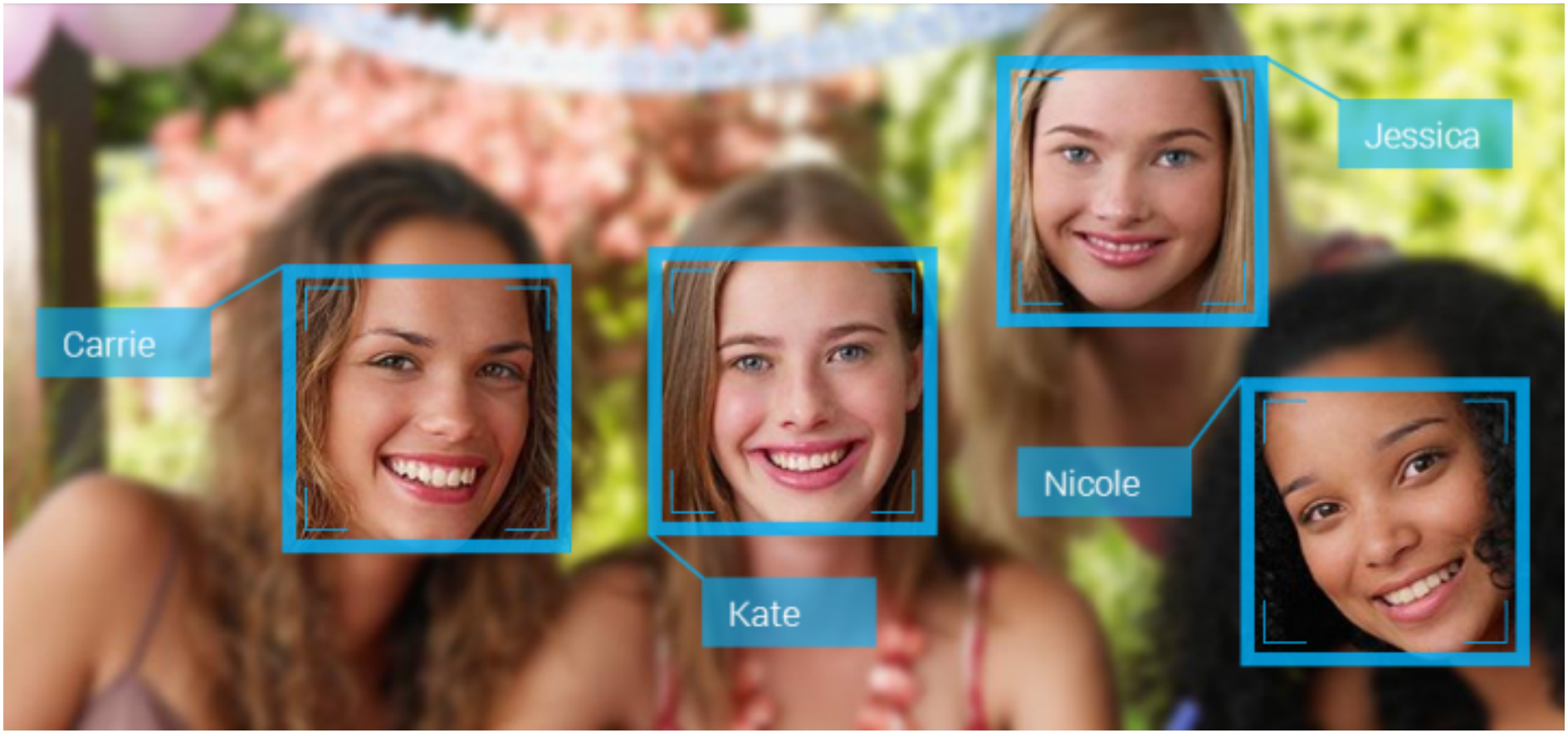
Summary

- Detect keypoints
- Look at neighborhood
- Try to find similarities
- Deduce movement

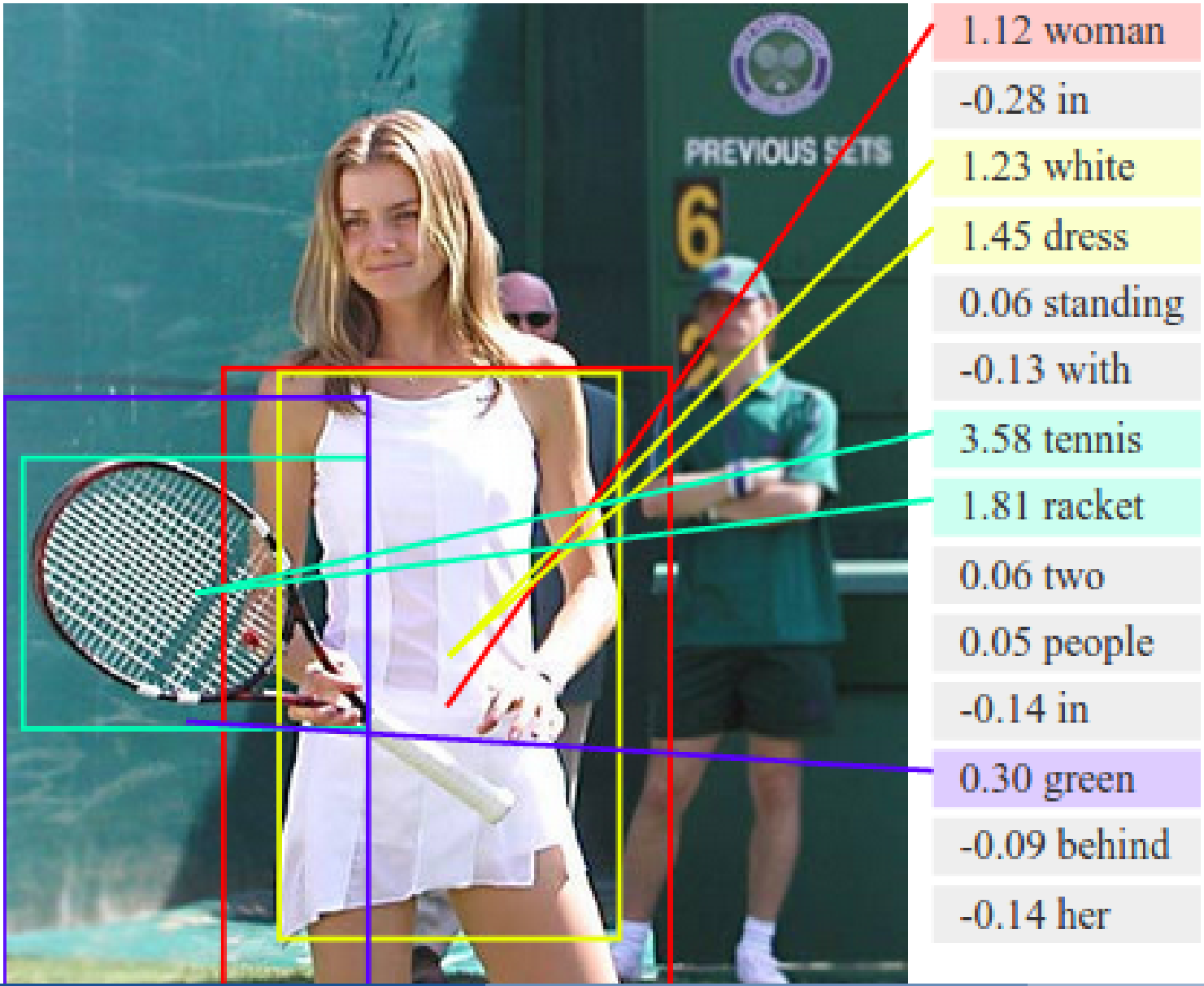
Tracking and motion estimation

Tracking and motion estimation

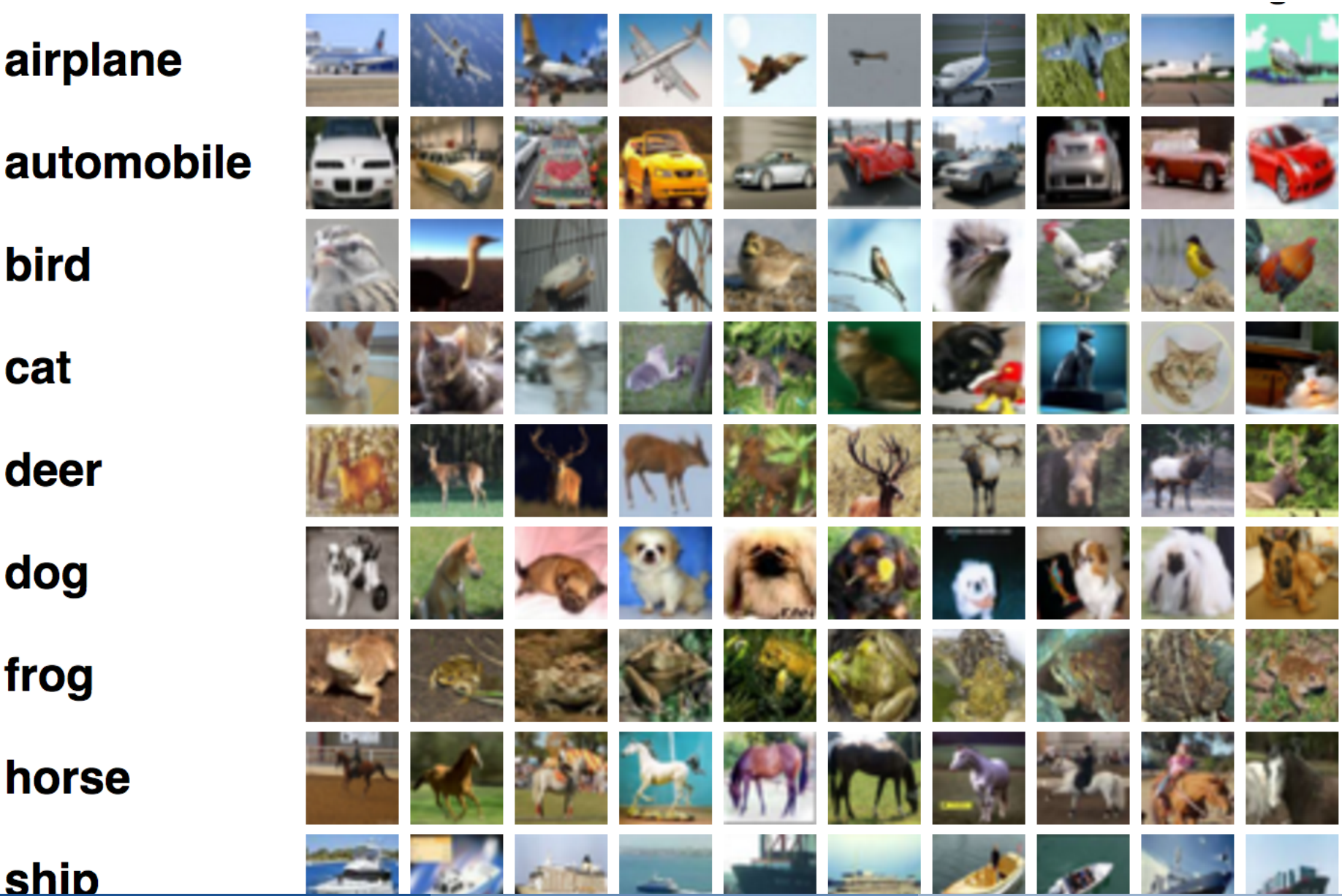
(Face) recognition



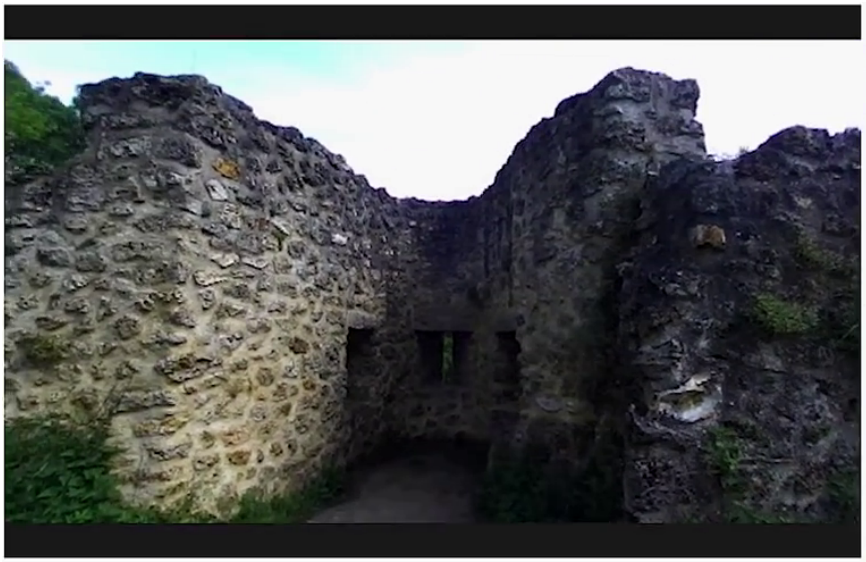
Classification

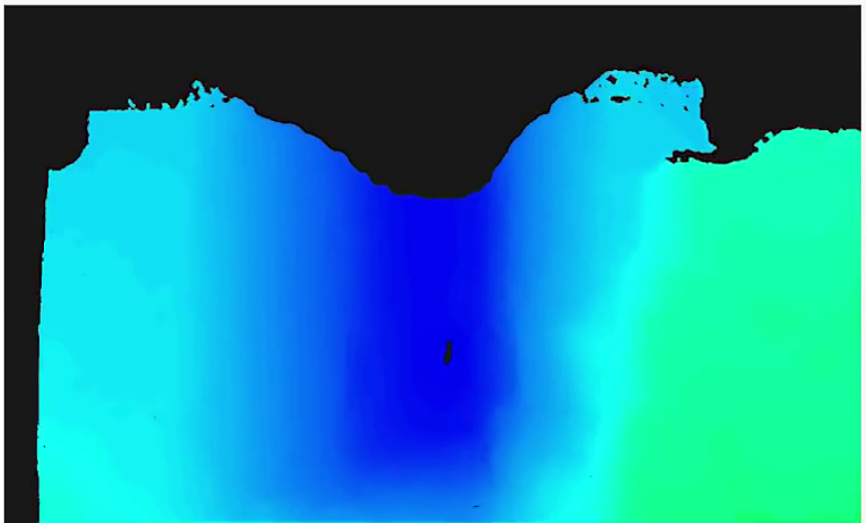


Classification

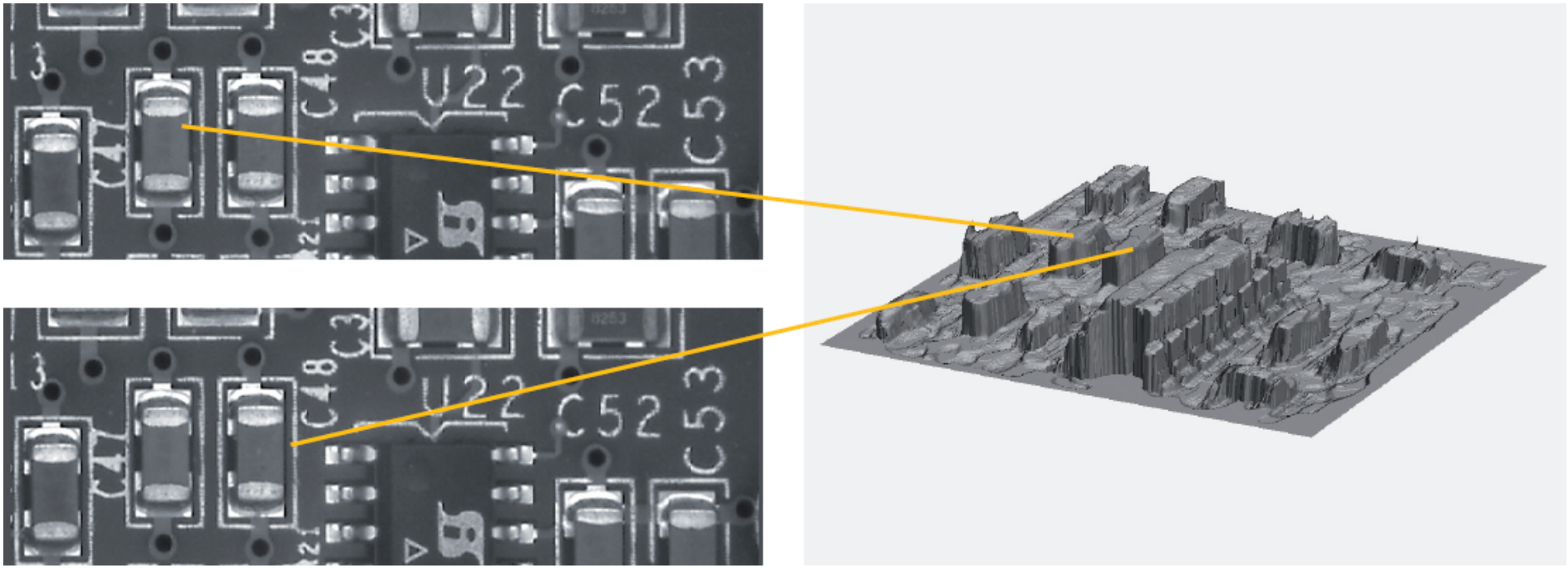


Stereovision, 3D reconstruction



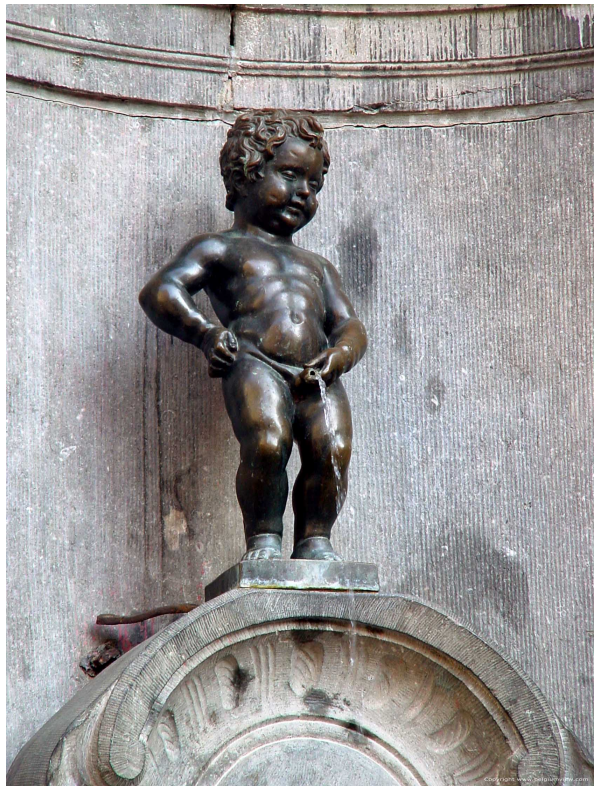


Stereovision, 3D reconstruction



Matching: difficulties

- Illumination, color changes
- orientation, viewpoint
- scale
- occlusion...



