# Machine learning techniques for video content analysis

**Frédéric Precioso**
ETIS UCP, FRANCE
precioso@ensea.fr
and
**Matthieu Cord**
LIP6, Univ. UPMC-PARIS 6, FRANCE
matthieu.cord@lip6.fr

December 3, 2009

## Machine learning techniques for video content analysis

- Fast k-NN clustering for Video Summarization
- Machine Learning for Shot Boundary Detection
- Active learning for video retrieval
- Kernel design and data representation for actor retrieval
- Approximate k-NN for fast similarity approximation
- Optimization active learning on large scale databases

## Next Multimedia Challenges

Part of this year ACM Multimedia conference, the Multimedia Grand Challenge 2009 aimed at collecting information on the specific problems and issues companies like Google, Yahoo, Nokia, HP, Radvision and CeWe see arising on the multimedia horizon for the next 2-5 years.
The six companies have put forward what they consider as challenges for multimedia processing research.

Yahoo! put forward this chalenge:

**Robust Automatic Segmentation of Video According to Narrative Themes**

The challenge to researchers in the multi-media community is to develop methods, techniques, and algorithms to automatically generate narrative themes for a given video, as well as present the content in an easy-to-consume manner to end-users in a search engine experience.

How can machine learning techniques help us to solve this problem ?

## TRECVid Campaign

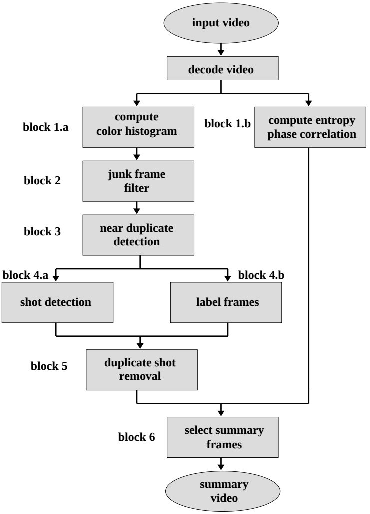
- the TREC conference series sponsored by NIST focus on information retrieval, providing a large test collection, uniform scoring procedures, and a forum for comparing results.
- from 2003, TRECVID, devoted to automatic segmentation, indexing, and Content-Based Retrieval of digital video, became an independent evaluation.

## TRECVid BBC Rushes Summarization Context

- The BBC Archive provided unedited material from about five UK dramatic series.
- In 2008 data were about :
  - 35 hours (57 clips) for rushes summarization development
  - 18 hours (40 clips) for rushes summarization test

to be summarized

# Our Summarization System

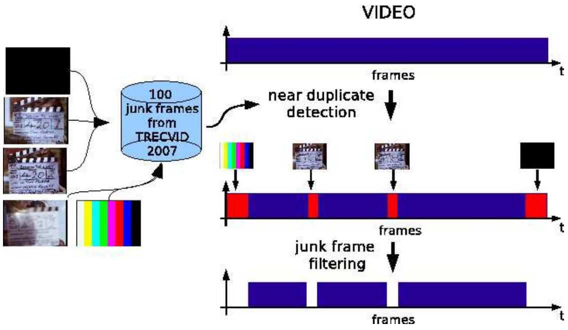


# Block 1: Descriptor

- HSV color histogram of 64 bins used for near-duplicate detection parts.
- entropy of phase correlation used as an action detector to select the most informative frames.

# Block 2: Junk frame filter

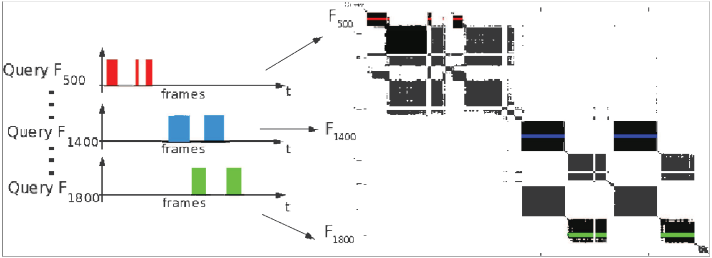
- Dataset of 100 samples of junk frame like rainbow color bars, clap-board, black frames extrated from TRECVid 2007.
- Elimination of junk frames using near duplicate detection.



# Block 3: Near-duplicate detection

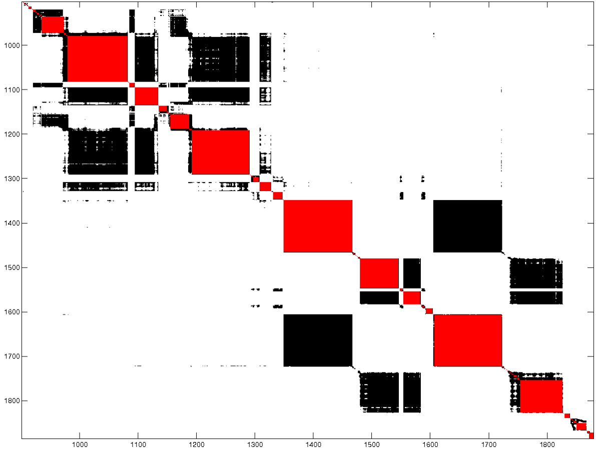
- Computation of the near-duplicate detection matrix M.
- As huge video, ”brute force” style search not feasible $\Rightarrow$ approximate near duplicate detection (Locality Sensitive Hashing [Datar et al. 2004]).

