# Aggregating local descriptors into a compact representation

Hervé Jégou[1], Matthijs Douze[2], Cordelia Schmid[2] and Patrick Pérez[3]

1: INRIA Rennes, TEXMEX team, France

2: INRIA Grenoble, LEAR team, France

3: Technicolor, France


## Problem setup: Image indexing

- Retrieval of images representing the same object/scene:
  - different viewpoints, backgrounds, …
  - copyright attacks: cropping, editing, …
  - short response time
  - **billions** of images

queries

relevant answers

INRIA

## Related work on large scale image search

- Most systems build upon the BoF framework [Sivic & Zisserman 03]
  - Large (hierarchical) vocabularies [Nister Stewenius 06]
  - Improved descriptor representation [Jégou et al 08, Philbin et al 08]
  - Geometry used in index [Jégou et al 08, Perdoc'h et al 09]
  - Query expansion [Chum et al 07]
  - …

→ memory tractable for a few million images only

- Efficiency improved by
  - Min-hash and Geometrical min-hash [Chum et al. 07-09]
  - compressing the BoF representation [Jégou et al. 09]

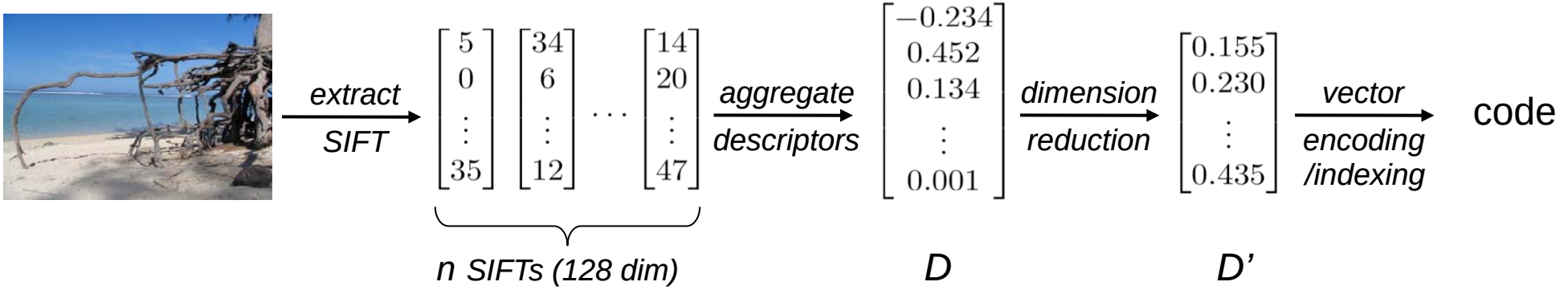
But still hundreds of bytes are required to obtain a "reasonable quality"

- Alternative: GIST descriptors with Spectral Hashing or similar techniques

→ very limited invariance to scale/rotation/crop

# Objective and proposed approach

- Aim: optimizing the trade-off between
  - search quality
  - search speed
  - memory usage
- Approach: joint optimization of three stages
  - local descriptor aggregation
  - dimension reduction
  - indexing algorithm



## Aggregation of local descriptors

- Problem: represent an image by a single fixed-size vector:

set of *n* local descriptors → 1 vector

- Most popular idea: BoF representation [Sivic & Zisserman 03]
  - sparse vector
  - highly dimensional

→ strong dimensionality reduction introduces loss

- Alternative: Fisher Kernels [Perronnin et al 07]
  - non sparse vector
  - excellent results with a small vector dimensionality

  → our method in the spirit of this representation

INRIA

technicolor

# VLAD : Vector of Locally Aggregated Descriptors

- Learning: *k*-means
  - output: *k* centroids : $c_1,\ldots,c_i,\ldots c_k$

- VLAD computation:

