Keypoints and Descriptors[1]

Lecture 22

See Sections 9.2 and 9.3 in
Reinhard Klette: Concise Computer Vision
Springer-Verlag, London, 2014

[1]See last slide for copyright information.

Agenda

1. Speeded-Up Robust Features
2. Oriented Robust Binary Features
3. Evaluation of Features

SURF Masks and the Use of Integral Images

Detector *speeded-up robust features* (SURF) follows similar ideas as SIFT

Designed for better run-time performance

Utilizes the integral images I_{int} and simplifies filter kernels

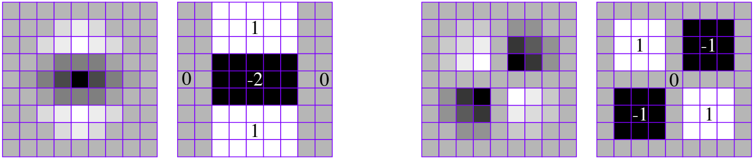


Illustration for $\sigma = 1.2$, the lowest scale, and 9×9 discretised and cropped Gaussian second-order partial derivatives and corresponding filter kernels in SURF

SURF Masks

Two of the four used masks (or filter kernels) are illustrated above

SURF's masks for x-direction and the other diagonal direction are analogously defined

Size of the mask corresponds to the chosen scale

After 9×9, SURF uses then masks of sizes 15×15, 21×21, 27×27, and so on

Values in Filter Kernels

Values in those filter kernels are either 0, -1, +1, or -2

Values -1, +1, and +2 are constant in rectangular subwindows W of the mask

This allows us to use integral images for calculating time-efficiently the sum S_W of all intensity values in W

It only remains to multiply the sum S_W with the corresponding coefficient (i.e., value -1, +1, or -2)

Sum of those three or four products is then the convolution result at the given reference pixel for one of the four masks

Scales and Keypoint Detection

Value $\sigma = 1.2$ is chosen for the lowest scale (i.e. highest spatial resolution) in SURF

Convolutions at a pixel location p in input image I with four masks approximate the four coefficients of the Hessian matrix

Four convolution masks produce values $D_{xx}(p,\sigma)$, $D_{xy}(p,\sigma)$, assumed to be equal to $D_{yx}(p,\sigma)$, and $D_{yy}(p,\sigma)$

$$S(p,\sigma) = D_{xx}(p,\sigma) \cdot D_{yy}(p,\sigma) - [c_\sigma \cdot D_{xy}(p,\sigma)]^2$$

as an approximate value for the determinant of the Hessian matrix at scale σ

With $0 < c_\sigma < 1$ is a weighting factor which could be optimized for each scale; SURF uses constant $c_\sigma = 0.9$

Keypoint p detected by a local maximum of a value $S(p,\sigma)$ within a $3 \times 3 \times 3$ array of S-values, analogously to keypoint detection in LoG or DoG scale space

SURF Descriptor

SURF descriptor is a 64-vector of floating point values

Combines local gradient information, similar to the SIFT descriptor

Uses weighted sums in rectangular subwindows (known as *Haar-like features*

Windows around the keypoint for simple and more time-efficient approximation of gradient values

Agenda

1. Speeded-Up Robust Features
2. Oriented Robust Binary Features
3. Evaluation of Features

FAST, BRIEF, ORB

Oriented robust binary features (ORB) based on *binary robust independent elementary features* (BRIEF) and keypoint detector FAST; both together characterize ORB

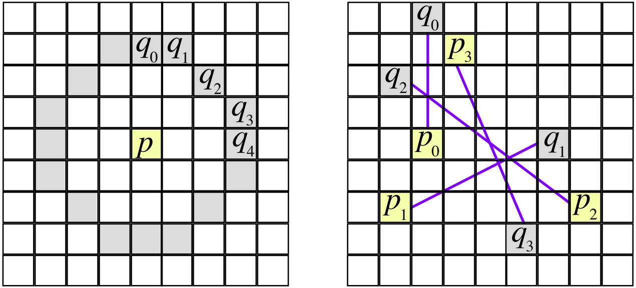
Binary Patterns

BRIEF reduces a keypoint descriptor from a 128-vector (such as defined for SIFT) to just 128 bits

Given floating-point information is binarized into a much simpler representation

Same idea has been followed when designing the census transform, when using *local binary patterns* (LBPs), or when proposing simple tests for training a set of classification trees

LBP and BRIEF



Pixel location p and 16 pixel locations q around p; $s(p,q) = 1$ if $I(p) - I(q) > 0$; 0 otherwise;
$s(p,q_0)\cdot 2^0 + s(p,q_1)\cdot 2^1 + \ldots + s(p,q_15)\cdot 2^{15}$ is LBP code at p