Constructing alignment

- salient points detection
- corresponding pairs
- alignment

Problems

- detect the same points
- make the correspondance
- rotation, scale, geometry

Constructing alignment

- salient points detection
- corresponding pairs
- alignment

Problems

- detect the same points
- make the correspondance
- rotation, scale, geometry

Method

1. Feature points detection
2. Feature descriptors
3. Matching two sets of points
4. Transformation evaluation

Objective: robustness

Ensure invariances by:

- rotation, translation, scale
- illumination, color changes
- viewpoint change

Automatic scale detection

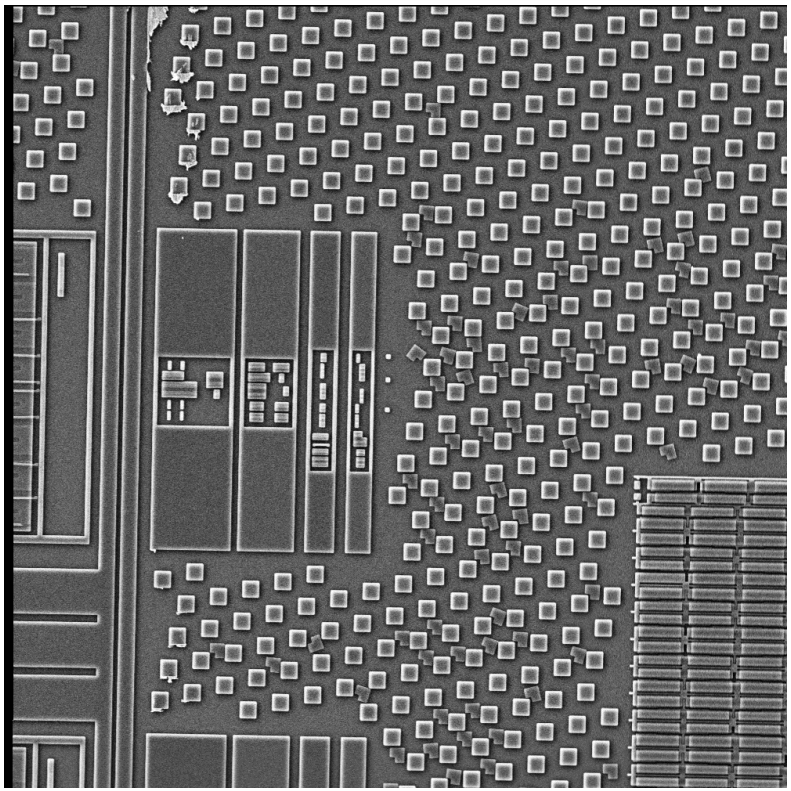
Detect scale between two different images of the same scene.

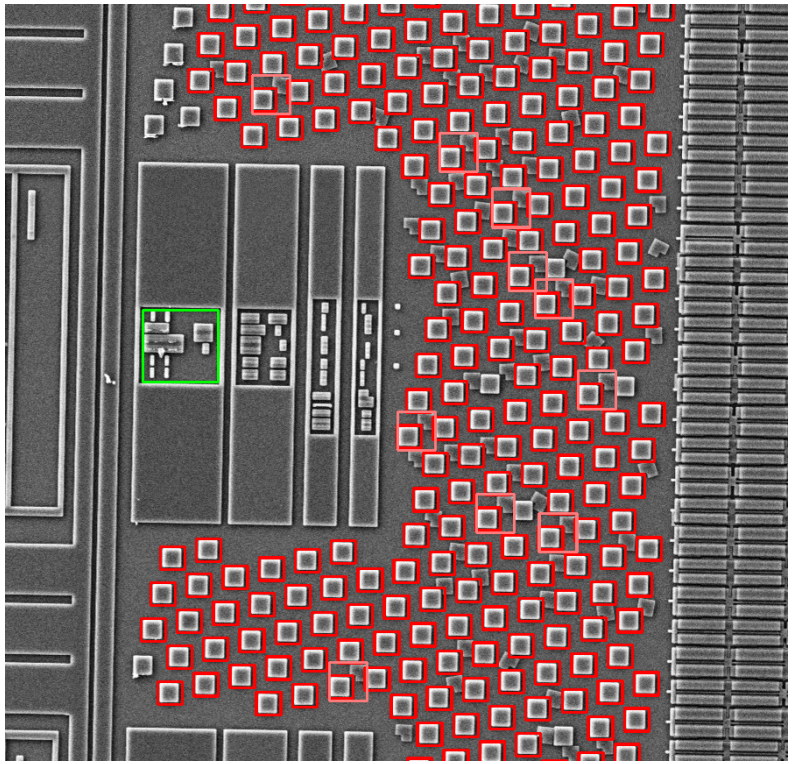
- Choose a scale invariant function
- Apply it to the same areas of the two images
- Locate the peaks in the function

Template matching

Objectives

Find areas that look like the template image ■





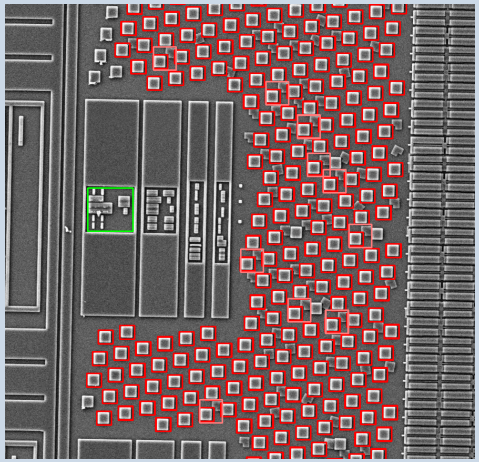
Template matching

Solution

- Use a correlation map.
- Take the maxima
- Definition:

$$(f \star g)(\tau) = \int_{-\infty}^{\infty} f^*(t)\, g(t+\tau)\, dt$$

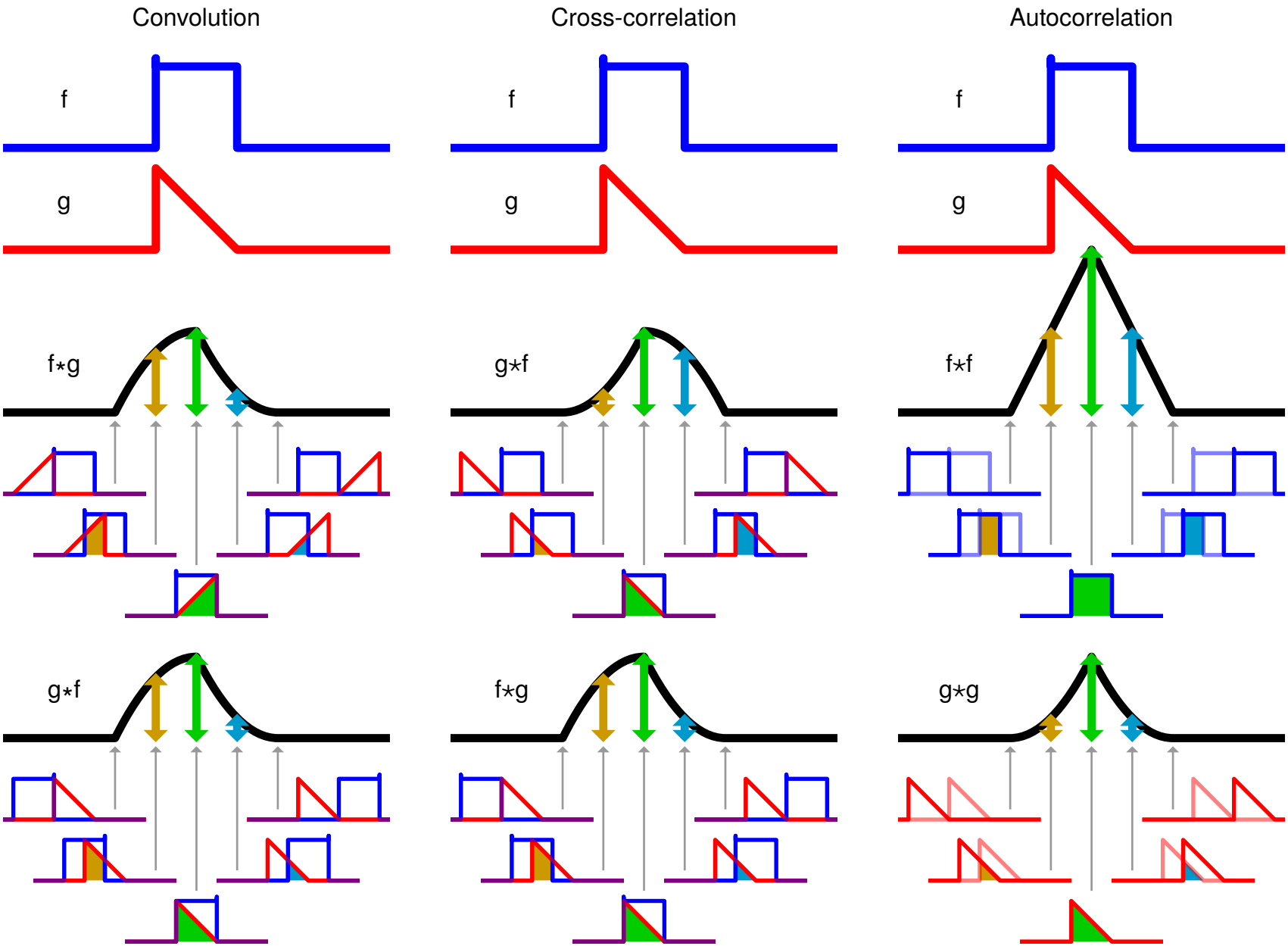
f^* is the complex conjugate.



Limitations

Intolerant to small variations:

- illumination,
- scale changes,
- rotation,
- more generally: shape variations.


Convolution
Cross-correlation
Autocorrelation
f
g
f
g
f
g
f⋆g
g⋆f
f⋆f
g⋆f
f⋆g
g⋆g

Algorithms

- Like convolution.
- Use FFT for speed

Algorithms

- Like convolution.
- Use FFT for speed

Other measures

- SAD: Sum of Absolute Differences
- SSD: Sum of Squared Differences

Algorithms

- Like convolution.
- Use FFT for speed

Other measures

- SAD: Sum of Absolute Differences
- SSD: Sum of Squared Differences

Efficiency

- Might be slow
- Value is sensitive to contrast and illumination changes
- Use ZNCC or NCC (Zero-Normalized Cross Correlation)
  - uses integral image
  - parallel version
  - FPGA friendly

- ZSAD
- ZSSD

To go further:
Census Transform

Section 2

Feature detectors

Point features: Corner vs Blob detectors

Corners:

- Harris,
- Shi-Tomasi,
- SUSAN
- FAST

Blob:

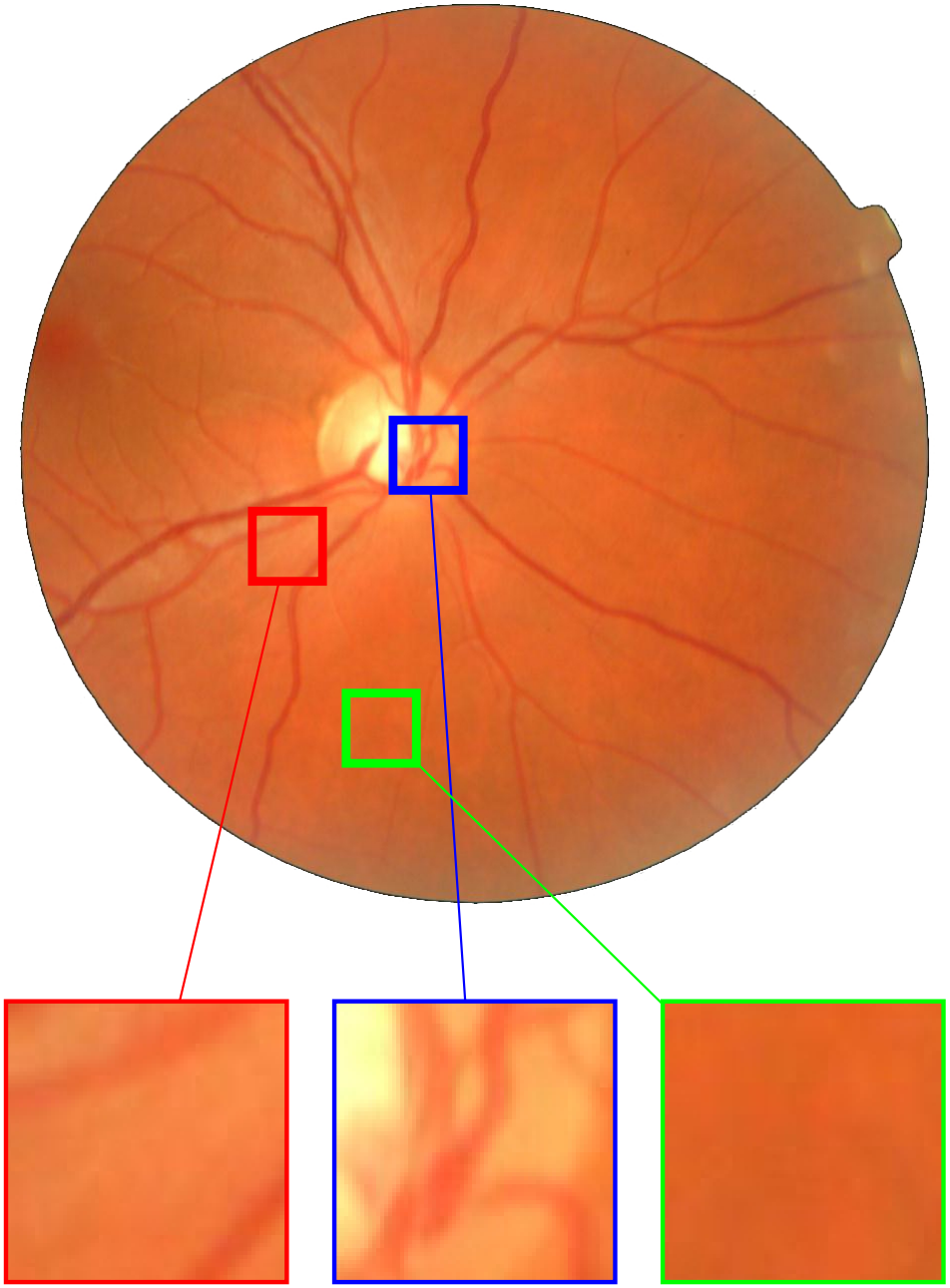
- MSER
- DoG (SIFT), LoG
- SURF
- CenSurE

Vocabulary

- Edges: contours
- Points
- Blobs: regions
- Ridges: medial axis

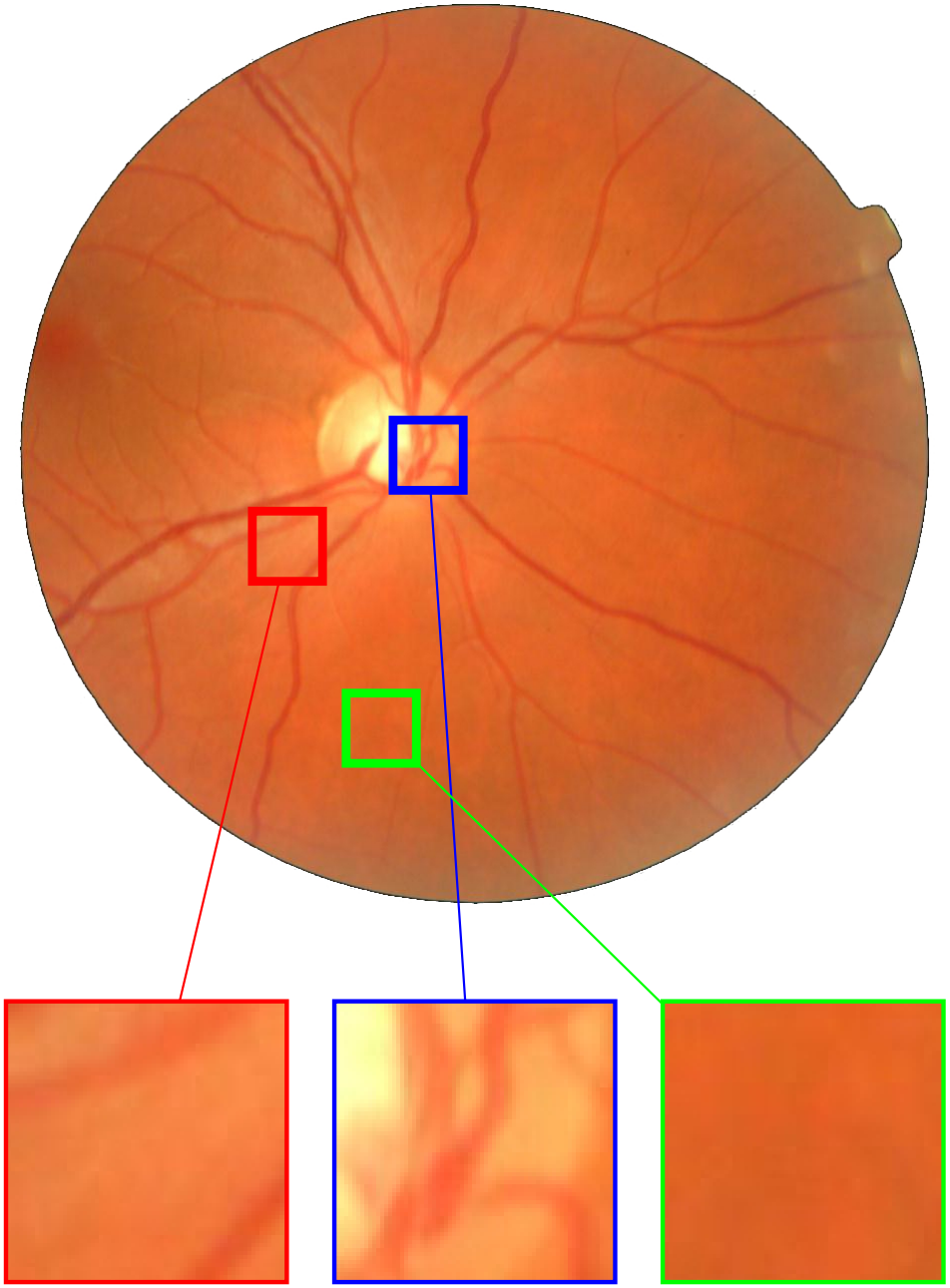
Properties

- Robustness (scales, deformations...)
- Repeatability (different conditions)
- Accuracy (spatial precision)
- Generality of applications
- Efficiency (speed)
- Quantity (all features detected)