① $$c(x) = \arg\min_{c_i} ||c_i - x||^2$$

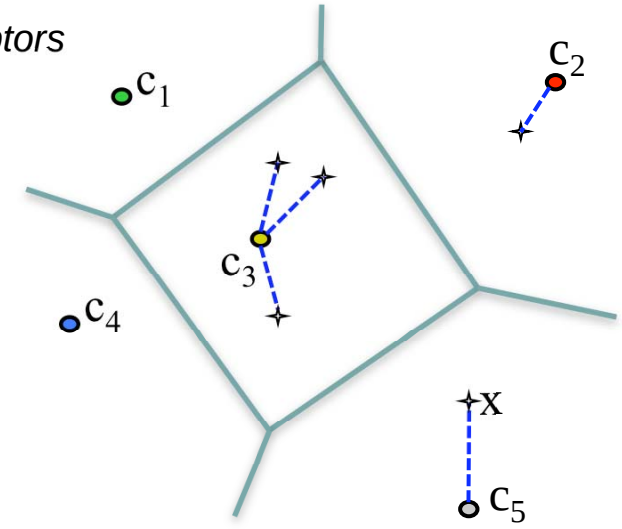
②③ $$v_i = \sum_{x:c(x)=c_i} x - c_i$$

$$v = [v_1, \ldots, v_i, \ldots, v_k], \; v_i \in \mathbb{R}^{128}$$

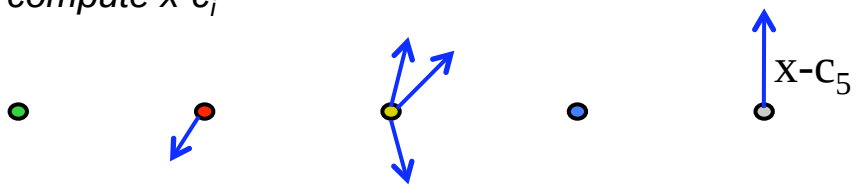
⇒ dimension *D = k *128*

- L2-normalized
- Typical parameter: k=64 (D=8192)

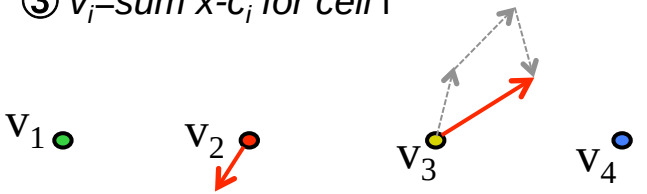
① *assign descriptors*




② *compute x-c$_i$*



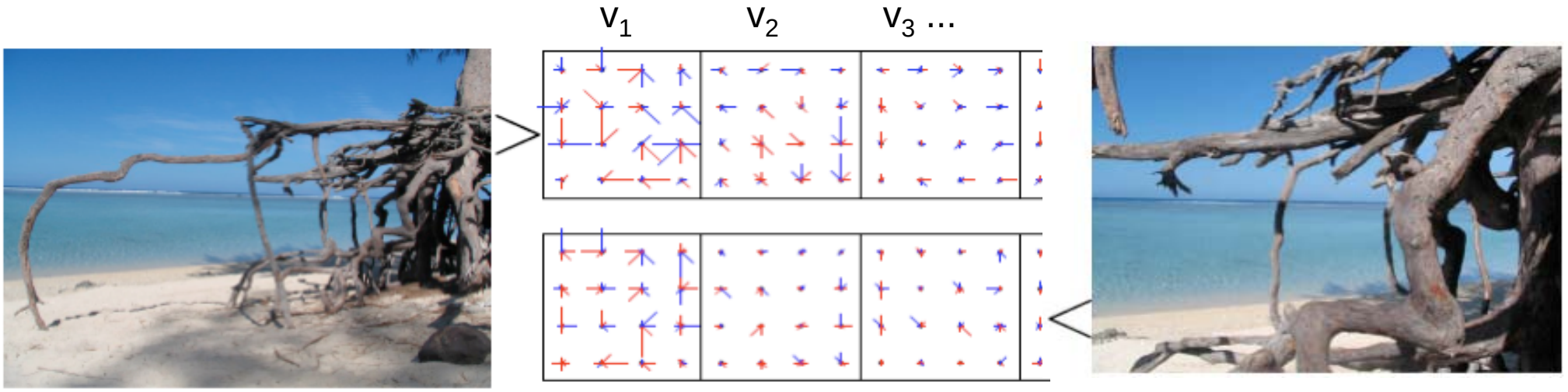

③ *v$_i$=sum x-c$_i$ for cell* i




INRIA

technicolor

# VLADs for corresponding images




*SIFT-like representation per centroid (+ components: blue, - components: red)*

- good coincidence of energy & orientations

# VLAD performance and dimensionality reduction

- We compare VLAD descriptors with BoF: INRIA Holidays Dataset (mAP,%)
- Dimension is reduced to from D to D' dimensions with PCA