BRIEF uses an order of random pairs of pixels within a square neighborhood; here four pairs (p_i, q_i), defining
$s(p_0,q_0)\cdot 2^0 + s(p_1,q_1)\cdot 2^1 + s(p_2,q_2)\cdot 2^2 + s(p_3,q_3)\cdot 2^3$

BRIEF

LBP defined for a selection of n pixel pairs (p, q), selected around the current pixel in some defined order in a $(2k+1) \times (2k+1)$ neighborhood (e.g., $k = 4$ to $k = 7$)

After Gaussian smoothing defined by $\sigma > 0$ in the given image I

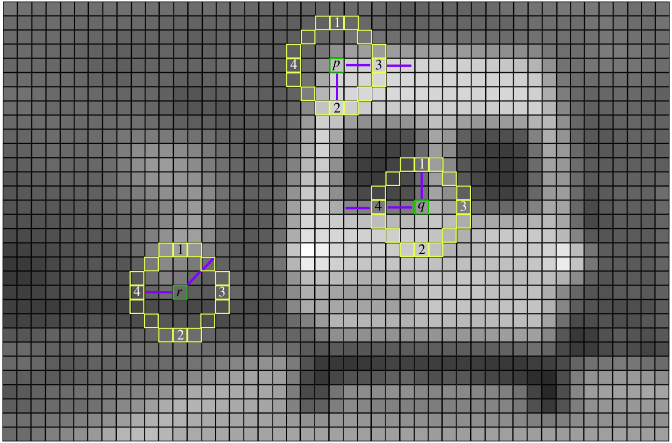
Order of those pairs, parameters k and σ define a BRIEF descriptor

Smoothing can be minor (i.e. a small σ) and the original paper suggested a random order for pairs of pixel locations

Scale or rotation invariance was not intended by the designers of the original BRIEF

FAST

Features from an accelerated segment test FAST: corner by considering image values on digital circle around given p



16 image values on a circle of radius $\rho = 3$

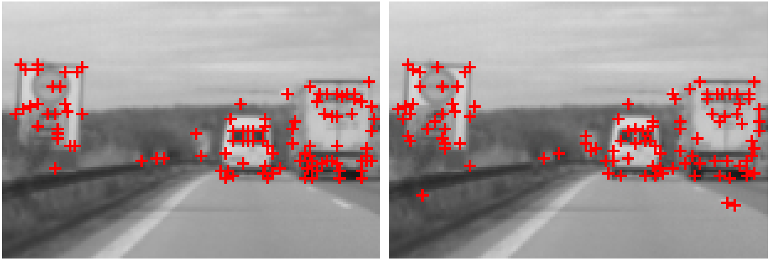
Cornerness test

Pixels p, q, and r are at intersections of edges; directions of those edges are indicated by the shown blue lines

Cornerness test

Value at center pixel needs to be darker (or brighter) compared to more than 8 (say, 11 for really identifying a corner and not just an irregular pixel on an otherwise straight edge) subsequent pixels on the circle, and "similar" to values of remaining pixels on circle

Harris versus FAST



Left: Detected corners using the Harris detector
Right: Corners detected by FAST.

Corner Detector by Harris and Stephens

This *Harris detector* uses first-order derivatives of smoothed $L(.,.,\sigma)$, for some $\sigma > 0$

$$\mathbf{G}(p,\sigma) = \begin{bmatrix} L_x^2(p,\sigma) & L_x(p,\sigma)L_y(p,\sigma) \\ L_x(p,\sigma)L_y(p,\sigma) & L_y^2(p,\sigma) \end{bmatrix}$$

Eigenvalues λ_1 and λ_2 of $\mathbf{G}$ represent changes in intensities in orthogonal directions in image I

For small $a > 0$ (e.g., $a = 1/25$) consider *cornerness measure*

$$\mathcal{H}(p,\sigma,a) = \det(\mathbf{G}) - a \cdot \mathrm{Tr}(\mathbf{G})$$

$$\mathcal{H}(p,\sigma,\lambda) = \lambda_1\lambda_2 - a \cdot (\lambda_1 + \lambda_1)$$

One large and one small eigenvalue (such as on a step edge), then $\mathcal{H}(p,\sigma,a)$ remains reasonably small

Back to FAST: Time Efficiency

First: compare value at center pixel against values at locations 1, 2, 3, and 4 in this order

If still possible that the center pixel passes the cornerness test, we continue with testing more pixels on the circle

Original FAST paper proposes to learn a decision tree for time optimization

FAST detector in `OpenCV` (and also the one in `libCVD`) applies SIMD instructions for concurrent comparisons, which is faster then the use of the originally proposed decision tree

Non-maxima Suppression

For a detected corner, calculate maximum difference T between value at center pixel and values on discrete circle being classified as "darker" or "brighter" such that we still detect this corner