$$M_{ij} = 1 \qquad \textit{if and only if} \qquad F_j \in LSH(F_i)$$



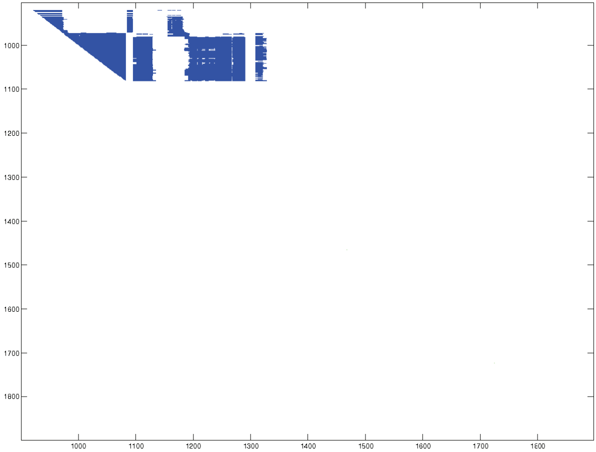
# Block 4.a: shot detection

Clustering of the adjacent near-duplicate frames along the diagonal of the near-duplicate matrix M.



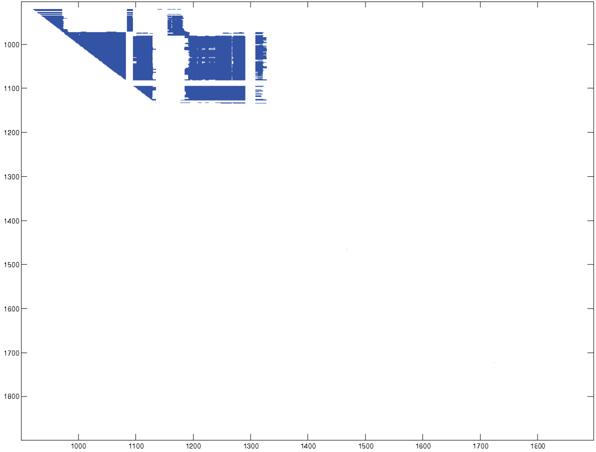
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



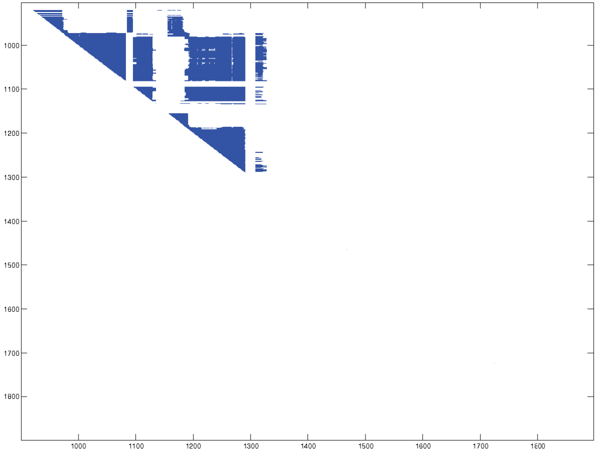
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



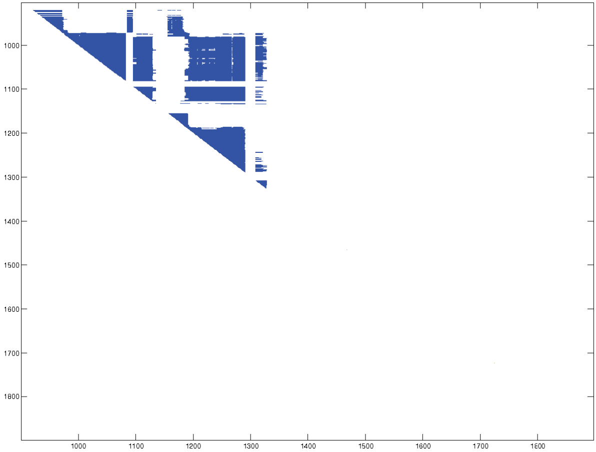
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



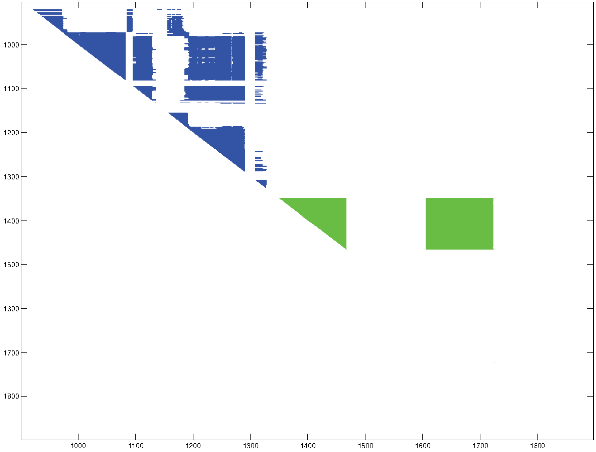
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