| Aggregator | k | D | D'=D (no reduction) | D'=128 | D'=64 |
|---|---|---|---|---|---|
| BoF | 1,000 | 1,000 | 41.4 | 44.4 | 43.4 |
| BoF | 20,000 | 20,000 | 44.6 | 45.2 | 44.5 |
| BoF | 200,000 | 200,000 | 54.9 | 43.2 | 41.6 |
| VLAD | 16 | 2,048 | 49.6 | 49.5 | **49.4** |
| VLAD | 64 | 8,192 | 52.6 | **51.0** | 47.7 |
| VLAD | 256 | 32,768 | **57.5** | 50.8 | 47.6 |

- Observations:
  - VLAD better than BoF for a given descriptor size
    → comparable to Fisher kernels for these operating points
  - Choose a small D if output dimension D' is small

INRIA

technicolor

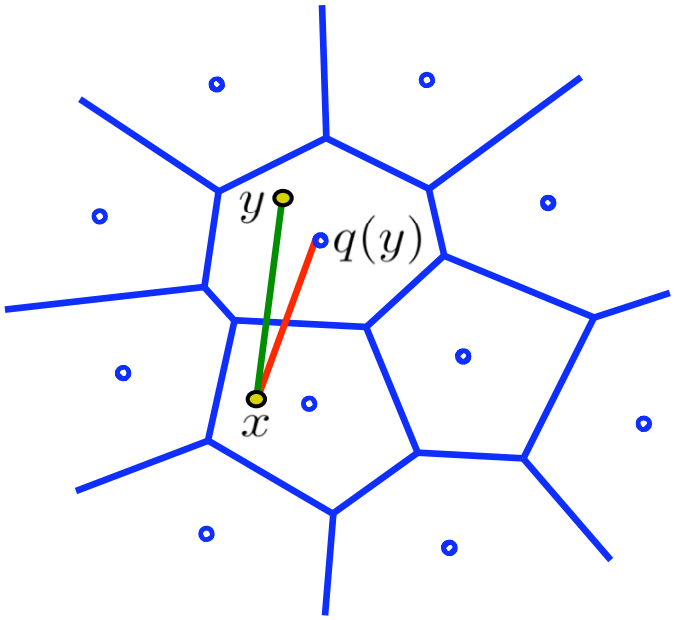
## Indexing algorithm: searching with quantization [Jegou et al. 10]

- Search/Indexing = distance approximation problem
- The distance between a query vector *x* and a database vector y is estimated by

$$d(x, y) \approx d(x, q(y))$$

where q(.) is a quantizer

→ vector-to-code distance




- The choice of the quantizer is critical
  - needs many centroids
  - regular k-means and approximate k-means can not be used
    → we typically want $k=2^{64}$ for 64-bit codes