Goal

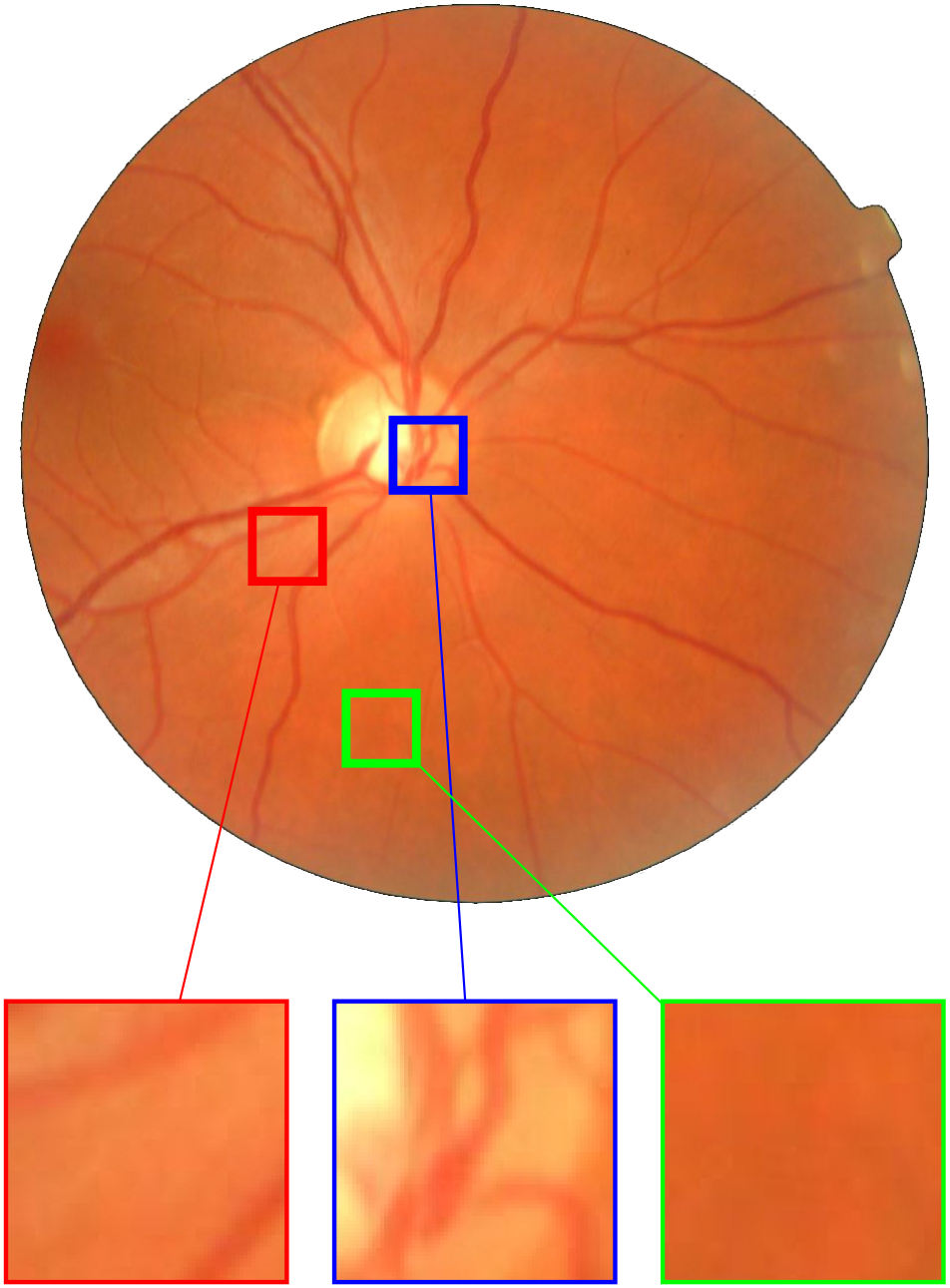
- Find and match patches
- What is a good patch?



Goal

- Find and match patches
- What is a good patch?

- can see differences
- corners ?
- non flat areas



Goal

- Find and match patches
- What is a good patch?

- can see differences
- corners ?
- non flat areas

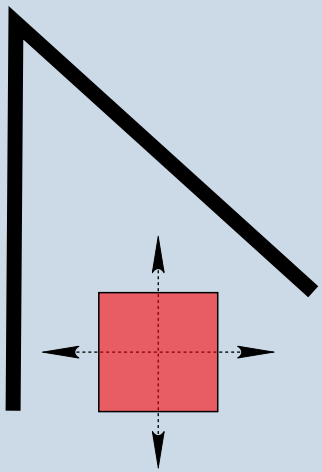
What is a corner?

- junctions of contours/edges
- intensity variations in neighborhood

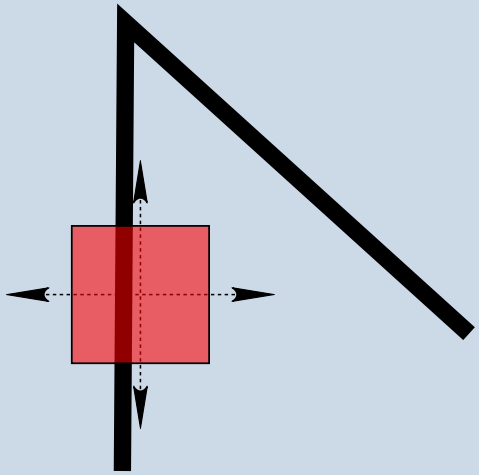
Corner detection

Basic idea

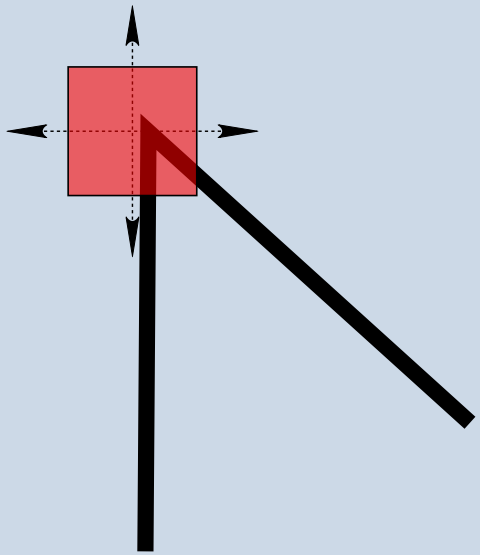
- Look at neighborhood
- Evaluate intensity change
- For all directions



No change



No change along edge

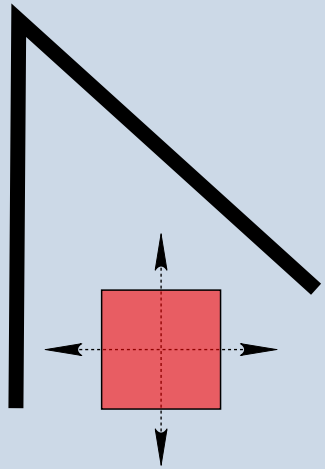


Big changes

Moravec's detector

Intensity variation evaluation

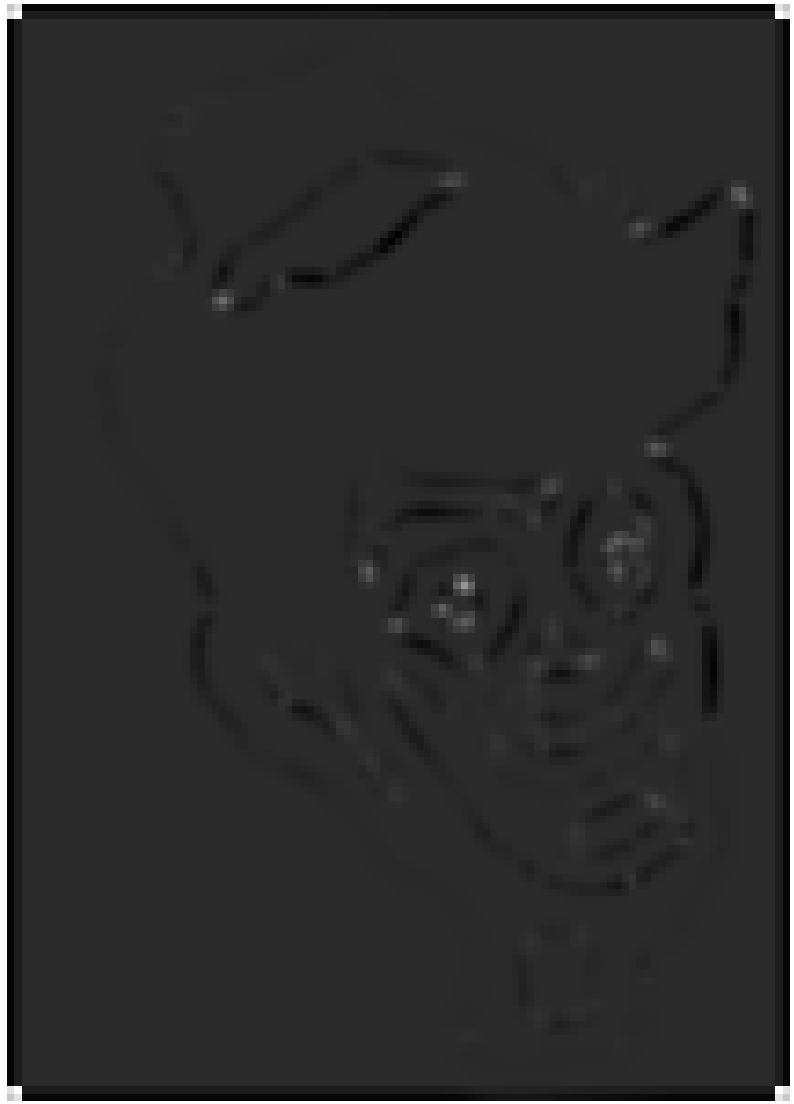
- Mathematical formulation of the "changes".
- Let I be a grayscale image
- W is a window (rectangular, gaussian...)
- $\Delta x, \Delta y$ is the shift



Intensity Variation : V

We want to maximize V

Intensity variation



Moravec's detector

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

Cornerness map

$\Delta x, \Delta y$ vary in a neighborhood.

$$C(x,y) = \min\left(V_{\Delta x,\Delta y}(x,y)\right)$$

Treshold value T

Keep local maxima that are higher than threshold value T.
Remaining points are corners.

Pros

- Detect the majority of the corners.

Cons

- Anisotropic: thus, not invariant by rotation.

Harris Corner detector

Taylor series expansion

$$I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y$$

$$I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}$$

Harris Corner detector

Taylor series expansion

$$I(x + \Delta x, y + \Delta y) \approx I(x, y) + I_x(x, y)\Delta x + I_y(x, y)\Delta y$$

$$I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}$$

Approximation of intensity Variation V

$$V_{\Delta x, \Delta y}(x, y) \approx \sum_{(u,v) \in W(x,y)} (I_x(u, v)\Delta x + I_y(u, v)\Delta y)^2$$

Harris Corner detector

Taylor series expansion

$$I(x+\Delta x, y+\Delta y) \approx I(x,y) + I_x(x,y)\Delta x + I_y(x,y)\Delta y$$

$$I_x = \frac{\partial I}{\partial x}, I_y = \frac{\partial I}{\partial y}$$

Approximation of intensity Variation V

$$V_{\Delta x,\Delta y}(x,y) \approx \begin{pmatrix}\Delta x & \Delta y\end{pmatrix} M \begin{pmatrix}\Delta x \\ \Delta y\end{pmatrix}$$

$$M = \sum_{(u,v)\in W} \omega(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

ω is a weight function (Gaussian filter)

Harris corner detector

M is the structure tensor.

- second-moment matrix (symmetric)
- predominant directions of the gradient
- depend on neighborhood and point
- 2×2 matrix

I_x and I_y are the gradients in x and y directions for the considered point.

$$M = \sum_{(x,y) \in W} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \sum_{(x,y) \in W} I_x^2 & \sum_{(x,y) \in W} I_x I_y \\ \sum_{(x,y) \in W} I_x I_y & \sum_{(x,y) \in W} I_y^2 \end{bmatrix}$$

Harris corner detector

$$M = \begin{bmatrix} \sum_{(u,v)\in W} \omega(u,v) I_x(u,v)^2 & \sum_{(u,v)\in W} \omega(u,v) I_x(u,v) I_y(u,v) \\ \sum_{(u,v)\in W} \omega(u,v) I_x(u,v) I_y(u,v) & \sum_{(u,v)\in W} \omega(u,v) I_y(u,v)^2 \end{bmatrix}$$

Only pointwise operations are used!

- ω introduces isotropy
- use uniform filter or Gaussian filter

Harris corner detector

Eigenvalues of M

- M is symmetric, it can be diagonalized, with eigenvalues λ_1, λ_2
- Interpretation
  - Distribution of gradient vectors
  - an ellipse with axis lengths λ_1 and λ_2

Shi and Tomasi detector

$$C(x, y) = \min(\lambda_1, \lambda_2)$$

Harris corner detector

Eigenvalues of M

- M is symmetric, it can be diagonalized, with eigenvalues λ_1, λ_2
- Interpretation
  - Distribution of gradient vectors
  - an ellipse with axis lengths λ_1 and λ_2

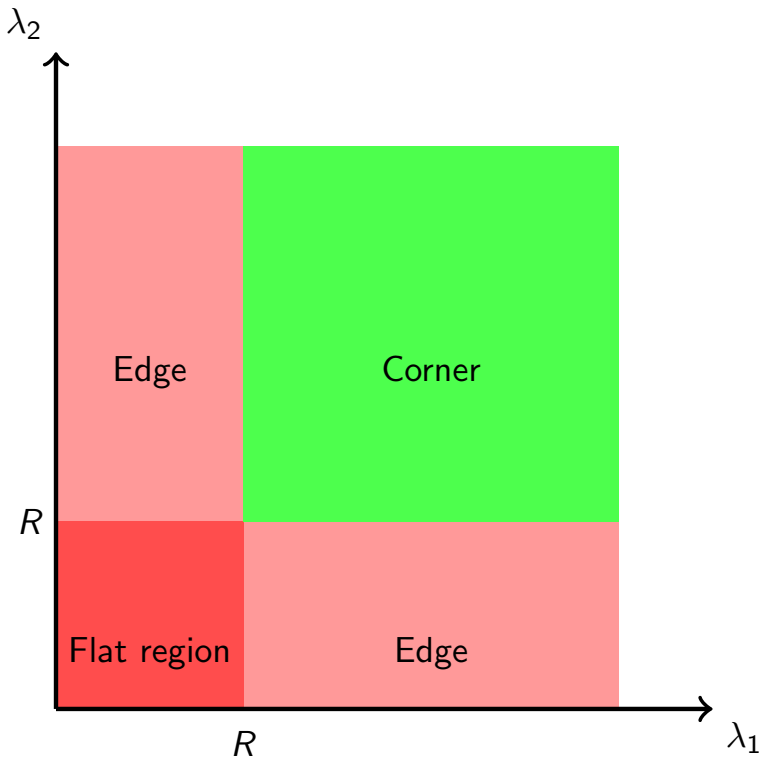
Cornerness measure (Harris and Stephens)

- $\det(M) = \lambda_1 \cdot \lambda_2$
- $\mathrm{trace}(M) = \lambda_1 + \lambda_2$
- λ_1, λ_2 not computed in practice, for efficiency reasons
- K is the *magic number*, between 0.04 and 0.15.