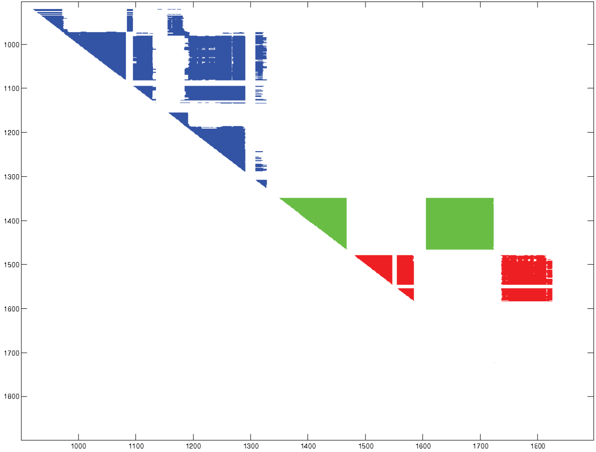
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



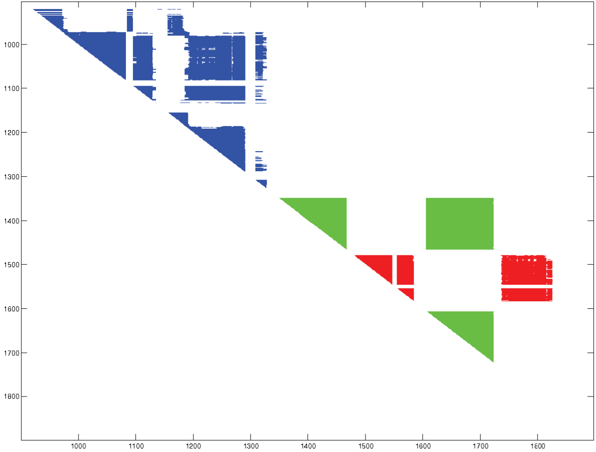
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



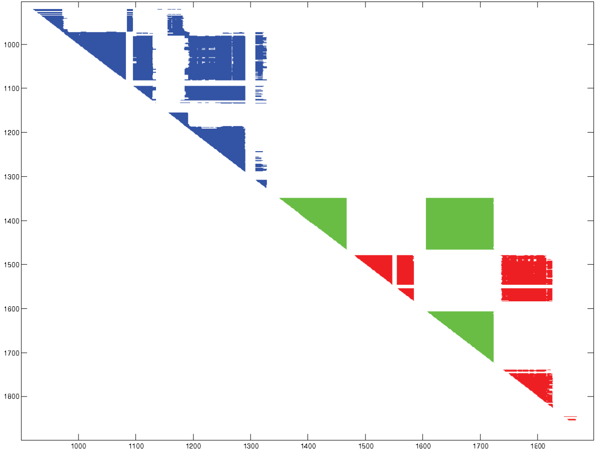
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



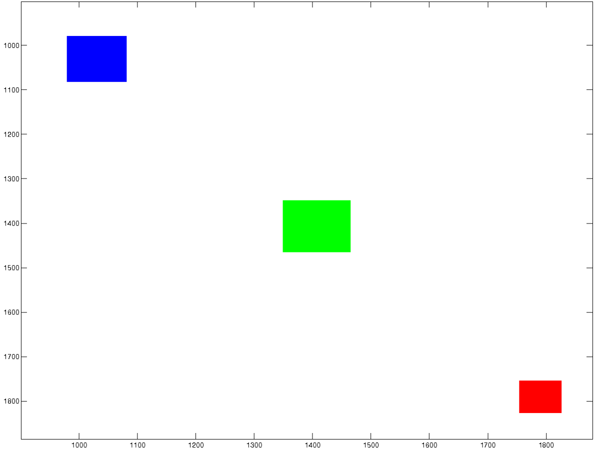
# Block 4.b: frame labelling

Clustering of all frames belonging to the same semantic information.



# Block 5: duplicate shot removal

- merging of the two previous results
- best block selection



# Block 6: summary frame rate (skimming)

- Automatic frame sampling = extremely short clips
- $\Rightarrow$ constraint for all shots to last, at least, 1 second (  lower duration for humans to recognize non-trivial visual content)
- As long as summary duration exceeds the target limit, shortest shot is removed
- Then, as long as the summary is not long enough, increase of the duration of shots that contain more motion
- Once the time is divided between shots this way, each shot is regularly sampled

## TRECVid BBC Rushes Summarization Task

To evaluate the TRECVid 2008 Rushes Task, the NIST considers 8 criteria :