# Product quantization for nearest neighbor search

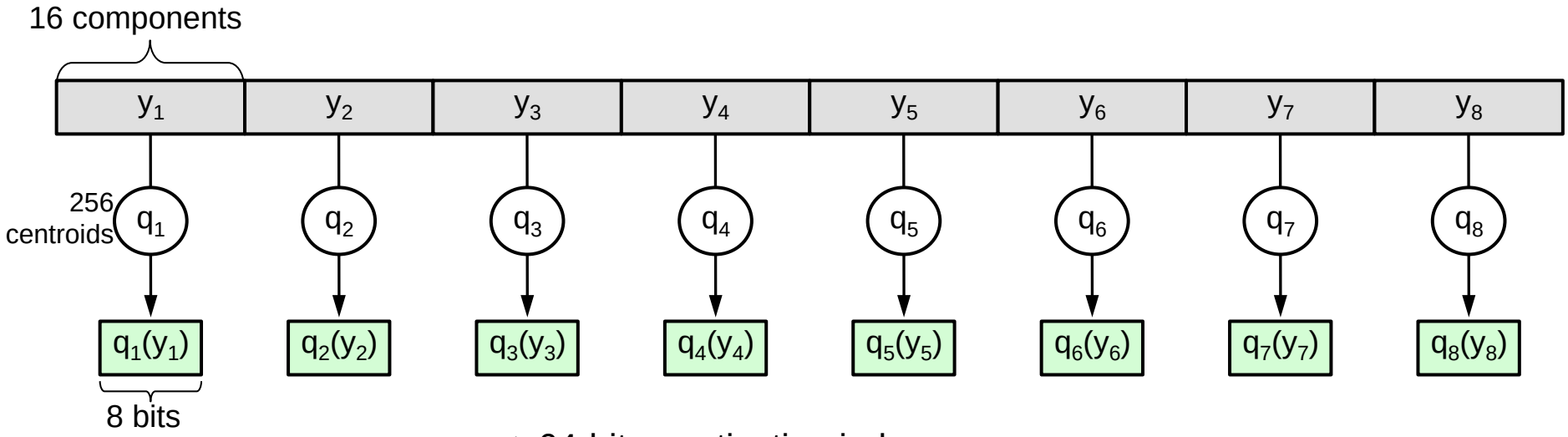
- Vector split into *m* subvectors: $y \rightarrow \left[y_1 | \ldots | y_m\right]$

- Subvectors are quantized separately by quantizers

$$q(y) = \left[q_1(y_1) | \ldots | q_m(y_m)\right]$$

where each $q_i$ is learned by *k*-means with a limited number of centroids

- Example: y = 128-dim vector split in 8 subvectors of dimension 16




⇒ 64-bit quantization index

INRIA

technicolor

## Product quantizer: asymmetric distance computation (ADC)

- Compute the square distance approximation in the compressed domain

$$d(x, y)^2 \approx \sum_{i=1}^{m} d(x_i, q_i(y_i))^2$$

- To compute distance between query $x$ and many codes
  - compute $d(x_i, c_{i,j})^2$ for each subvector $x_i$ and all possible centroids

  → stored in look-up tables
  - for each database code: sum the elementary square distances

- Each 8x8=64-bits code requires only **m=8 additions per distance**!

- IVFADC: combination with an inverted file to avoid exhaustive search

INRIA

technicolor

## Optimizing the dimension reduction and quantization together

- VLAD vectors suffer two approximations
  - mean square error from PCA projection: $e_p(D')$
  - mean square error from quantization: $e_q(D')$

- Given k and bytes/image, choose D' minimizing their sum

Ex, k=16:

| D' | $e_p(D')$ | $e_q(D')$ | $e_p(D')+e_q(D')$ |
|---|---|---|---|
| 32 | 0.0632 | 0.0164 | 0.0796 |
| 48 | 0.0508 | 0.0248 | 0.0757 |
| 64 | 0.0434 | 0.0321 | **0.0755** |
| 80 | 0.0386 | 0.0458 | 0.0844 |

INRIA

technicolor

## Results on standard datasets

- Datasets
  - University of Kentucky benchmark — score: nb relevant images, max: 4
  - INRIA Holidays dataset — score: mAP (%)