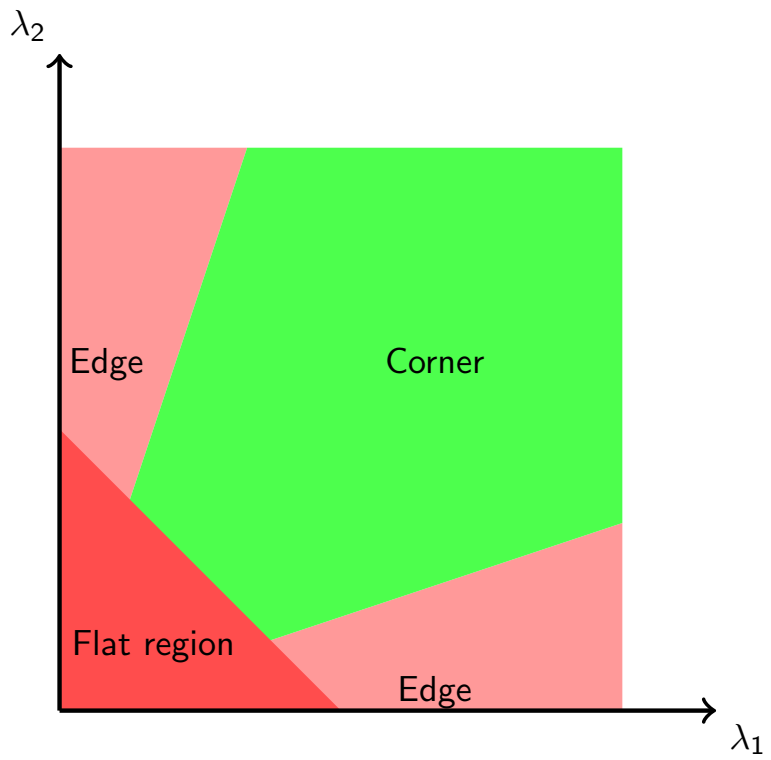
$$C(x, y) = \det(M) - K\mathrm{trace}(M)^2$$

Shi/Tomasi vs Harris corner detector

- $C(x, y) = \min(\lambda_1, \lambda_2)$
- R: threshold value



- $C(x, y) = \det(M) - K\mathrm{trace}(M)^2$
- $\lambda_1 \cdot \lambda_2 - K \cdot (\lambda_1 + \lambda_2)^2$



Harris corner detector

Algorithm

- Compute gradients in x and y directions (Sobel, prewitt...),
- Compute I_x^2, $I_x \cdot I_y$, I_y^2 (pointwise operations),
- Convolve them with ω (uniform/Gaussian filter),
- Compute the cornerness measure C,
- Keep local maxima that are above a threshold value.

Harris corner detector

Algorithm

- Compute gradients in x and y directions (Sobel, prewitt...),
- Compute I_x^2, $I_x \cdot I_y$, I_y^2 (pointwise operations),
- Convolve them with ω (uniform/Gaussian filter),
- Compute the cornerness measure C,
- Keep local maxima that are above a threshold value.

Practical problems

- Sub-pixels computation (discretization)
- Border effects
- Treshold value: keep strongest values?

Harris corner detector

Algorithm

- Compute gradients in x and y directions (Sobel, prewitt...),
- Compute I_x^2, $I_x \cdot I_y$, I_y^2 (pointwise operations),
- Convolve them with ω (uniform/Gaussian filter),
- Compute the cornerness measure C,
- Keep local maxima that are above a threshold value.

Practical problems

- Sub-pixels computation (discretization)
- Border effects
- Treshold value: keep strongest values?

Extension

- Multiscale (Gaussian pyramid...)

SUSAN detector

- Consider a mask M (here, a disk)
- Its center: nucleus p_0
- Look at pixels $p \in M$
- Treshold value t

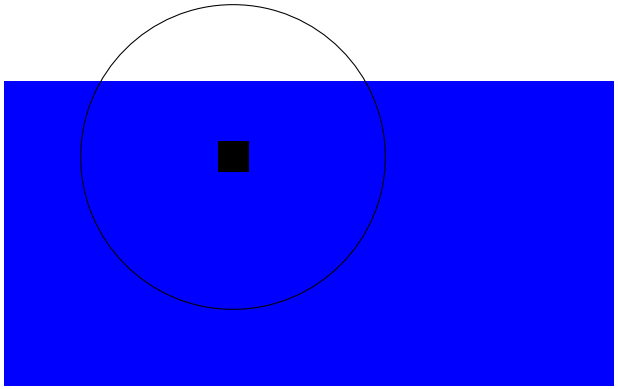
Principle

- Evaluates part of the mask that recovers the object

$$C(p, p_0) = \begin{cases} 1 \text{ if } |I(p) - I(p_0)| \leq t \\ 0 \text{ otherwise} \end{cases}$$

$$n(M) = \sum_{p \in M} C(p, p_0)$$

SUSAN detector



- Consider a mask M (here, a disk)
- Its center: nucleus p_0
- Look at pixels $p \in M$
- Treshold value t

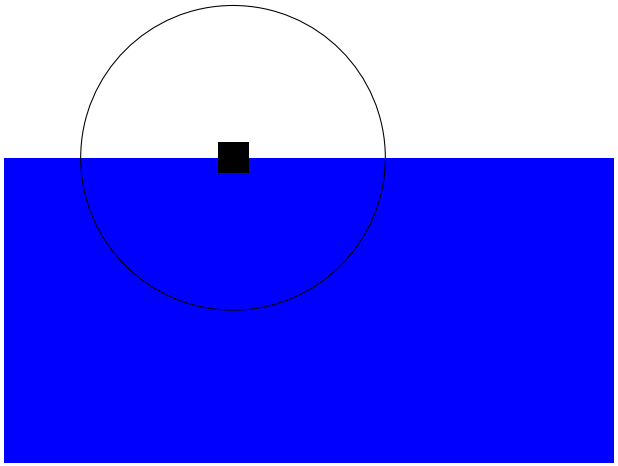
Principle

- Evaluates part of the mask that recovers the object

$$C(p, p_0) = \begin{cases} 1 \text{ if } |I(p) - I(p_0)| \leq t \\ 0 \text{ otherwise} \end{cases}$$

$$n(M) = \sum_{p \in M} C(p, p_0)$$

SUSAN detector



- Consider a mask M (here, a disk)
- Its center: nucleus p_0
- Look at pixels $p \in M$
- Treshold value t

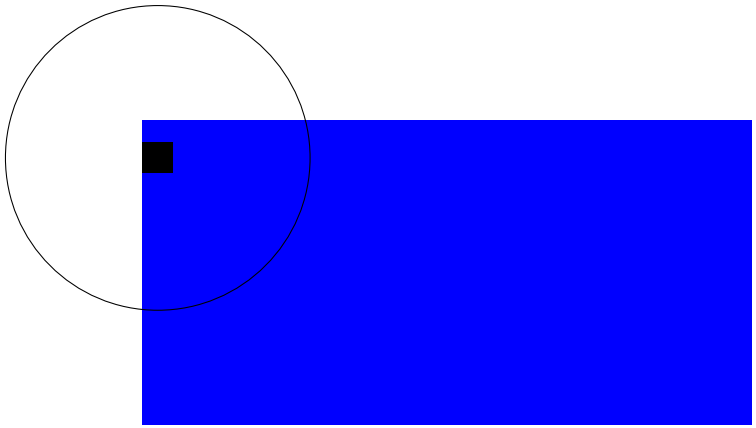
Principle

- Evaluates part of the mask that recovers the object

$$C(p, p_0) = \begin{cases} 1 \text{ if } |I(p) - I(p_0)| \leq t \\ 0 \text{ otherwise} \end{cases}$$

$$n(M) = \sum_{p \in M} C(p, p_0)$$

SUSAN detector



- Consider a mask M (here, a disk)
- Its center: nucleus p_0
- Look at pixels $p \in M$
- Treshold value t

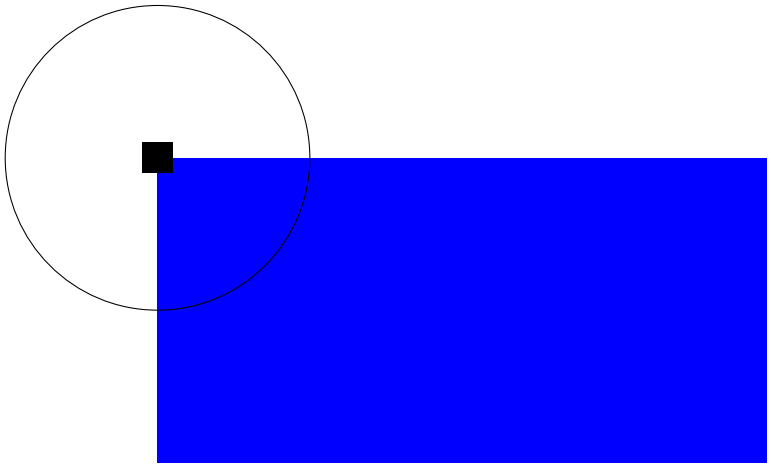
Principle

- Evaluates part of the mask that recovers the object

$$C(p, p_0) = \begin{cases} 1 \text{ if } |I(p) - I(p_0)| \leq t \\ 0 \text{ otherwise} \end{cases}$$

$$n(M) = \sum_{p \in M} C(p, p_0)$$

SUSAN detector



- Consider a mask M (here, a disk)
- Its center: nucleus p_0
- Look at pixels $p \in M$
- Treshold value t

Principle

- Evaluates part of the mask that recovers the object

$$C(p, p_0) = \begin{cases} 1 \text{ if } |I(p) - I(p_0)| \leq t \\ 0 \text{ otherwise} \end{cases}$$

$$n(M) = \sum_{p \in M} C(p, p_0)$$

g: geometric threshold, $R(M)) = \begin{cases} g - n(M) \text{ if } n(M) \leq g \\ 0 \text{ otherwise} \end{cases}$

- corner: n must be less than half its maximum possible value
- $g = \frac{1}{2} n_{\max}$
- threshold R, detect local maxima
- can be used as edge detector
- other distances C can be used

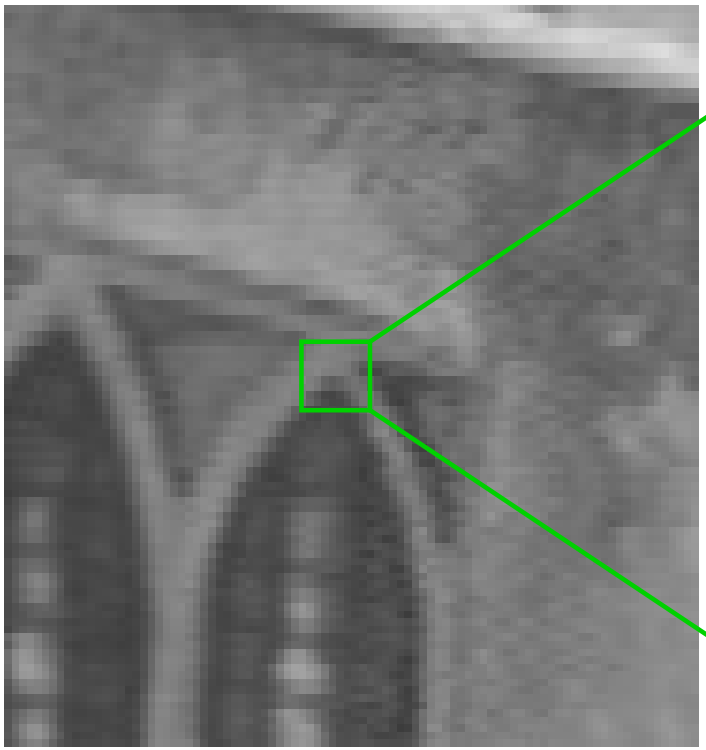
Pros and cons

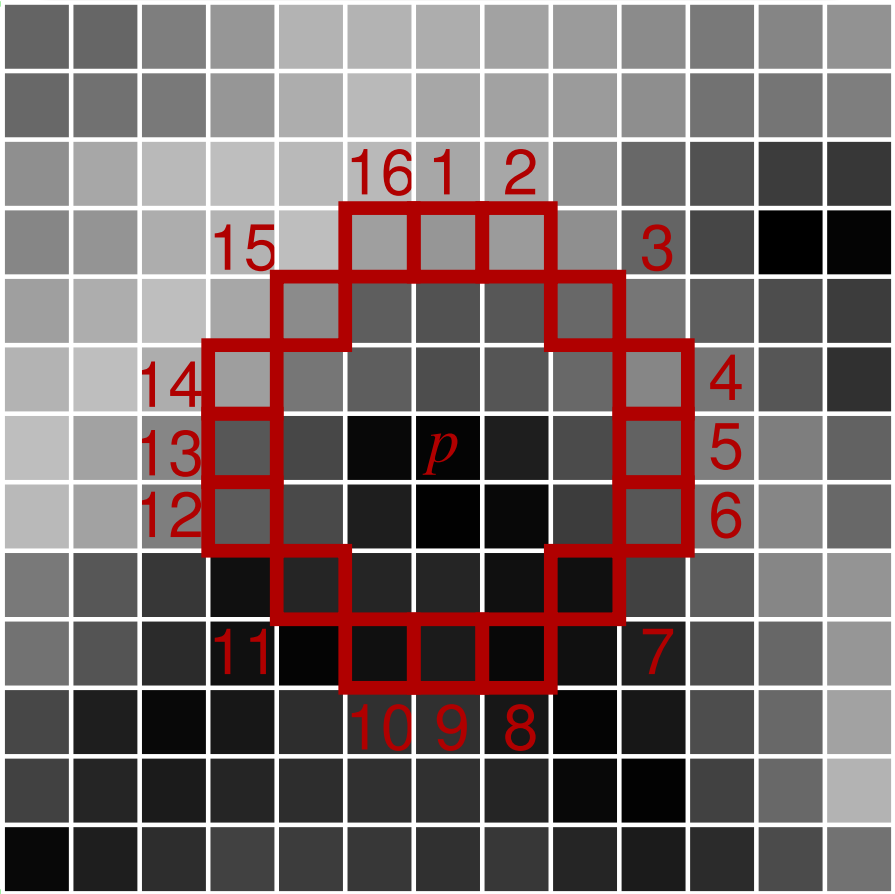
- Invariant by translation and rotation (depends on M)
- Sensitive to scaling
- Fixed global threshold g not always adapted

FAST: Feature from Accelerated Segment Test [2]

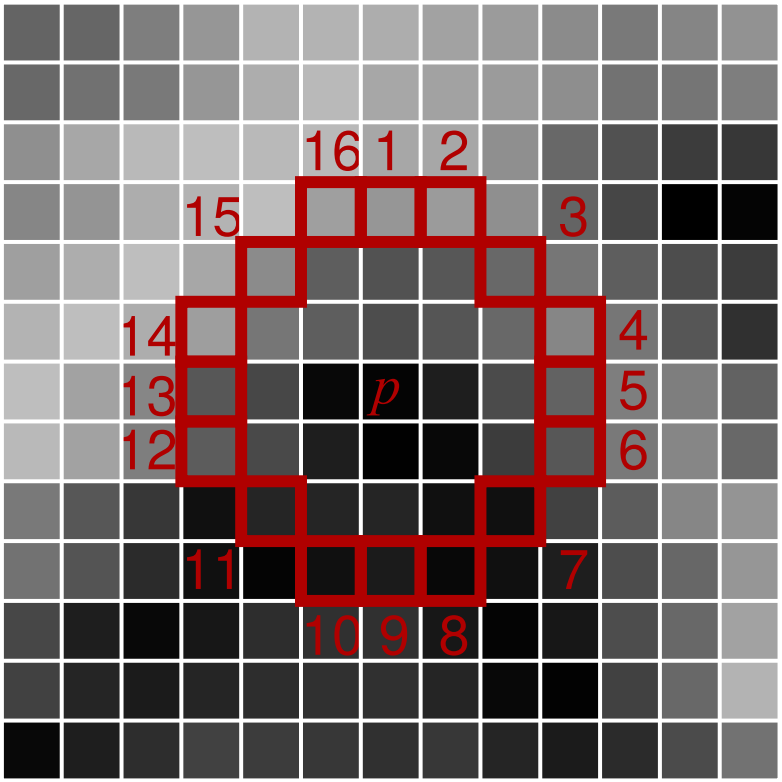
- evolution from SUSAN
- real-time operator