- DU - duration of the summary (sec)
- XD - difference between target (2%) and actual summary size (sec)
- TT - total time spent judging the inclusions (sec)
- VT - total video play time (versus pause) judging the inclusions
- IN - fraction of inclusions found in the summary
- JU - Summary contained lots of junk
- RE - Summary contained lots of duplicate video
- TE - Summary had a pleasant tempo/rythm.

result of a summary

## TRECVid 2008 Rushes Task evaluation

- Good trade-off between non-detection and false detection:
  - Results for fraction of inclusions found in the summary (18th)
  - Summary containing lots of duplicate video (17th)
- Not visually good results:
  - Regular frame rate (after shot duration assignment) is not a so good choice $\Rightarrow$ summary are jerky $\Rightarrow$ pleasant tempo/rhythm (37th over 43 runs)
  - total time spent to judge inclusions (38th) and total video play time (versus pause) to judge inclusions (34th)

## Some conclusions on Summarization Task

- Most groups, majority, used some form of clustering of shots/scenes in order to detect redundancy;
- Most groups used visual-only, though some also used audio in selecting segments to include in summary;
- Finally, most groups used whole frame for selecting, though some also used frame regions;

**However BBC (and many others) is also interested in:**

Automatic summarization of produced video for mobile devices (mobisodes)

- ***catch-up:*** find the video in episode x needed to understand episode x+1
- ***preview:*** find the video in an episode that will make a viewer want to see the episode but without destroying suspense

### Our System: Current stage

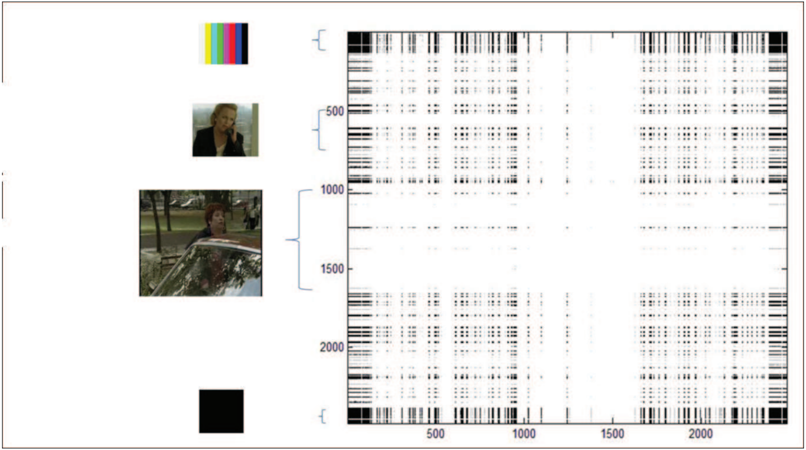
- Mainly one simple global descriptor: HSV color histogram
- approximate clustering based on LSH to reach good computational performance
- Good results for redundancy removal and relevant video fragment extraction
- Skimming process to adapt frame sampling rate in each relevant video fragment based on average motion estimation (phase correlation descriptor).

### Future work

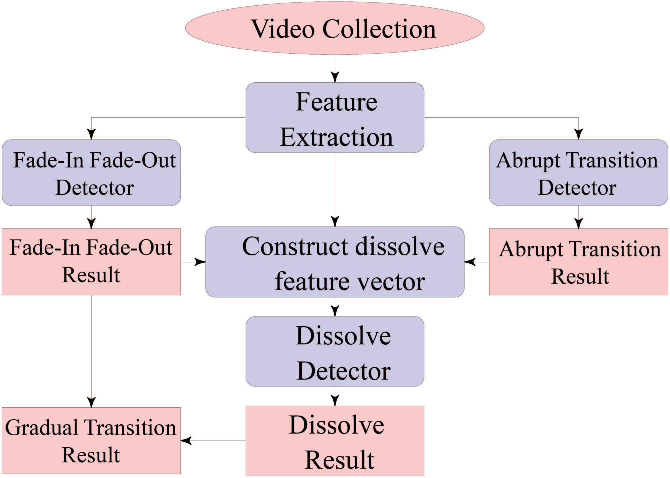
- jerky visual effects
- computational power of LSH approach allows larger or more complex features $\Rightarrow$ increase precision of semantic fragment extraction + ease skimming process + improve visual quality
- **extend to classic videos**

## From Rushes to classic video : Motion

- Motion descriptor = Optical flow estimated with KLT
- LSH provides a matrix of motion detection in the video (Black pixels stand for static scenes and white pixels for scene with detected motion)



# Our system


Video Collection
Feature Extraction
Fade-In Fade-Out Detector
Abrupt Transition Detector
Fade-In Fade-Out Result
Construct dissolve feature vector
Abrupt Transition Result
Dissolve Detector
Gradual Transition Result
Dissolve Result