Non-maxima suppression deletes then in the order of differences T

ORB

ORB also for *oriented FAST and rotated BRIEF*

Combines keypoints defined by extending FAST and an extension of descriptor BRIEF

1. Multi-scale detection following FAST (for scale invariance), calculates a dominant direction
2. Applies calculated direction for mapping BRIEF descriptor into a steered BRIEF descriptor (for rotation invariance)

Authors of ORB suggest ways for analyzing variance and correlation of components of steered BRIEF descriptor

Test data base can be used for defining a set of BRIEF pairs (p_i, q_i) which de-correlate the components of the steered BRIEF descriptor for improving the discriminative performance of the calculated features

Multi-Scale, Harris Filter, and Direction

Define discrete circle of radius $\rho = 9$

(Above: FAST illustrated for discrete circle of radius $\rho = 3$)

Scale pyramid of input image is used for detecting FAST keypoints at different scales

Harris filter. Use cornerness measure (of Harris detector) to select T "most cornerness" keypoints at those different scales, where $T > 0$ is a pre-defined threshold for numbers of keypoints

Moments m_{10} and m_{01} of the disk S, defined by radius ρ, specify direction

$$\theta = \mathrm{atan2}(m_{10}, m_{01})$$

By definition of FAST it can be expected that $m_{10} \neq m_{01}$

Let $\mathbf{R}_\theta$ be the 2D rotation matrix about angle θ

Descriptor with a Direction

Pairs (p_i, q_i) for BRIEF, with $0 \leq i \leq 255$, are selected by a Gaussian distribution within the disk used (of radius ρ)

Form matrix **S** which is rotated into

$$\mathbf{S}_\theta = \mathbf{R}_\theta \mathbf{S} = \mathbf{R}_\theta \begin{bmatrix} p_0 & \cdots & p_{255} \\ q_0 & \cdots & q_{255} \end{bmatrix} = \begin{bmatrix} p_{0,\theta} & \cdots & p_{255,\theta} \\ q_{0,\theta} & \cdots & q_{255,\theta} \end{bmatrix}$$

Steered BRIEF descriptor calculated as the sum $s(p_{0,\theta}, q_{0,\theta}) \cdot 2^0 + \ldots + s(p_{255,\theta}, q_{255,\theta}) \cdot 2^{255}$, where s is defined ias above

By going from original BRIEF to the steered BRIEF descriptor, values in the descriptor become more correlated

256 BRIEF Pairs

For time-efficiency reasons, a used pattern of 256 BRIEF pairs (generated by a Gaussian distribution) is rotated in increments of $2\pi/30$, and all those patterns are stored in a look-up table

This eliminates the need for an actual rotation; the calculated θ is mapped on the nearest multiple of $2\pi/30$

Agenda

Evaluation of Features

Evaluate feature detectors with respect to invariance properties

Caption

Rotated image; the original frame from sequence `bicyclist` from EISATS is 640×480 and recorded at 10 bit per pixel

Demagnified image

Uniform brightness change

Blurred image

Feature Evaluation Test Procedure

Four changes: rotation, scaling, brightness changes, and blurring; select sequence of frames, feature detector and do

1. Read next frame I, which is a gray-level image
2. Detect keypoints p in I and their descriptors $\mathbf{d}(p)$ in I
3. Let N_k be the number of keypoints p in I
4. For given frame, generate four image sequences
    1. Rotate I around its center in steps of 1 degree
    2. Resize I in steps of 0.01, from 0.25 to 2 times the original size
    3. Add scalar to pixel values in increments of 1 from -127 to 127
    4. Apply Gaussian blur with increments of 2 for σ from 3 to 41
5. Feature detector again: keypoints p_t and descriptors $\mathbf{d}(p_t)$
6. $N_t =$ number of keypoints p_t for transformed image
7. Descriptors $\mathbf{d}(p)$ and $\mathbf{d}(p_t)$ to identify matches between features in I and I_t
8. Use RANSAC to remove inconsistent matches
9. $N_m =$ number of detected matches

Repeatability Measure

Repeatability $\mathcal{R}(I, I_t)$

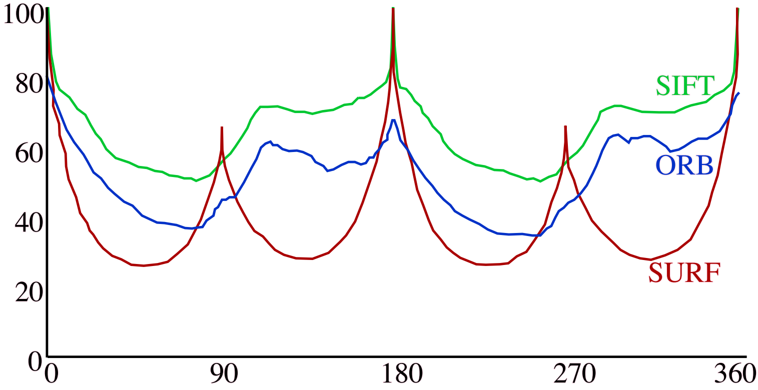
Ratio of number of detected matches to number of keypoints in the original image