- Bresenham circle instead of a mask M
- 16 pixels to test



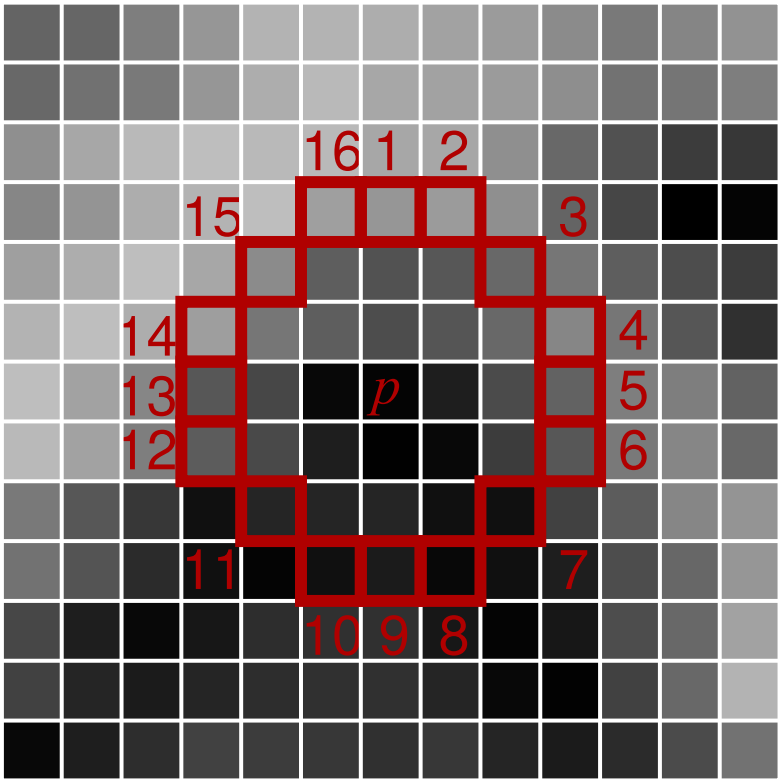


FAST



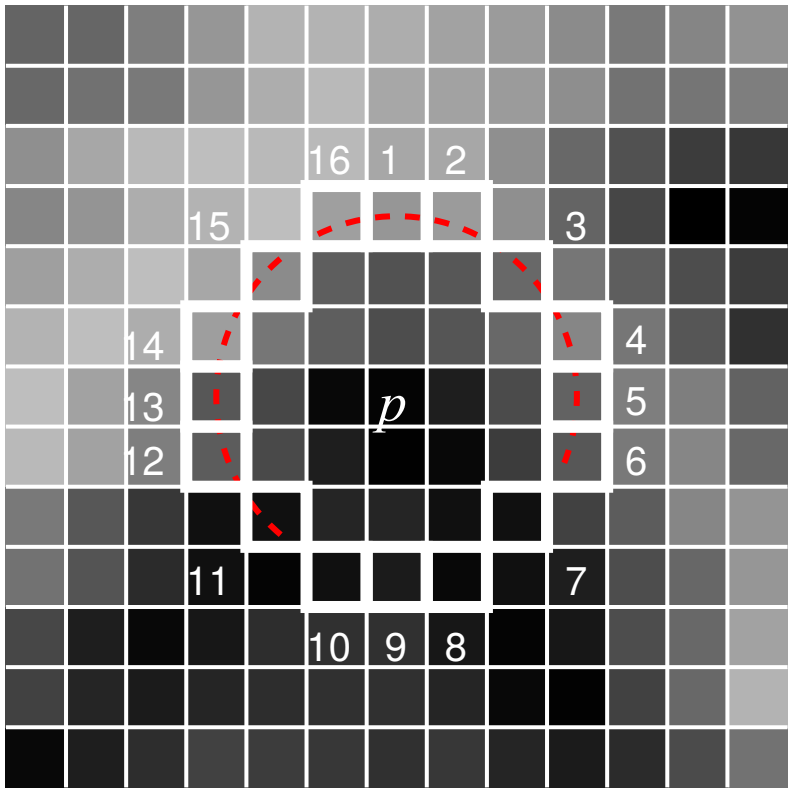
- Consider pixel p on image I

FAST



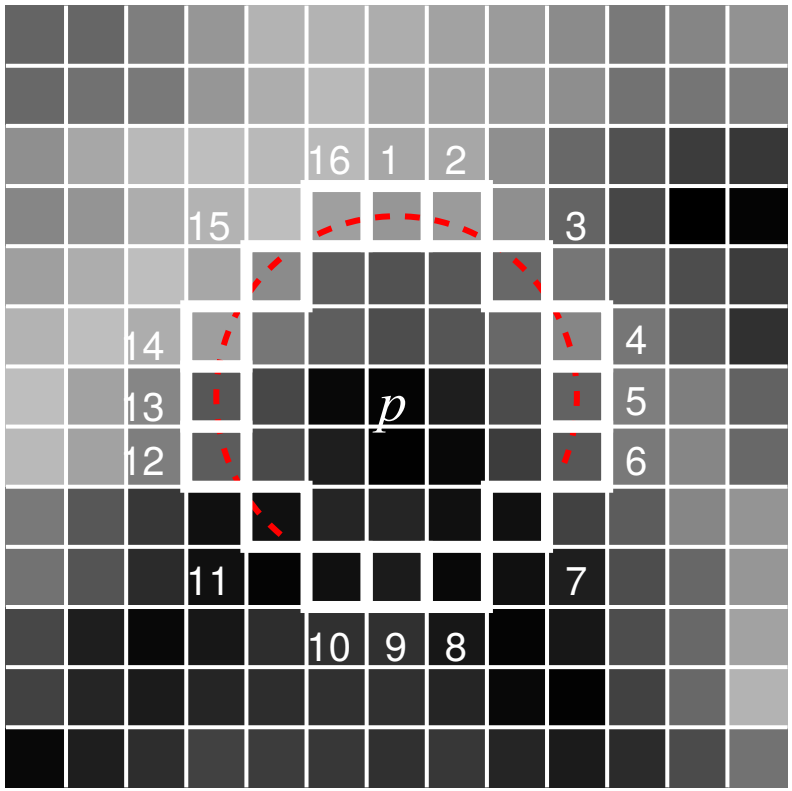
- Consider pixel p on image I
- Threshold value t

FAST



- Consider pixel p on image I
- Threshold value t
- Search for $n = 12$ contiguous pixels
  - brighter that $I_p - t$
  - or darker than $I_p + t$

FAST

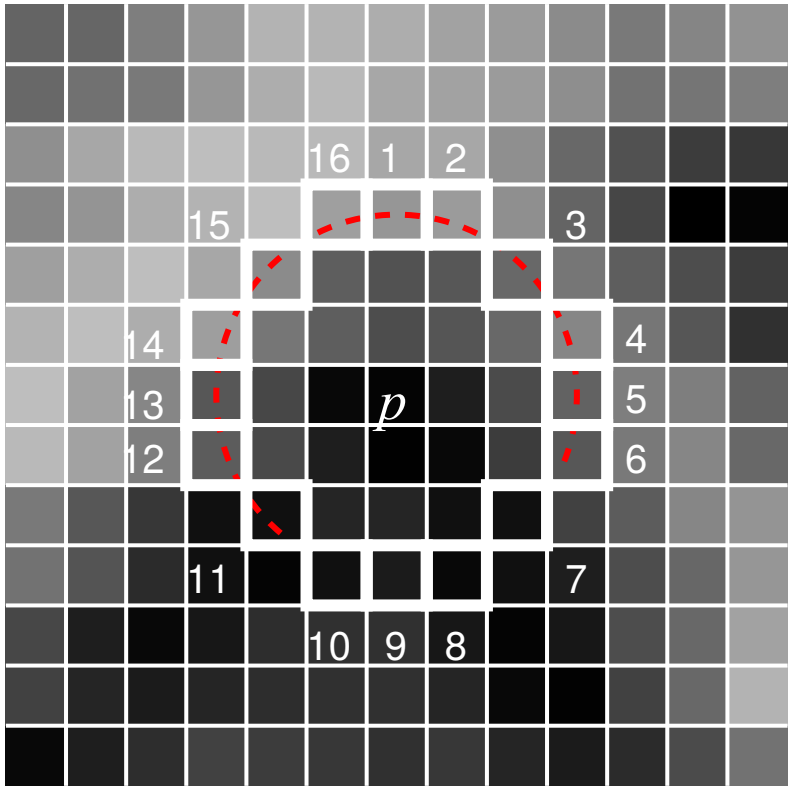


- Consider pixel p on image I
- Threshold value t
- Search for $n = 12$ contiguous pixels
  - brighter that $I_p - t$
  - or darker than $I_p + t$

High-speed test

- Test 1 and 9
- Test 5 and 13
- Test full segment

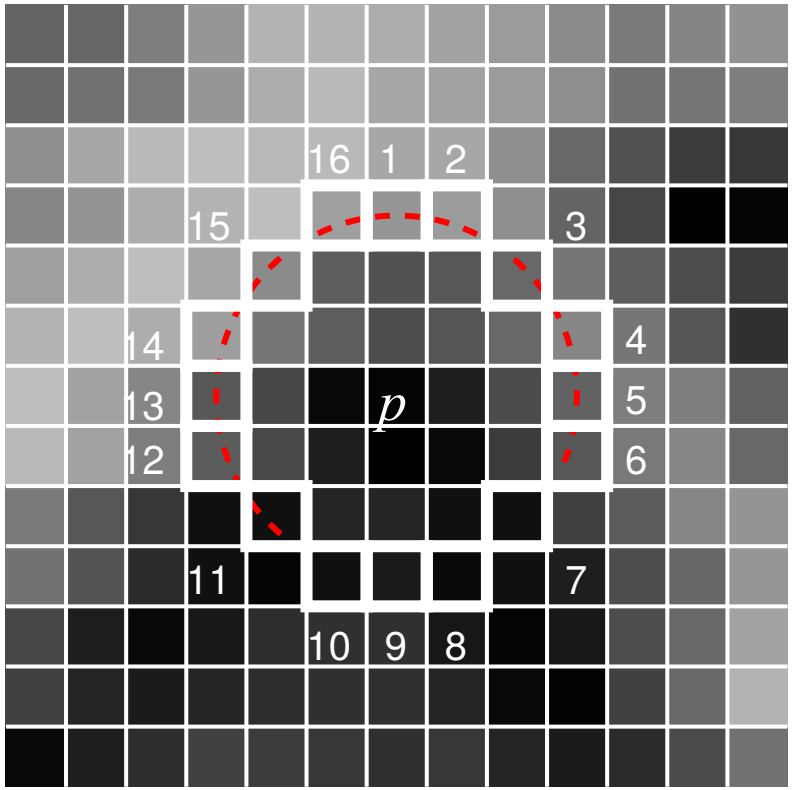
FAST-ER: machine learning enhancement [3]



Drawbacks

- It does not reject as many candidates for $n < 12$.
- choice of pixels not optimal
  - its efficiency depends on ordering of the tests
  - and distribution of corner appearances.
- No memory in the tests.
- Multiple features are detected adjacent to one another.

FAST-ER: machine learning enhancement [3]



Summary

- Suitable for real-time video processing
- NOT invariant to scales changes
- sensitive to noise
- selecting threshold is not trivial

Hessian blob detector

Hessian Matrix

$$H(p,\sigma) = \begin{pmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{yx}(p,\sigma) & L_{yy}(p,\sigma) \end{pmatrix}$$

Gaussian function

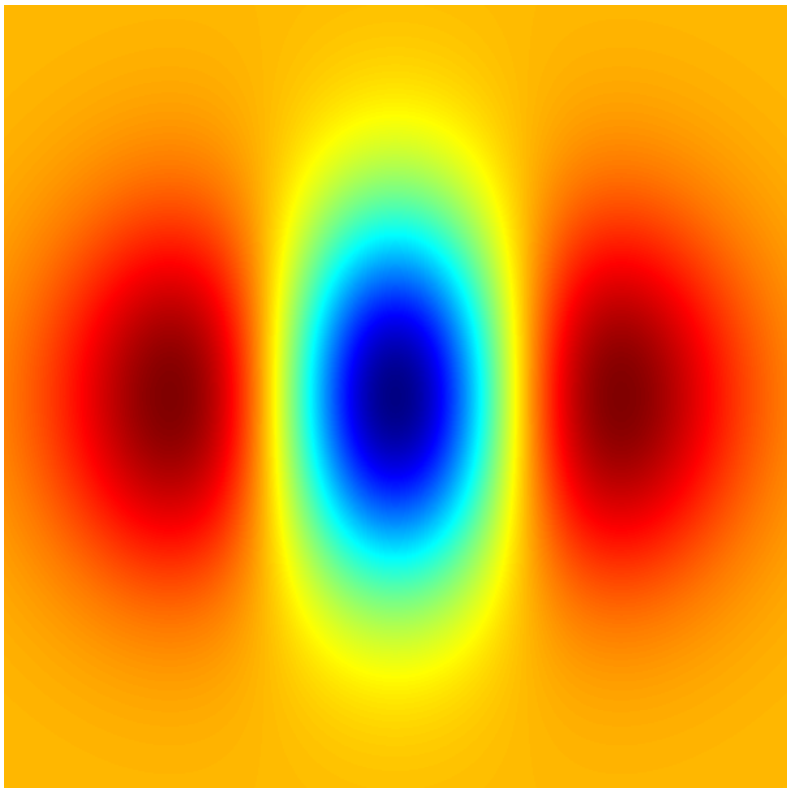
$$g(p,\sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{\|p\|^2}{2\sigma^2}}$$

$$L_{xx} = \frac{\partial^2 g}{\partial x^2} * I \qquad L_{xy} = L_{yx} = \frac{\partial^2 g}{\partial x \partial y} * I$$

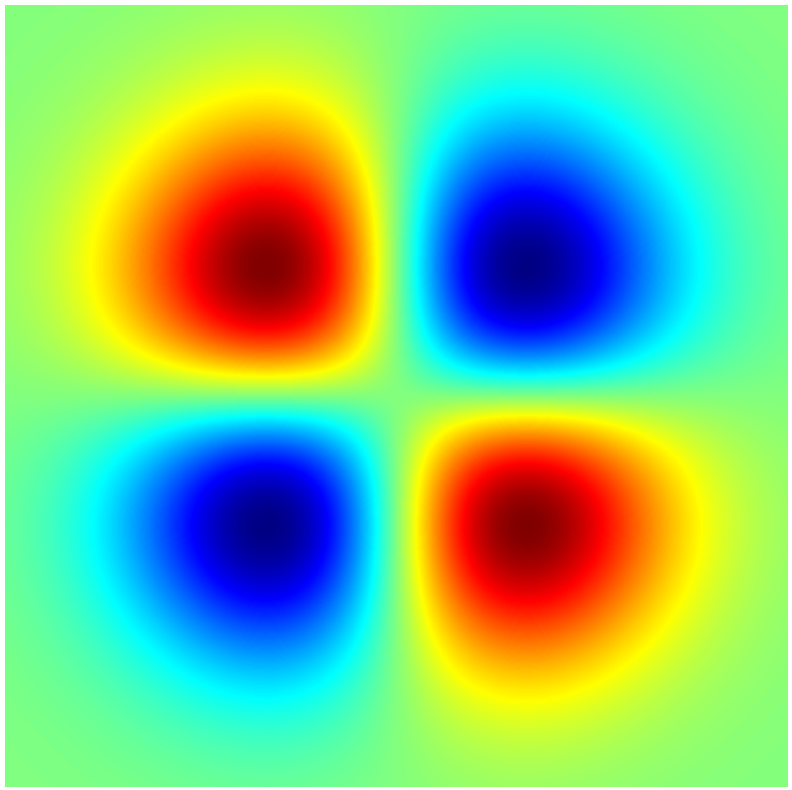
Convolution of second derivative of Gaussian function with the image.

Hessian blob detector: Gaussian derivatives

$$\frac{\partial^2 g}{\partial x^2}$$



$$\frac{\partial^2 g}{\partial x \partial y}$$



Hessian blob detector

Detector: determinant of Hessian matrix

$$\det(H) = I_{xx} I_{yy} - I_{xy}^2$$

Determine local maxima:

- global threshold
- regional maximum

- Sensitive to noise
- Scale-space approach

Laplacian of Gaussian LoG

Gaussian function

$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Scale-space representation

$$L(x, y; \sigma) = g(x, y, \sigma) * f(x, y)$$

Laplacian operator

Second derivatives L_{xx}, L_{yy}

$$\nabla^2 L = L_{xx} + L_{yy}$$

Feature detector

Scale-normalized Laplacian operator:

$$\nabla^2_{\text{norm}} L = \sigma^2 (L_{xx} + L_{yy})$$

Detect min and max according to (x, y, σ)

- Scale invariance
- Rotation invariance
- less noise-sensitve than Hessian detector

LoG: automatic scale detection [1]

Property of LoG

- (x_0, y_0, σ_0) is a max/min
- s scale parameter
- $(s \cdot x_0, s \cdot y_0, s \cdot \sigma_0)$ is also a max/min

Evolution

- Harris-Laplace
- Harris: rotation/illumination invariant
- adds scale

DoG: Difference of Gaussians

Definition

$$L(x, y; \sigma) = g(x, y, \sigma) * f(x, y)$$

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$$

- scale-space decomposition
- minima/maxima extraction

- High computation time for all scales
- also detects edges (not only corners)

Useful concepts

- octave: doubling the value of σ
- k: limits the number of scales per octave

Section 3

Feature descriptors

Feature descriptors: introduction

- Detection of points of interest
- objective: matching, pairing
- NEED for a comparison

Key question

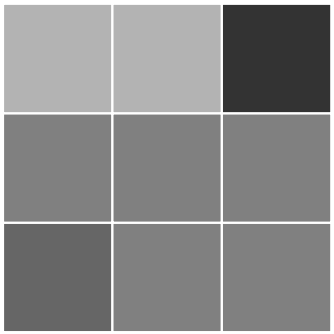
How to compare neighborhoods of keypoints?

Descriptor

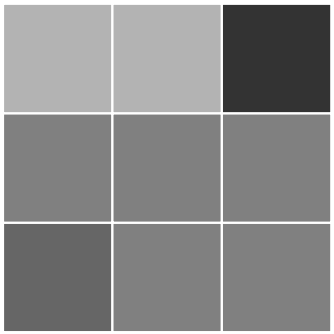
- signature
- distance between descriptors
- invariance to:
  - scale,
  - transformation,
  - noise,
  - viewing conditions...