# Machine Learning approach

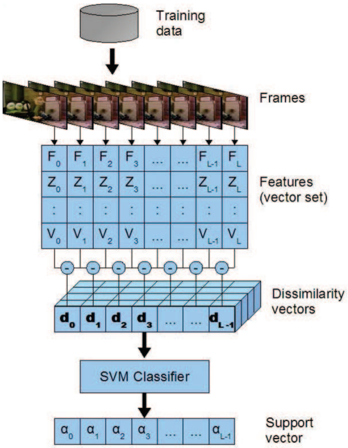



**Figure:** Train

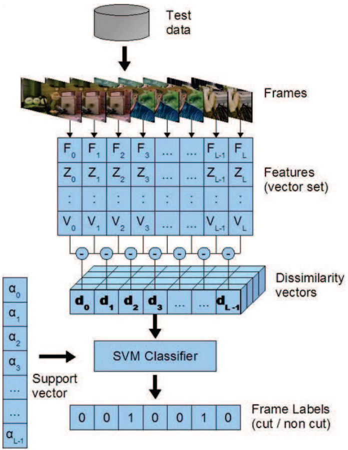



**Figure:** Test

# Abrupt transition features

- Color Histograms (RGB, HSV, Opponent color)
- Color moments (mean, variance)
- Phase correlation method
- Projection histograms
- Shape descriptors: Zernike and Fourier-Mellin moments.

# Result TRECVid 2006

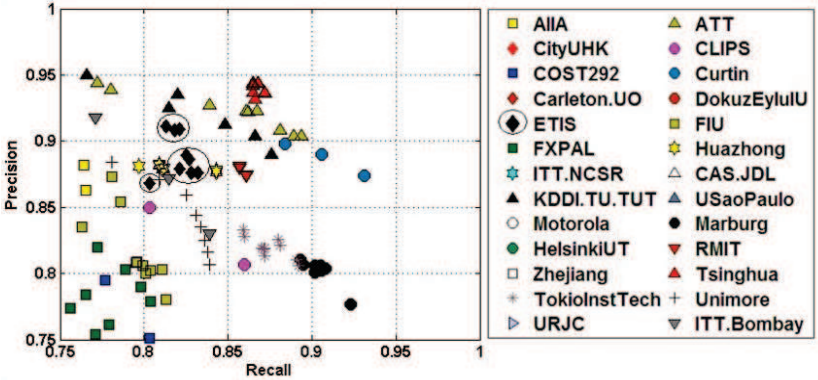
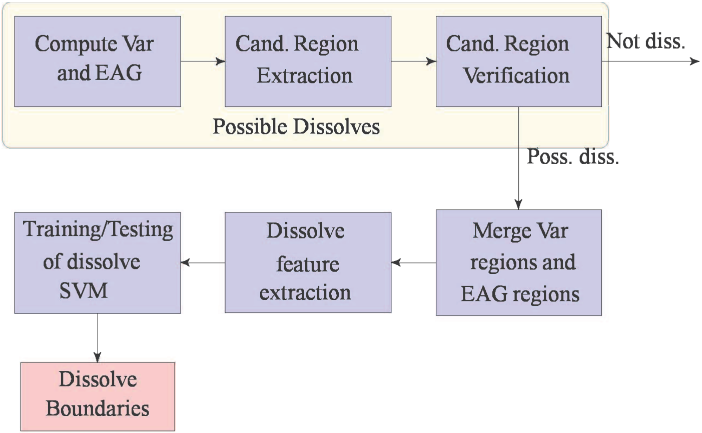



**Figure:** Precision/Recall abrupt transitions

# Gradual Transition Schema



# Dissolve detection

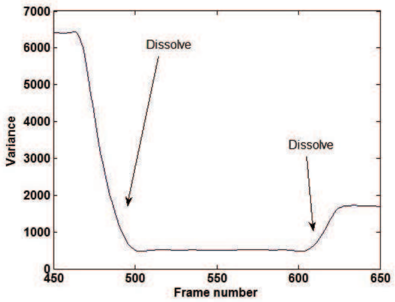



Figure: Luminance Variance

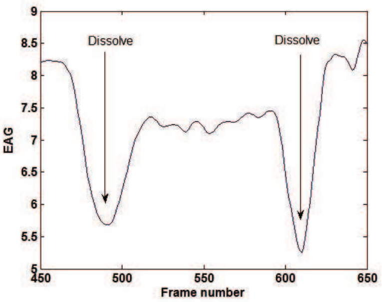



Figure: Edge Average Gradient

# Dissolve features

- Difference vectors: 2 histogram differences; Correlation coefficients [Han, 2003].
- Double chromatic difference (DCD) [Yu, 1997]
- Projection histograms
- Motion