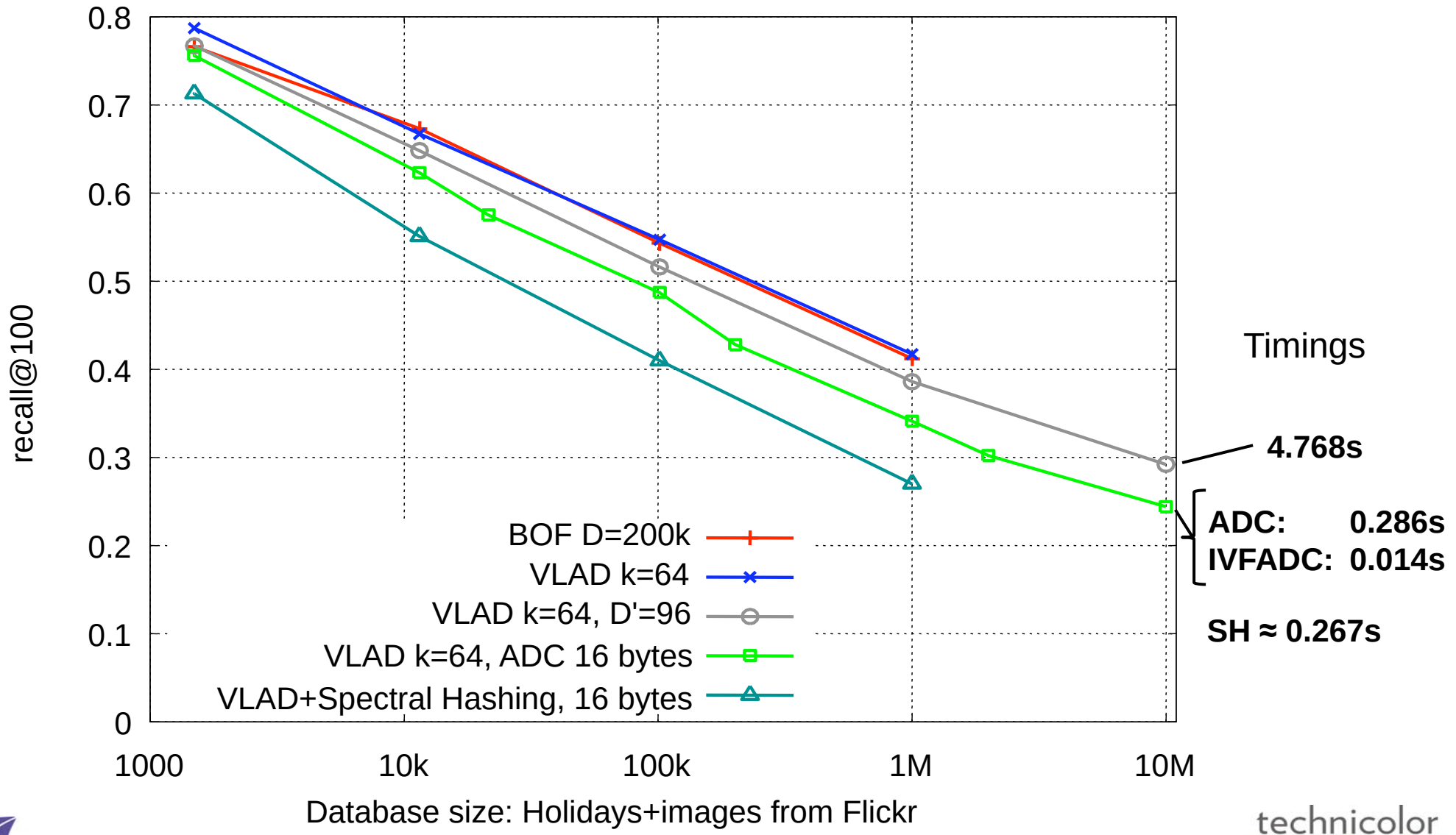
| Method | bytes | UKB | Holidays |
|---|---|---|---|
| BoF, k=20,000 | 10K | 2.92 | 44.6 |
| BoF, k=200,000 | 12K | 3.06 | 54.9 |
| miniBOF | 20 | 2.07 | 25.5 |
| miniBOF | 160 | 2.72 | 40.3 |
| VLAD k=16, ADC | **16** | **2.88** | **46.0** |
| VLAD k=64, ADC | **40** | **3.10** | **49.5** |

miniBOF: “Packing Bag-of-Features”, ICCV’09

INRIA

technicolor

# Large scale experiments (10 million images)




INRIA

technicolor

## Conclusion

- Competitive search accuracy with a few dozen bytes per indexed image
- Tested on up to 220 million video frames
  - extrapolation for 1 billion images: 20GB RAM, query < 1s on 8 cores
- Matlab package available, includes:
  - VLAD
  - Indexing algorithm (ADC/IVFADC)
  - extracted descriptors
- Improved Fisher kernels by Perronnin et al., CVPR'2010
- 10 million images indexed on my laptop:


INRIA

technicolor

END