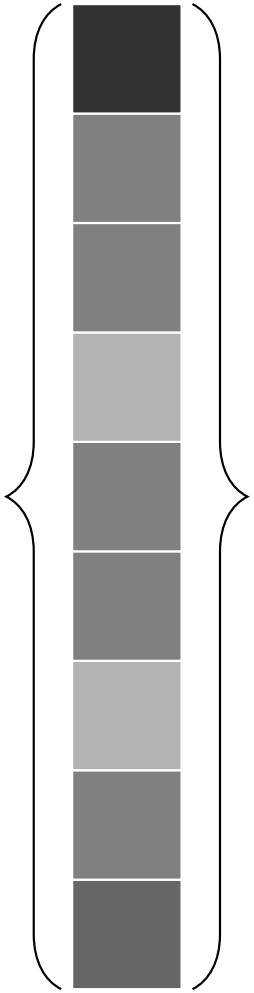
Feature descriptor: naive approach

- 3×3 neighborhood



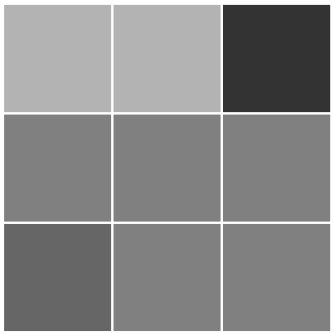
Feature descriptor: naive approach

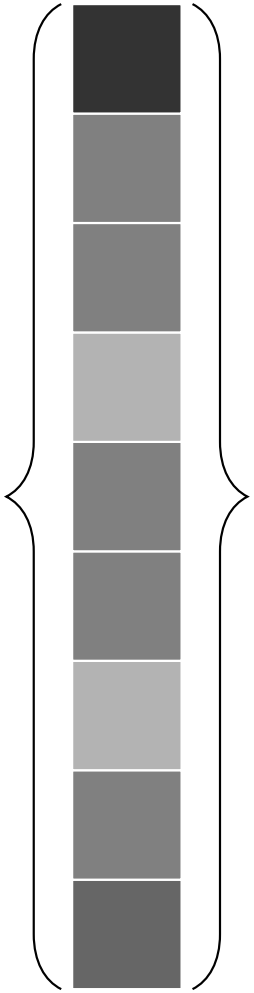




- 3×3 neighborhood
- Vector of intensities

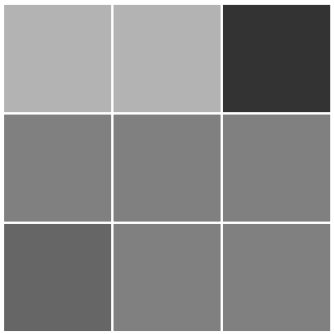
Feature descriptor: naive approach

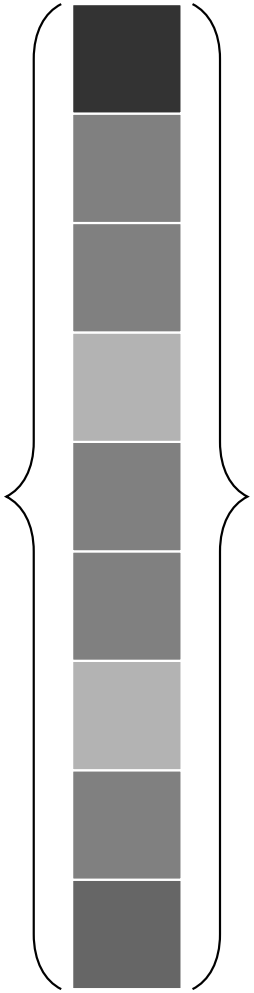




- 3×3 neighborhood
- Vector of intensities
- Euclidean distance

Feature descriptor: naive approach



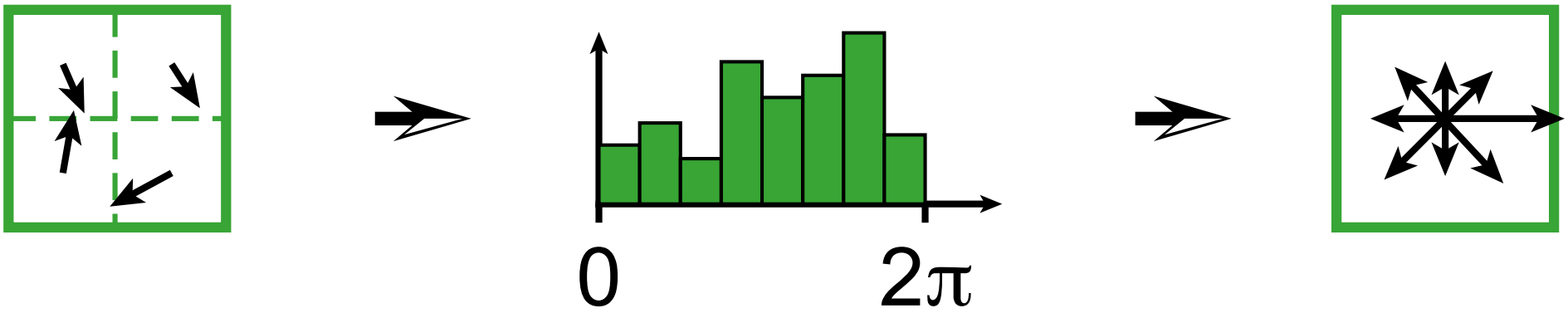


- 3×3 neighborhood
- Vector of intensities
- Euclidean distance

Problems

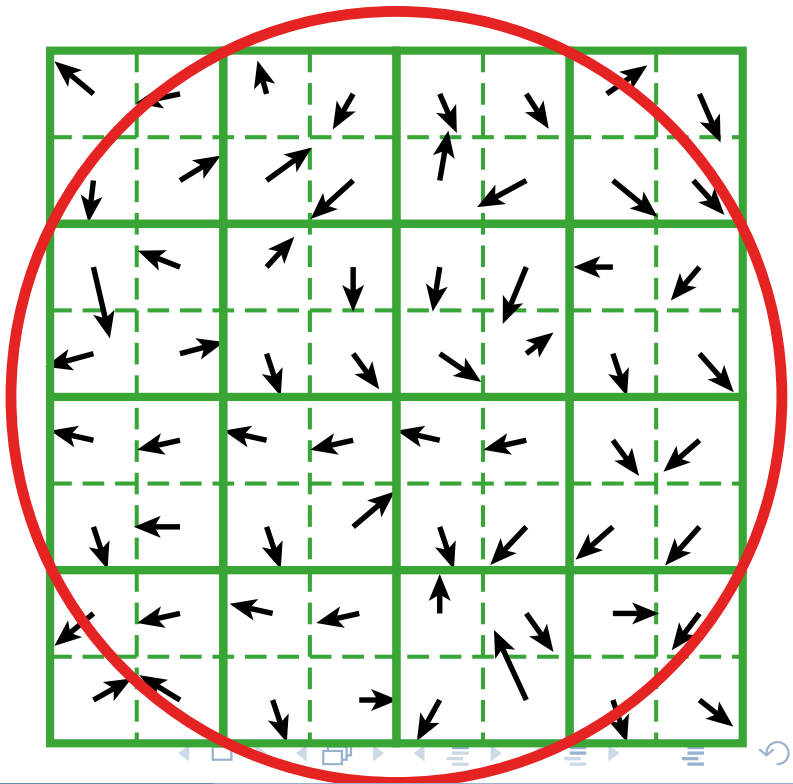
- Rotation?
- Scale?
- Noise?
- Performance?

HOG Histogram of Oriented Gradients

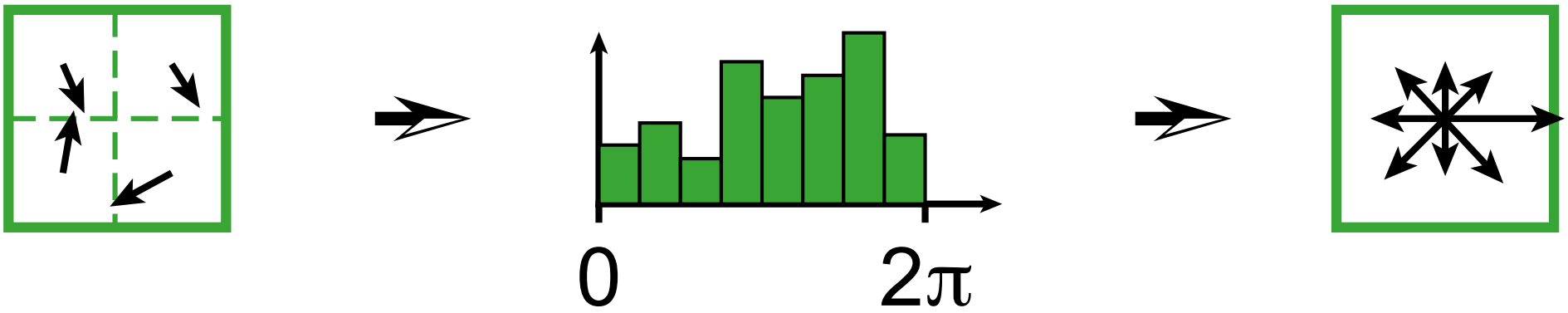


Principle

- Compute gradients (neighborhood of detected keypoints)
- Construct histogram (weighted by Gaussian function and amplitude)
- Concatenate all histograms in 1 vector

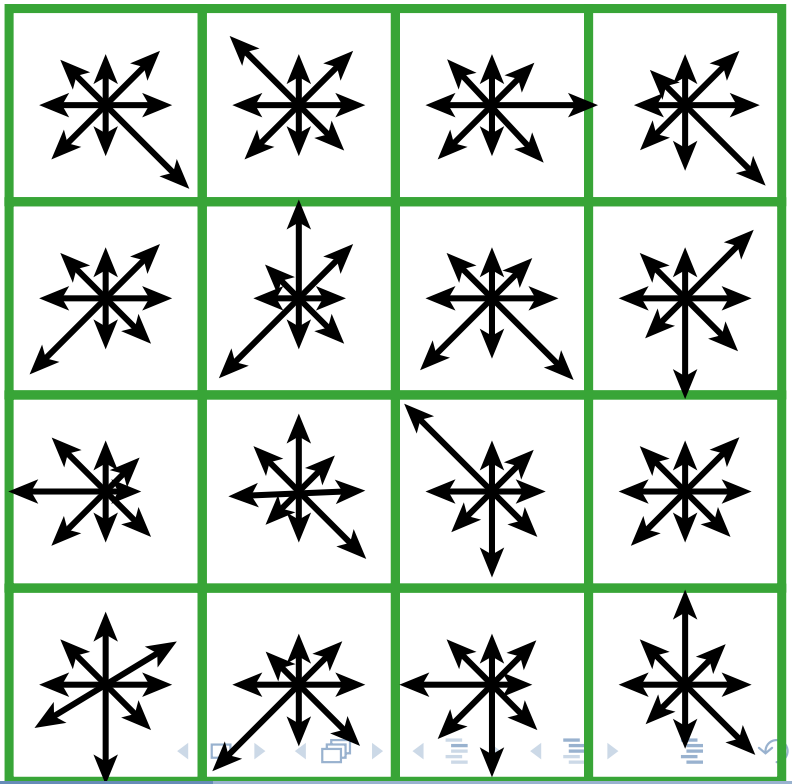


HOG Histogram of Oriented Gradients

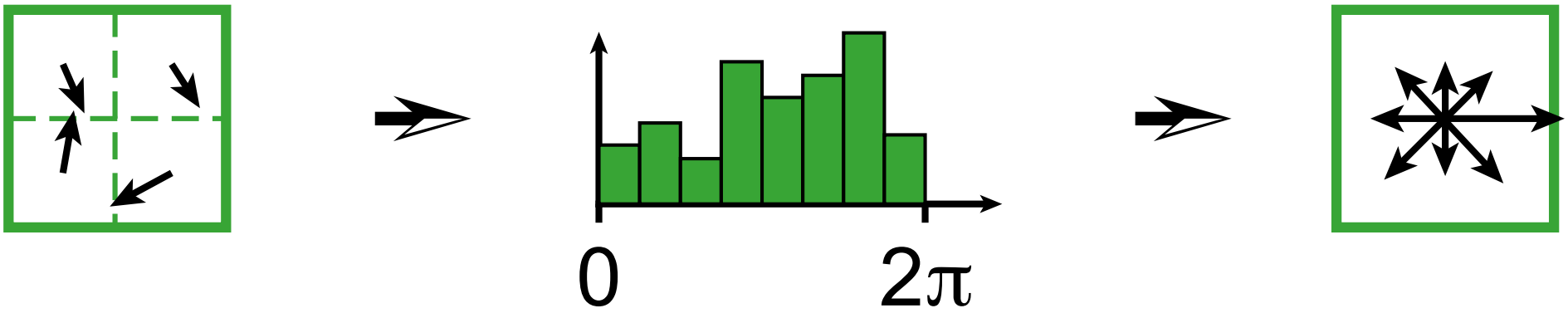


Principle

- Compute gradients (neighborhood of detected keypoints)
- Construct histogram (weighted by Gaussian function and amplitude)
- Concatenate all histograms in 1 vector

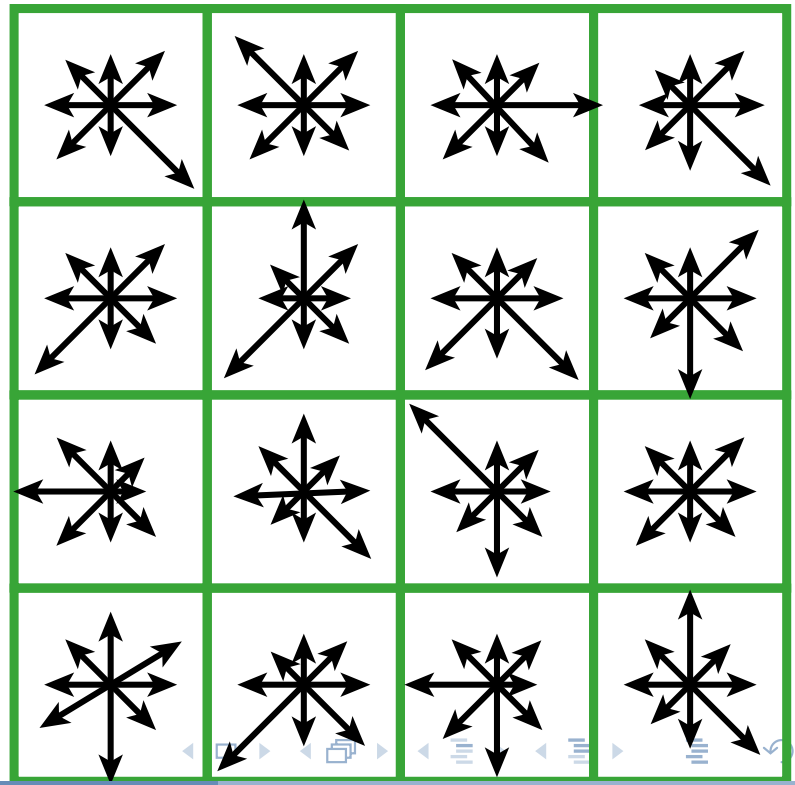


HOG Histogram of Oriented Gradients



SIFT case

- 16×16 pixels
- 4×4 descriptors of 8 bins
- Descriptor of 128 values



SIFT

Combination of

- DoG
- HOG

Additional operations

- Discard low-contrast keypoints
- Eliminate edge keypoints

Properties of invariance

- scale
- rotation (descriptor is oriented by gradient)
- translation

SIFT

Combination of

- DoG
- HOG

Additional operations

- Discard low-contrast keypoints
- Eliminate edge keypoints

Properties of invariance

- scale
- rotation (descriptor is oriented by gradient)
- translation

SIFT

Combination of

- DoG
- HOG

Additional operations

- Discard low-contrast keypoints
- Eliminate edge keypoints

Properties of invariance

- scale
- rotation (descriptor is oriented by gradient)
- translation

SURF: feature detector

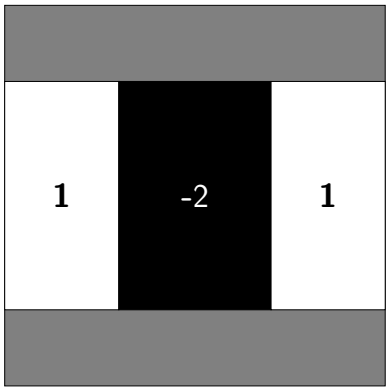
Key idea

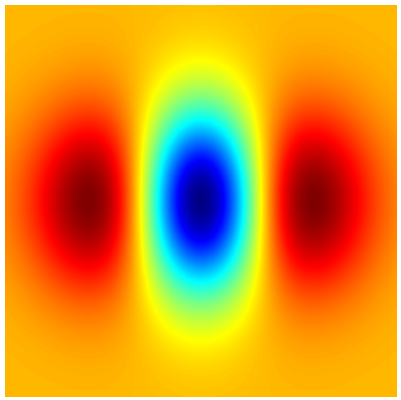
Approximation of Gaussian derivatives by using integral image, denoted by D_{xx}, D_{yy} and D_{xy}. Weight w.

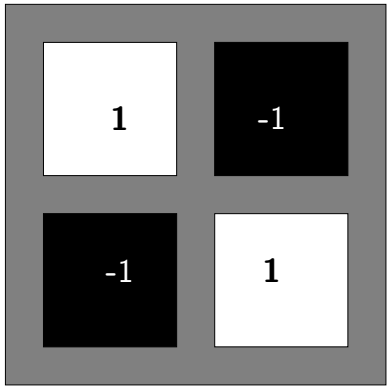
$$\det(H_{\text{approx}}) = D_{xx}D_{yy} - (wD_{xy})^2$$

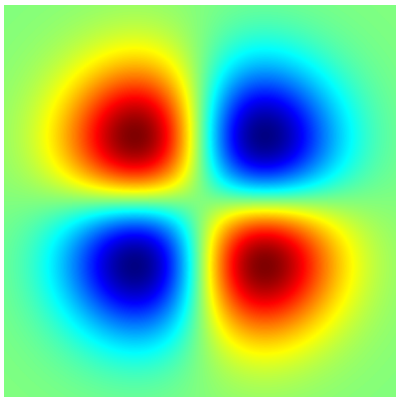
Comparison to SIFT

- much faster
- more robust
- octaves, scales per octave





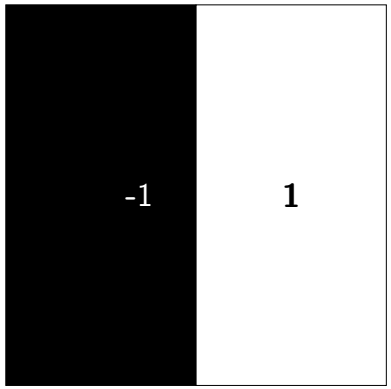


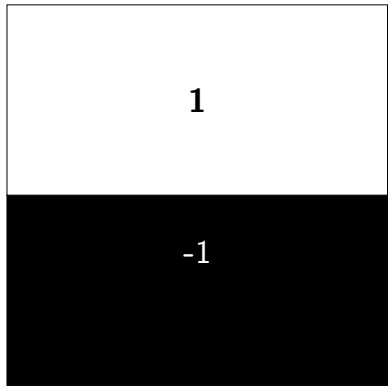


SURF: rotation invariance

Orientation estimation

- For each keypoint at scale s
- Haar response in a window of size $6s \times 6s$
- Use of integral image