# Result TRECVid 2006

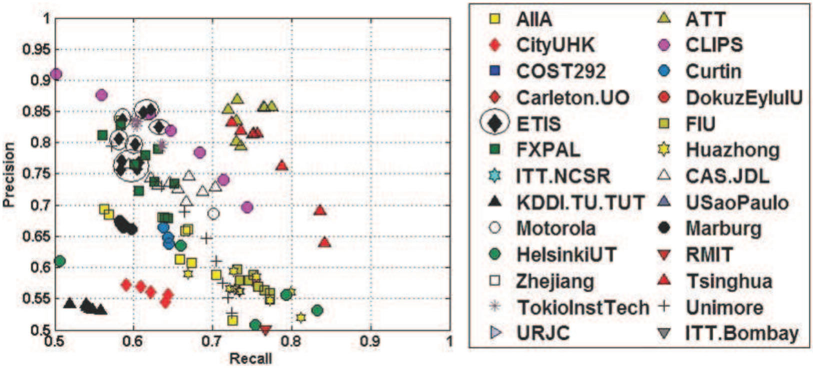



**Figure:** Precision/Recall gradual transitions

# Result TRECVid 2006

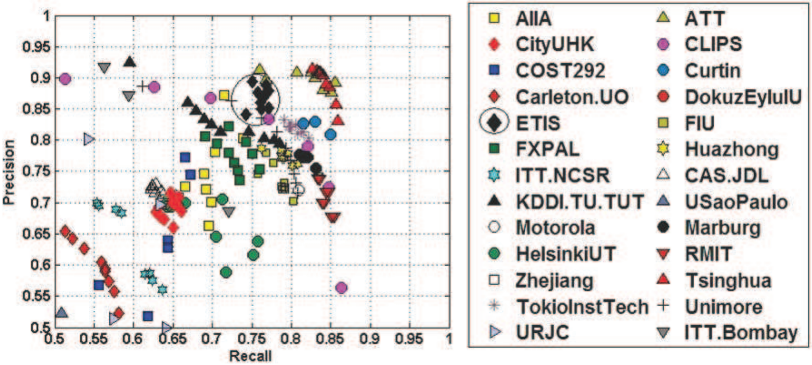



**Figure:** Precision/Recall for all transitions

# Some Conclusions on Shot Boundary Detection Task

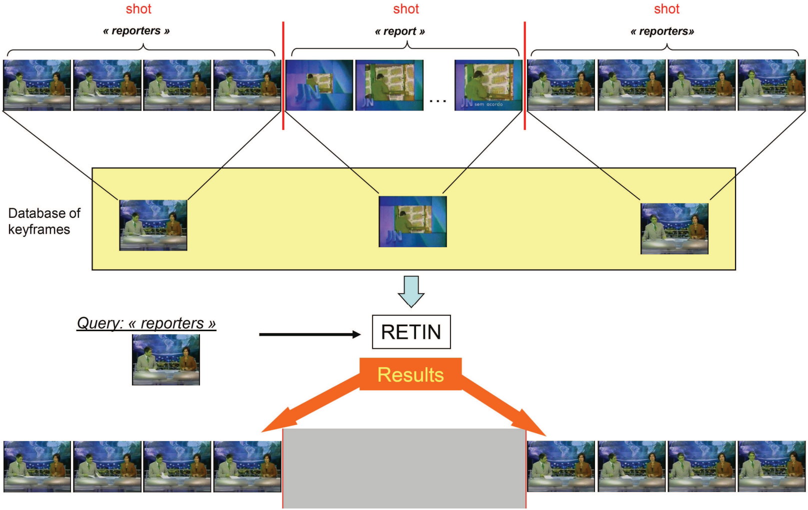
| Rank | Group | MLrn | ColHst [2] | Flash | LVals | Cmpr | AThr | MCmp | Edgs | STmp | Other |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Tsinghua University | ✓ | $\checkmark^{48}$ | ✓ | - | - | - | ✓ | - | - | - |
| 2 | National ICT Australia (NICTA) | ✓ | - | - | - | - | - | - | - | - | - |
| 3 | IBM Research | ✓ | $\checkmark^{512}$ | ✓ | ✓ | - | ✓ | - | ✓ | - | - |
| 4 | CLIPS-IMAG | - | - | ✓ | - | - | ✓ | ✓ | - | - | - |
| 5 | KDDI R&D Labs Inc. | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | - | - |
| 6 | University of Marburg | ✓ | $\checkmark^{512}$ | ✓ | ✓ | ✓ | - | ✓ | ✓ | - | - |
| 7 | RMIT University | ✓ | $\checkmark^{16}$ | - | - | - | ✓ | - | - | - | - |
| 8 | U. Central Florida & U. Modena | ✓ | ✓ | - | ✓ | ✓ | - | - | - | - | - |
| 9 | FX Palo Alto Laboratory (FXPal) | ✓ | ✓ | - | - | - | - | - | - | - | - |
| 10 | City Univ. Hong Kong | ✓ | $\checkmark^{512}$ | ✓ | ✓ | ✓ | - | - | ✓ | ✓ | - |
| 11 | Technical Univ. Delft | - | - | ✓ | ✓ | - | - | - | - | ✓ | - |
| 12 | Imperial College London | - | ✓ | ✓ | - | - | - | - | - | - | - |
| 13 | Hong Kong Polytechnic Univ. | - | ✓ | ✓ | - | - | - | - | - | - | ✓ |
| 14 | Fudan University | ✓ | ✓ | - | ✓ | ✓ | ✓ | - | - | - | - |
| 15 | University of Sao Paulo | - | ✓ | - | - | - | - | - | - | - | - |
| 16 | LaBRI | - | ✓ | - | ✓ | ✓ | - | ✓ | - | - | |
| 17 | Motorola Multimedia Res. Lab. | - | - | - | - | - | - | - | - | - | ✓ |
| 18 | Indian Institute of Technology (IIT) | - | - | ✓ | - | - | - | - | - | - | ✓ |
| 19 | University of Iowa | - | $\checkmark^{16}$ | - | - | - | - | - | ✓ | - | - |
| 20 | University Rey Juan Carlos | - | $\checkmark^{16}$ | ✓ | - | - | ✓ | - | - | - | ✓ |
| 21 | Florida International Univ. | - | - | - | - | - | - | - | - | - | ✓ |