$$\mathcal{R}(I, I_t) = \frac{N_m}{N_k}$$

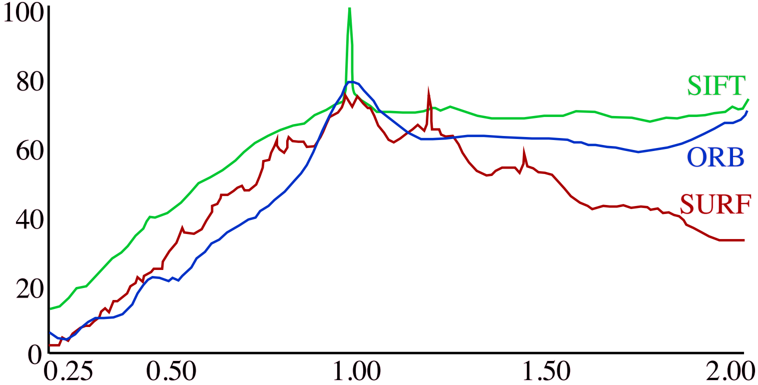
Report means for selected frames in test sequences

Use `OpenCV` default parameters for the studied feature detectors and a set of 90 randomly selected test frames

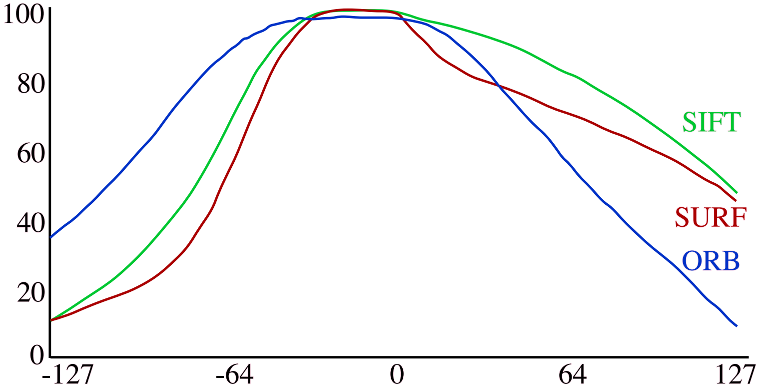
Repeatability Diagram For Rotation



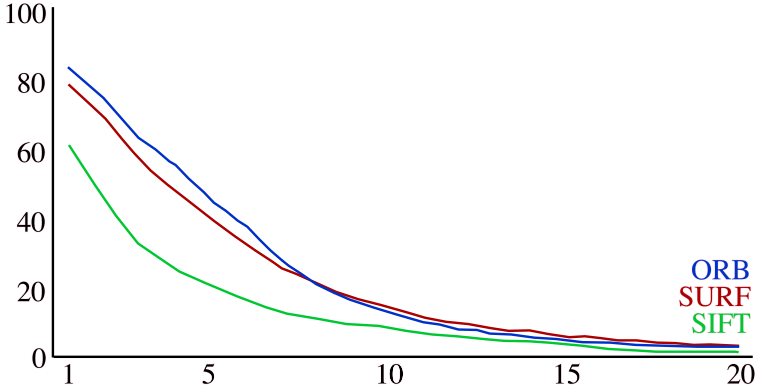
Repeatability Diagram For Scaling



Repeatability Diagram For Brightness Variation



Repeatability Diagram For Blurring



Discussion of the Experiments

Invariance has certainly its limits

If scaling, brightness variation, or blurring pass some limits then we cannot expect repeatability anymore

Rotation is a different case; here we could expect invariance close to the ideal case (of course, accepting that digital images do not rotate as continuous 2D functions in $\mathbb{R}^2$

Detector	Time per frame	Time per keypoint	Number N_k
SIFT	254.1	0.55	726
SURF	401.3	0.40	1,313
ORB	9.6	0.02	500

Mean values for 90 randomly selected input frames

Third column: numbers of keypoints for the frame used for generating the transformed images

Summary

SIFT is performing well (compared to SURF and ORB) for rotation, scaling, and brightness variation, but not for blurring

All results are far from the ideal case of invariance

If there is only a minor degree of brightness variation or blurring, then invariance can be assumed

Rotation or scaling leads already to significant drops in repeatability for small angles of rotation, or minor scale changes

There was no significant run-time difference between SIFT and SURF

There was a very significant drop in computation time for ORB, which appears (judging from this comparison) as a fast and reasonably competitive feature detector

Copyright Information