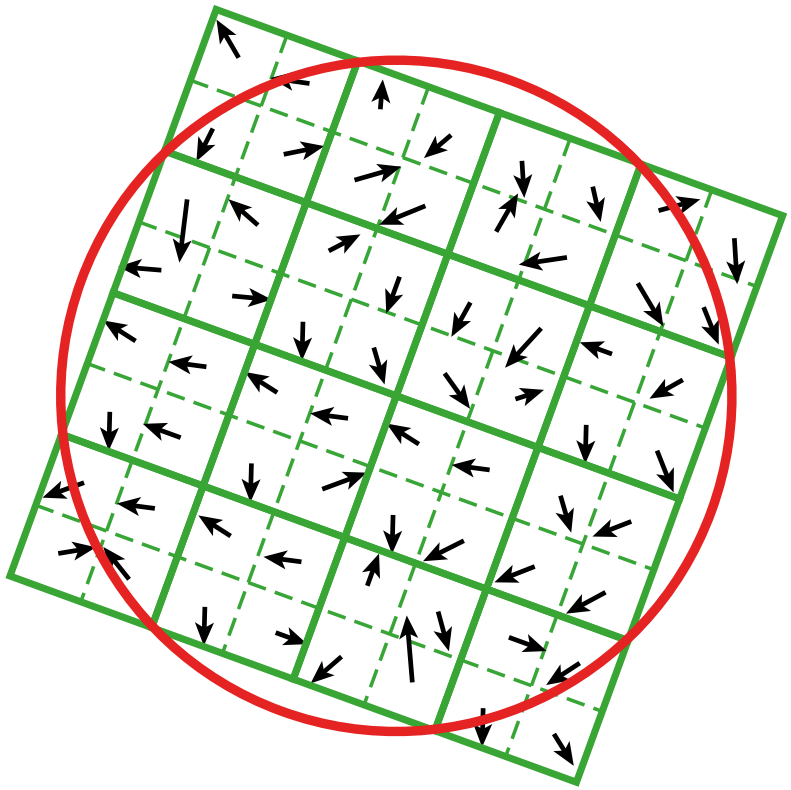


SURF: feature descriptor

Feature descriptor

- Window of size $20s \times 20s$
- Split into 4×4 areas
- Each area 5×5 squares
- Evaluate gradients in each Square
- 4 sums per area

$$\sum dx, \sum |dx|, \sum dy, \sum |dy|$$



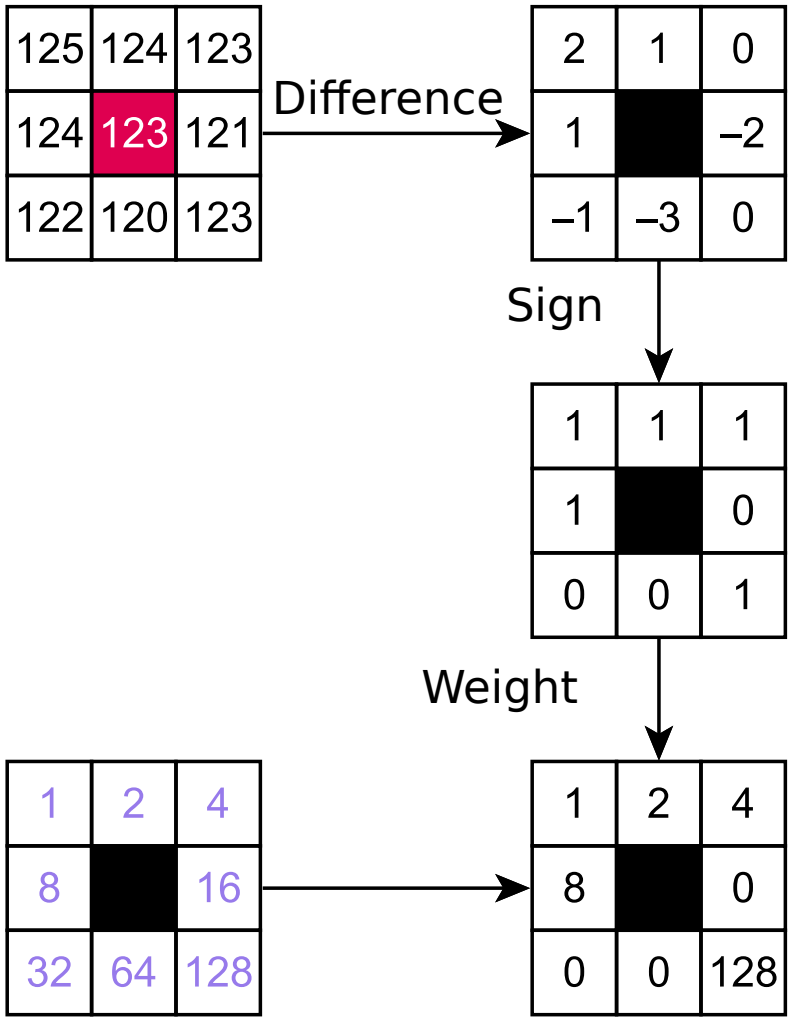
$4 \times 4 \times 4 = 64$ element descriptor

- high processing speed compared to SIFT
- Advantage to SIFT for translation, rotation, scaling, illumination changes

LBP: Local Binary Patterns

Construction

- Choose a window size

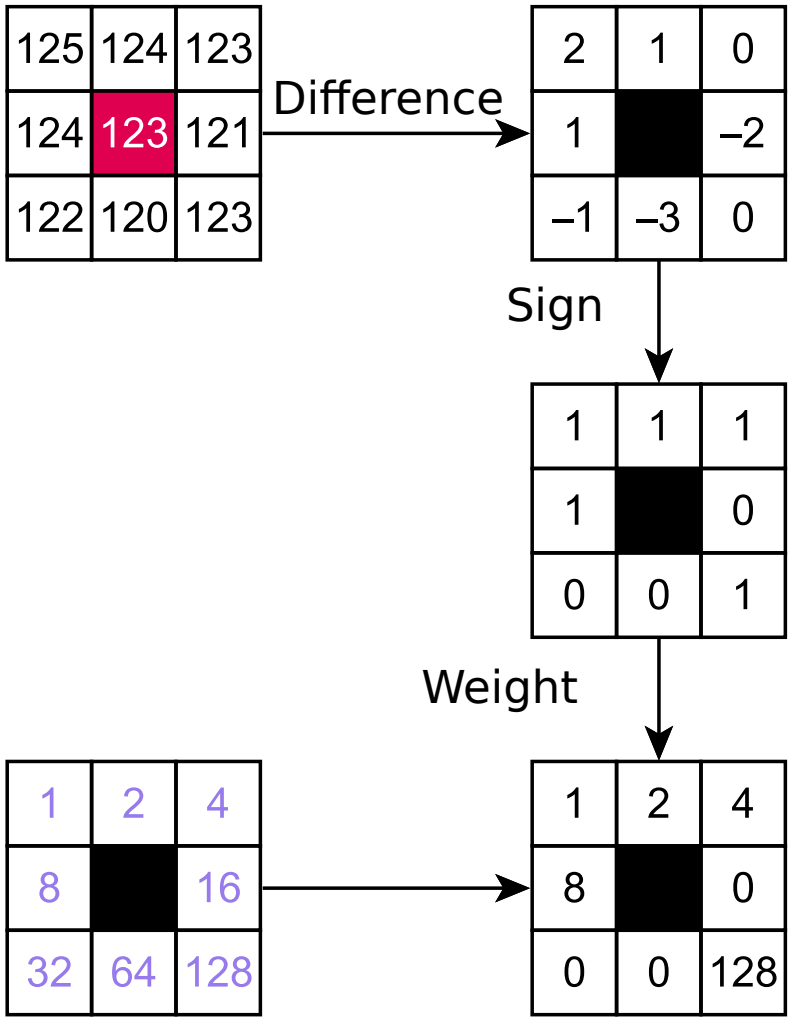


LBP = 1 + 2 + 4 + 8 + 128 = 143

LBP: Local Binary Patterns

Construction

- Choose a window size
- Each pixel in the neighborhood

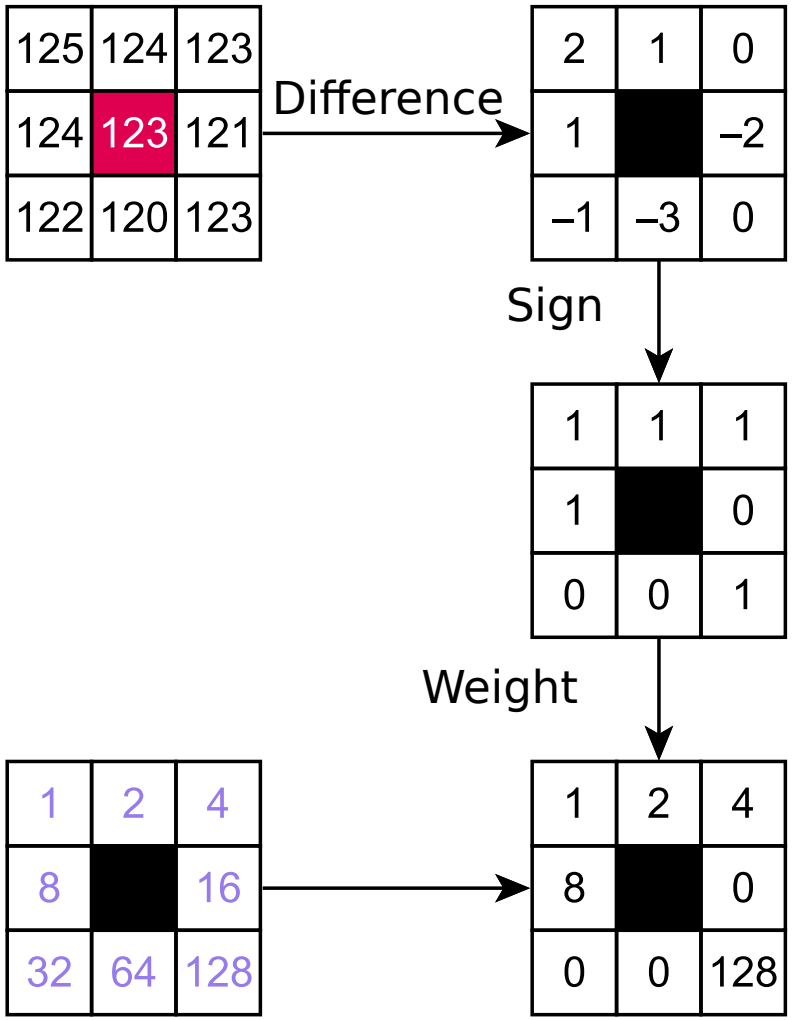


LBP = 1 + 2 + 4 + 8 + 128 = 143

LBP: Local Binary Patterns

Construction

- Choose a window size
- Each pixel in the neighborhood
- Compare to its neighbors

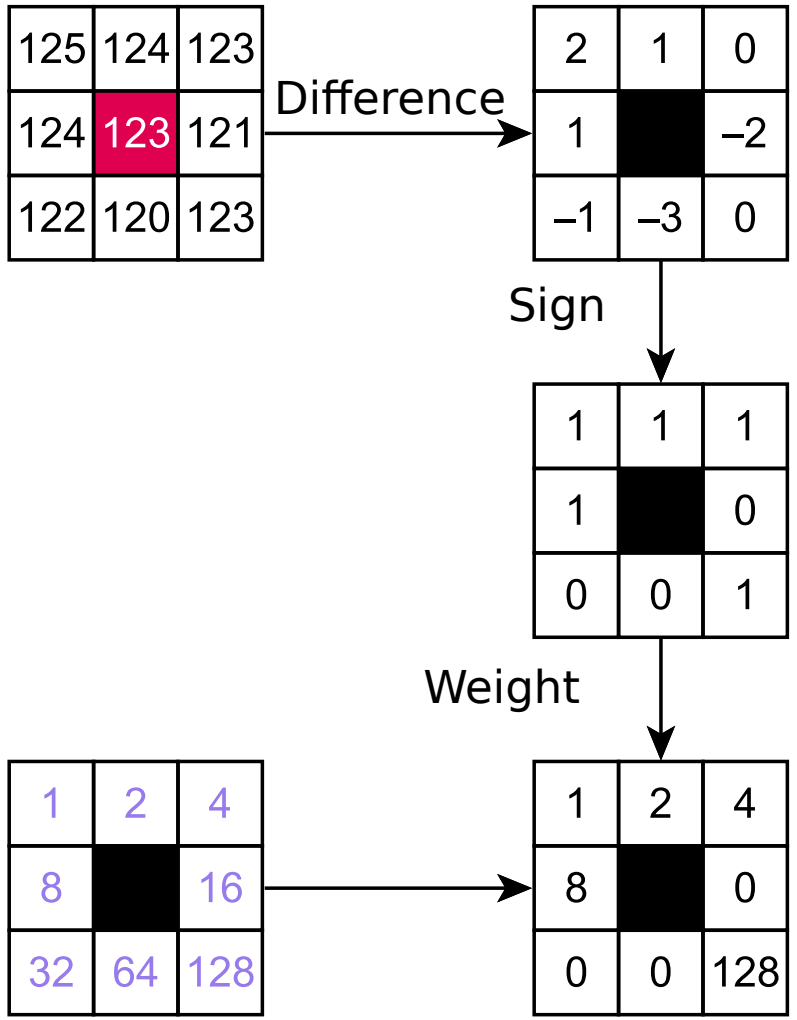


LBP = 1 + 2 + 4 + 8 + 128 = 143

LBP: Local Binary Patterns

Construction

- Choose a window size
- Each pixel in the neighborhood
- Compare to its neighbors
- Create a binary number (converted to decimal)

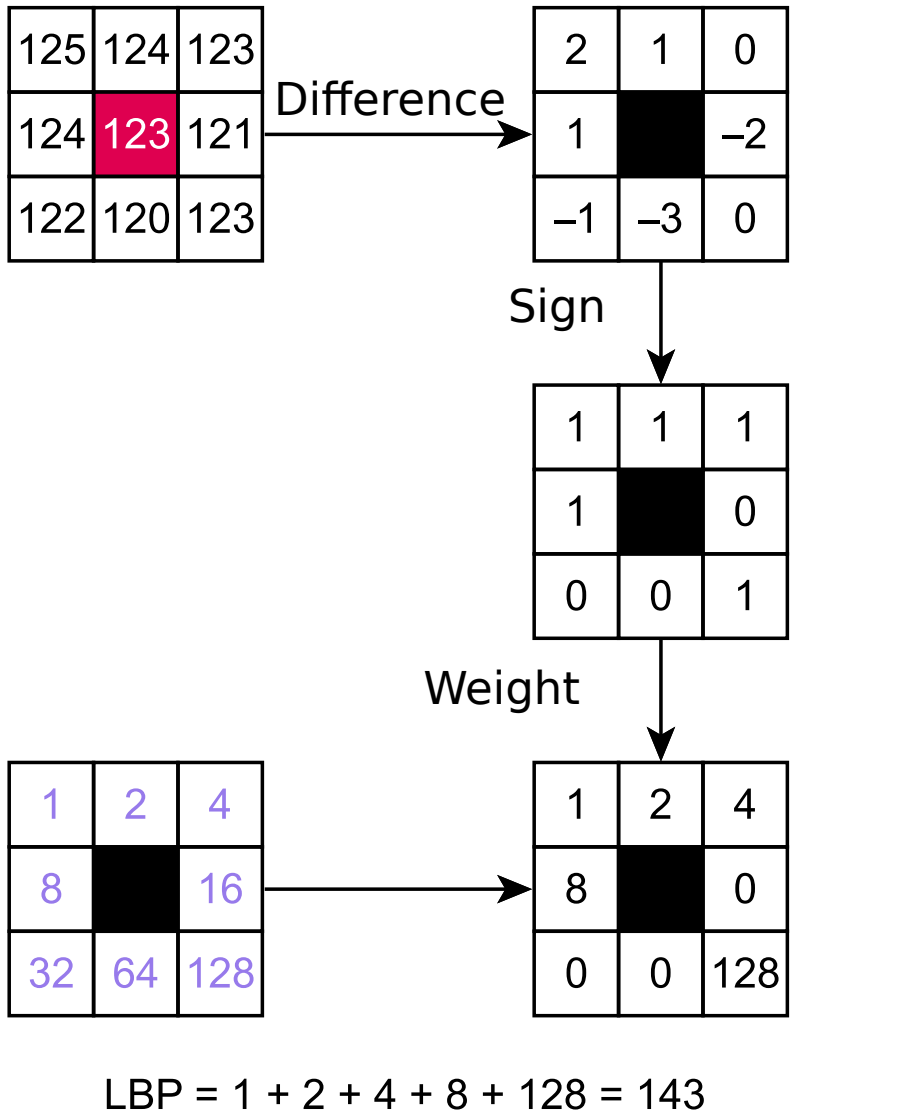


LBP = 1 + 2 + 4 + 8 + 128 = 143

LBP: Local Binary Patterns

Construction

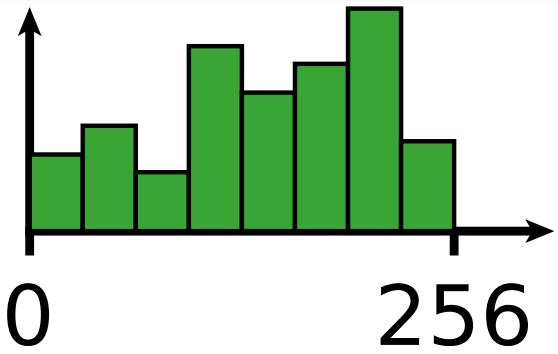
- Choose a window size
- Each pixel in the neighborhood
- Compare to its neighbors
- Create a binary number (converted to decimal)
- Compute the histogram of all numbers in window