# Content-based video shot retrieval

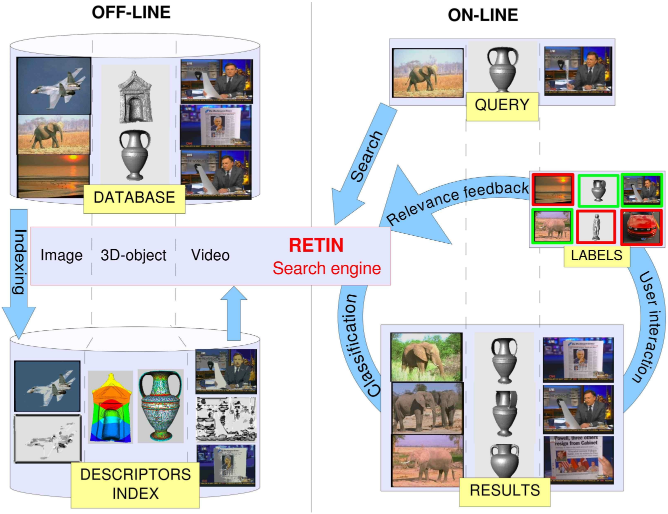
Now that we have a video shot segmentation algorithm, ***how can we exploit the result to build a content-based video shot retrieval system ?***

Extraction of 1 Keyframe per shot $\Rightarrow$ Shot retrieval = Image retrieval

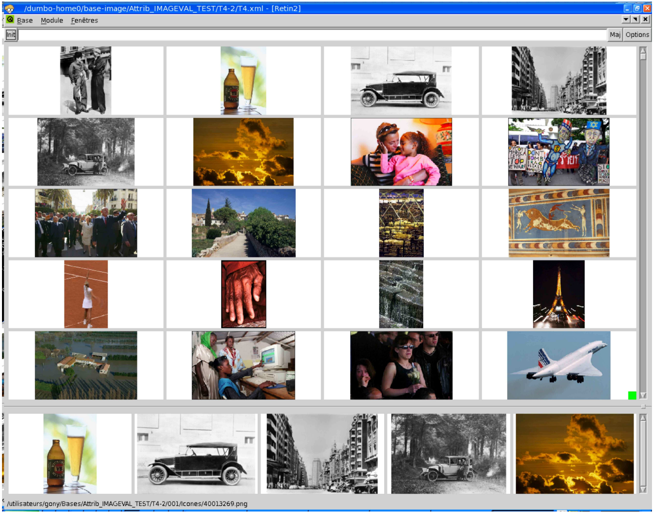
# Content-based video shot retrieval



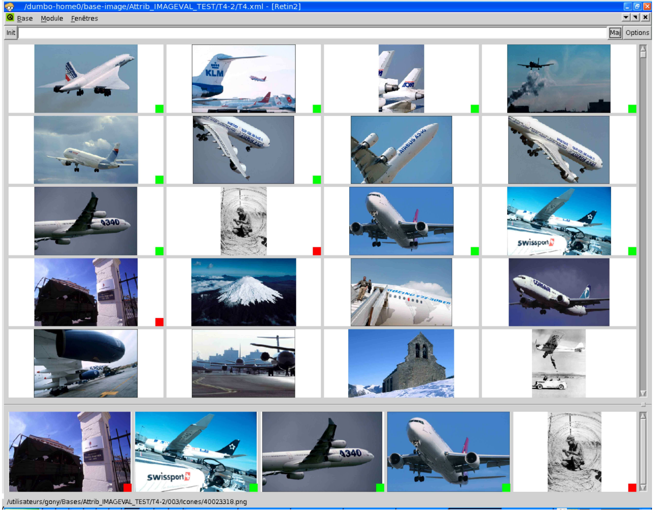
# CBR system:


OFF-LINE
ON-LINE
DATABASE
QUERY
Search
Relevance feedback
LABELS
Indexing
Image
3D-object
Video
RETIN
Search engine
Classification
User interaction
DESCRIPTORS
INDEX
RESULTS