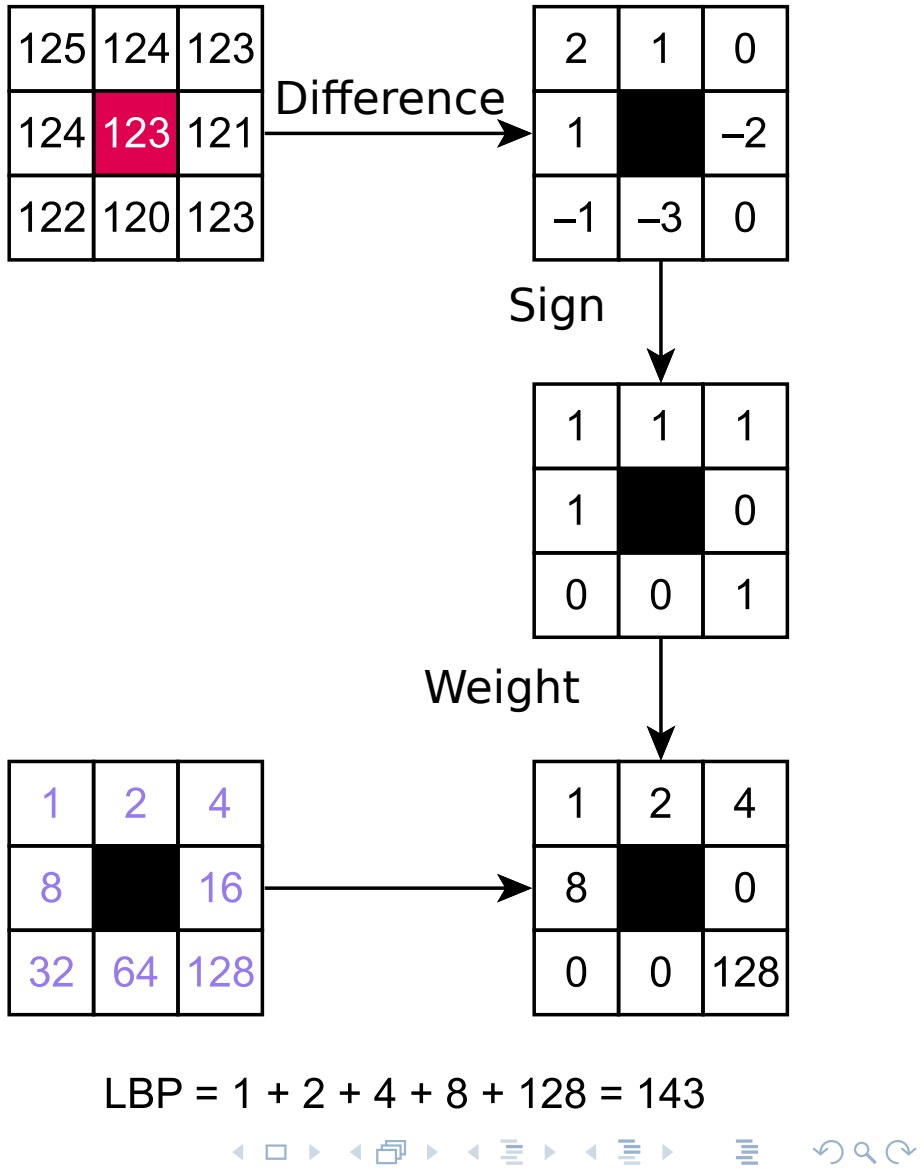


LBP: Local Binary Patterns

Construction

- Choose a window size
- Each pixel in the neighborhood
- Compare to its neighbors
- Create a binary number (converted to decimal)
- Compute the histogram of all numbers in window



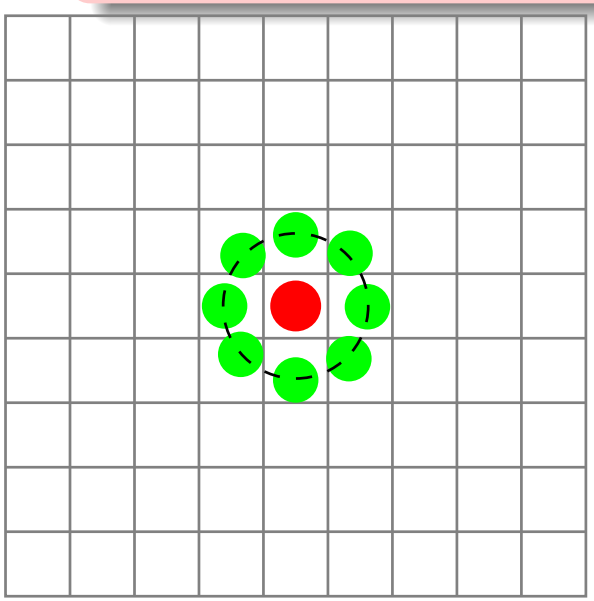


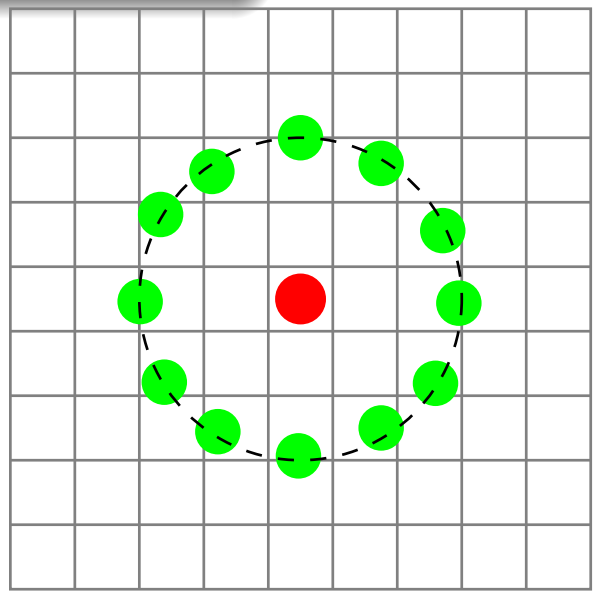
LBP

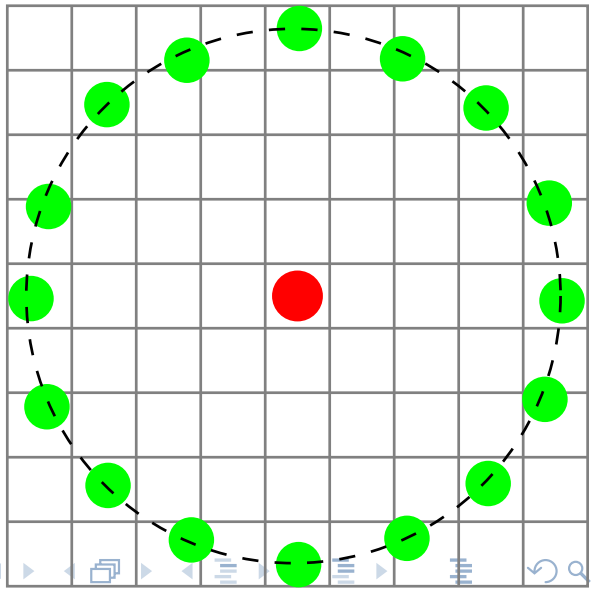
- Tolerance to illumination changes
- Computational simplicity
- Sensitive to noise
- Sensitive to scale
- Tolerant to orientation

Neighborhood

- Not classical neighborhoods
- Bilinear interpolation
- Simplicity: nearest pixel







Other binary descriptors

- BRIEF: Binary Robust Independant Elementary Features
- BRISK: Binary Robust Invariant Scalable Keypoints
- ...

Section 4

Feature matching

Feature matching

- 2 or more sets P, Q...of keypoints $p_i \in P, q_j \in Q$...
- associated descriptors: $\phi(p_i), \phi(q_j)$

Distance between features

$$d(p_i, q_j) = \|\phi(p_i) - \phi(q_j)\|$$

Nearest neighbor

of point p_0 in set Q

$$q_0 = \underset{q_j \in Q}{\operatorname{argmin}} d(p_0, q_j)$$

Solutions

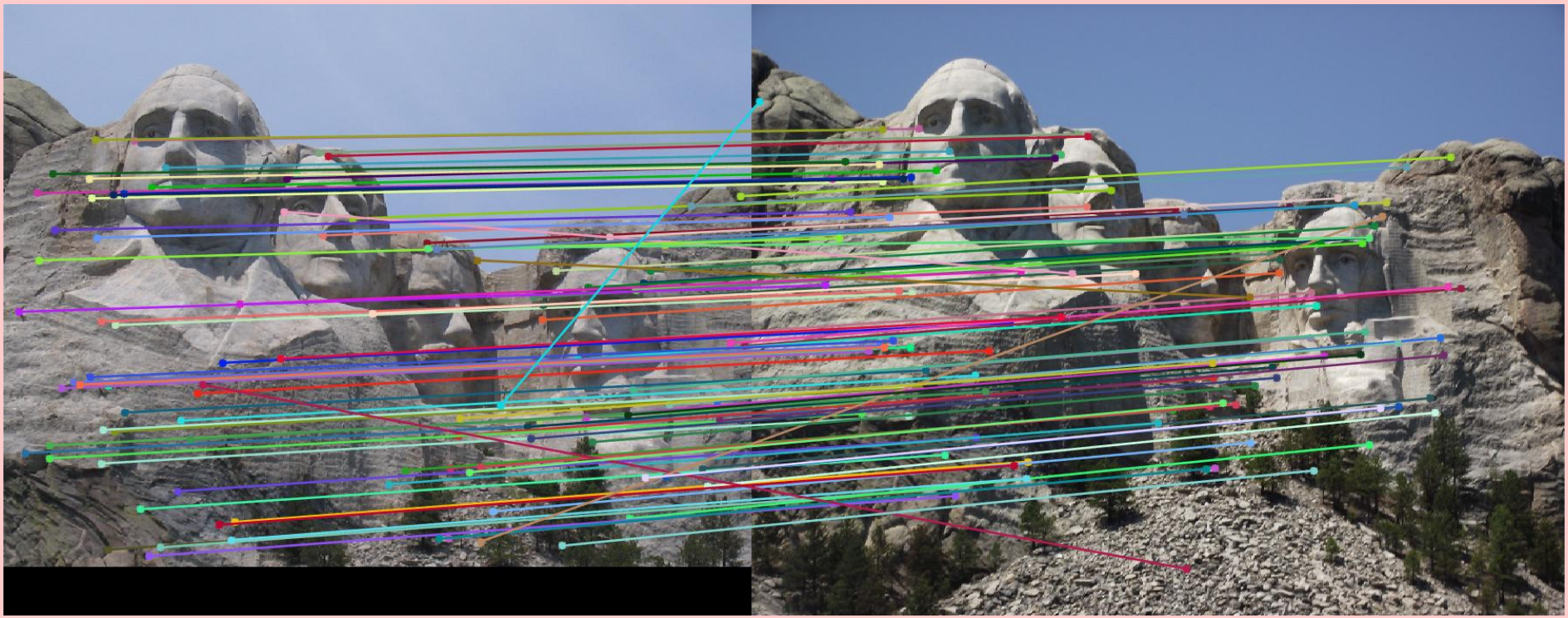
- Kd forests / Kd trees
- Nearest Neighbors, Approximate NN

Binary features

- not appropriate
- Hamming distance XOR

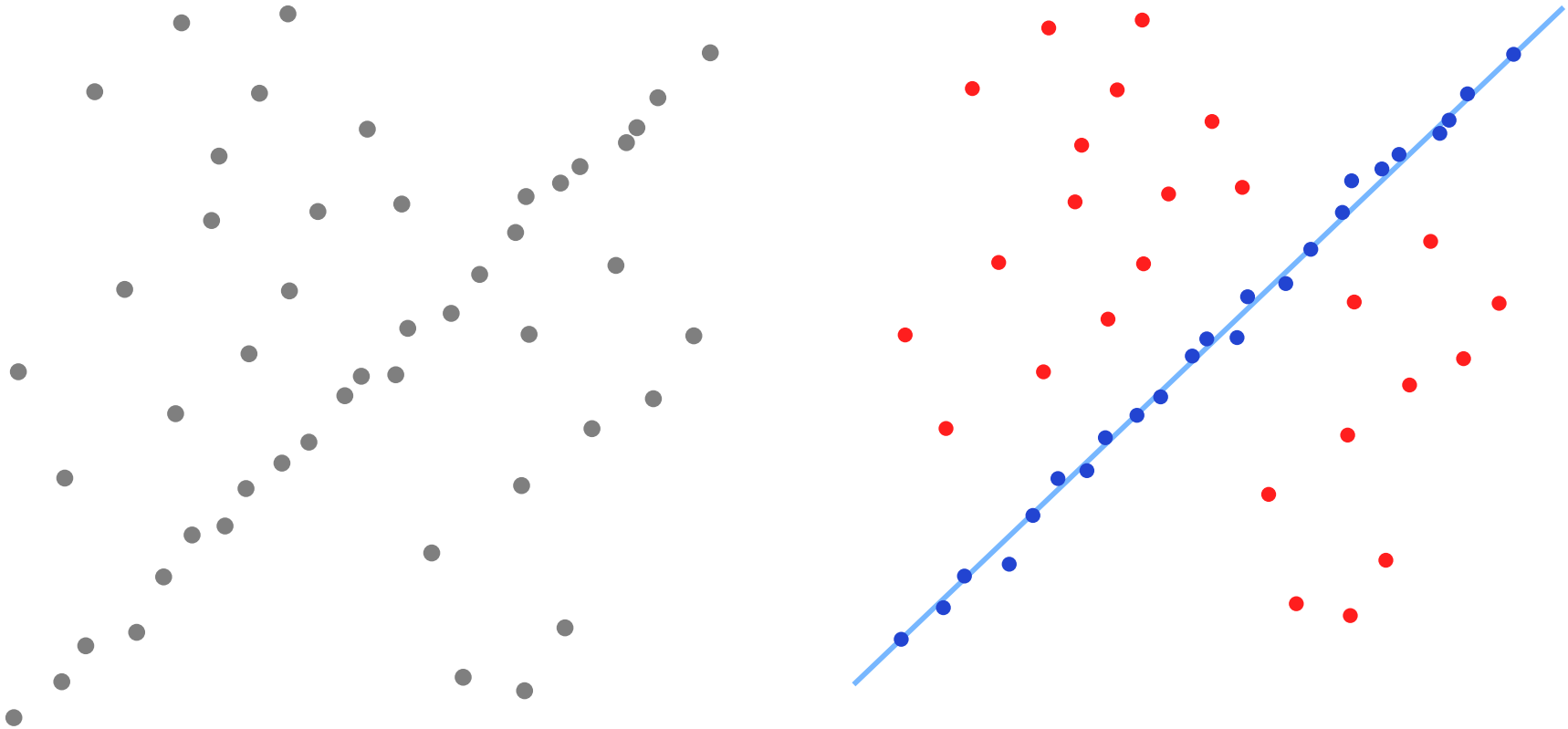
Feature matching

Problem: coherence between matchs?



Remove outliers: RANSAC

RANdom SAmple Consensus



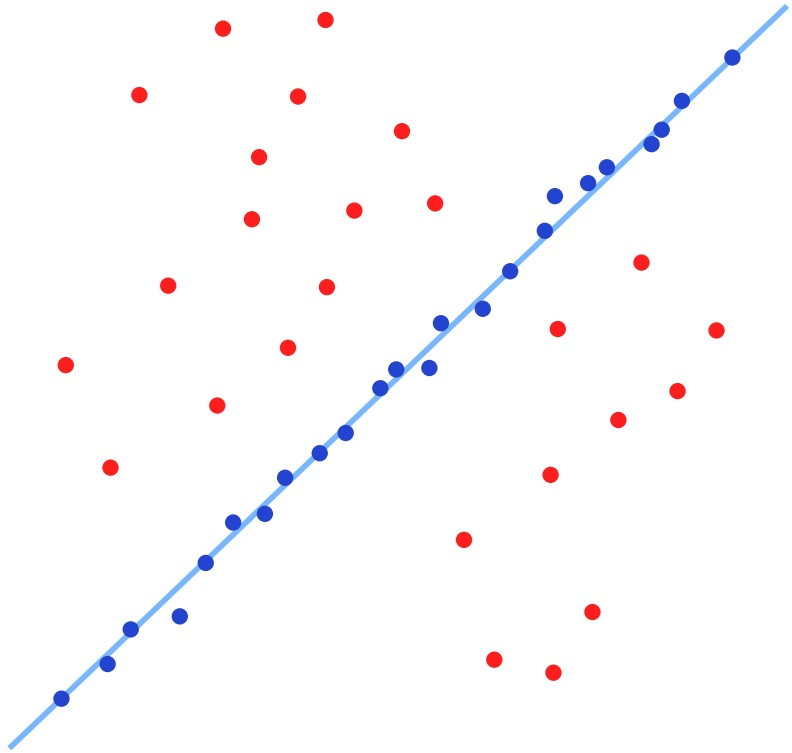
Remove outliers: RANSAC

RANdom SAmple Consensus

- Least Squares method would be inefficient
- RANSAC intends to remove outliers

Principle

- random selection of subset of data
- fit hypothesis
- verify vs all data
- iterate until enough inliers



RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)

RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)
2. Features descriptors (SIFT, SURF...)

RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)
2. Features descriptors (SIFT, SURF...)
3. Features matching (ANN, Kd trees...)

RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)
2. Features descriptors (SIFT, SURF...)
3. Features matching (ANN, Kd trees...)
4. Transform estimation (RANSAC)

RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)
2. Features descriptors (SIFT, SURF...)
3. Features matching (ANN, Kd trees...)
4. Transform estimation (RANSAC)

Elements of choice

RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)
2. Features descriptors (SIFT, SURF...)
3. Features matching (ANN, Kd trees...)
4. Transform estimation (RANSAC)

Elements of choice

- Tolerance to variations (deformations, scales...)

RANSAC: application to image registration

- Transform estimation via RANSAC

Image registration

1. Features detection (keypoints, Harris, FAST...)
2. Features descriptors (SIFT, SURF...)
3. Features matching (ANN, Kd trees...)
4. Transform estimation (RANSAC)

Elements of choice

- Tolerance to variations (deformations, scales...)
- Computation speed / real-time applications

Augmented reality


T. Lindeberg.
Scale selection properties of generalized scale-space interest point detectors.
Journal of Mathematical Imaging and Vision, 46(2):177–210, Jun 2013.


E. Rosten and T. Drummond.
Fusing points and lines for high performance tracking.
In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1508–1515. IEEE, 2005.


E. Rosten, R. Porter, and T. Drummond.
Faster and better: A machine learning approach to corner detection.
IEEE Trans. Pattern Analysis and Machine Intelligence, 32:105–119, 2010.