# Example of search by similarity



# Example of search by similarity



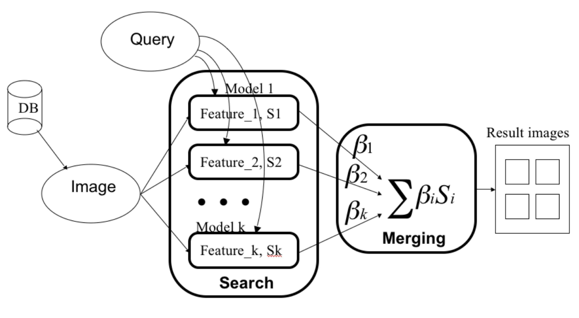
# Online learning

## Optimisation of the ranking using $\mathcal{A}$

# Online learning

## Optimisation of the ranking using $\mathcal{A}$

- similarity function updating $f(\mathbf{x})$



# Online learning

## Optimisation of the ranking using $\mathcal{A}$

- similarity function updating $f(\mathbf{x})$
- classification scheme

# Online learning

## Optimisation of the ranking using $\mathcal{A}$

- similarity function updating $f(\mathbf{x})$
- classification scheme

## $\mathcal{A}$ Enhancement

- Show the top rank examples

# Online learning

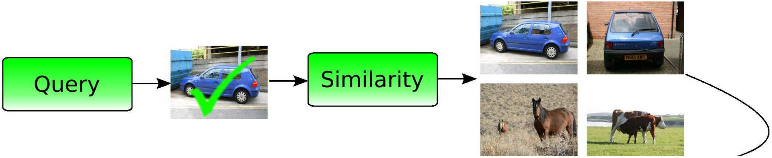
## Optimisation of the ranking using $\mathcal{A}$

- similarity function updating $f(\mathbf{x})$
- classification scheme

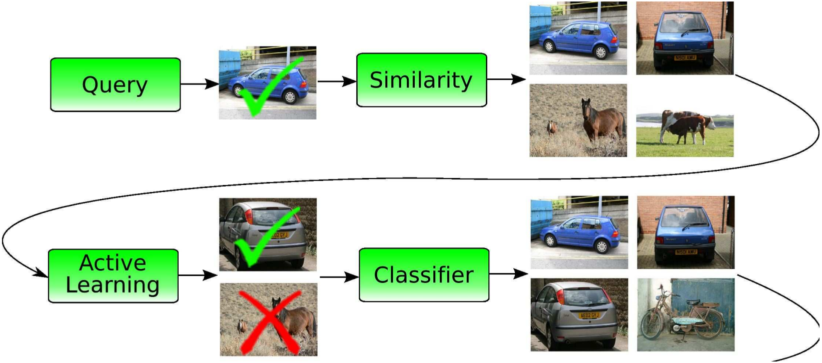
## $\mathcal{A}$ Enhancement

- Show the top rank examples
- or show the best ones to enhance ranking: Active learning strategy

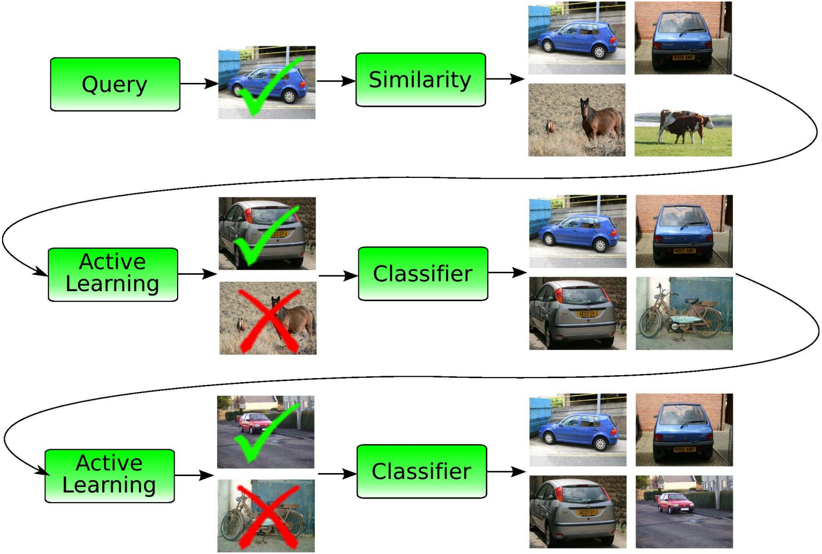
# Online learning



# Online learning



# Online learning



# Content-based Video Retrieval: Query

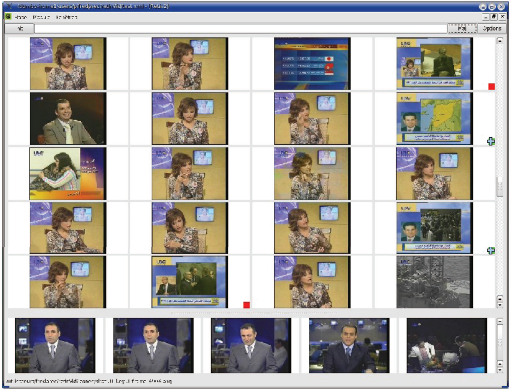


**Figure:** Query

# Content-based Video Retrieval: Result

**Figure:** Top ranked

# Content-based Video Retrieval: Result

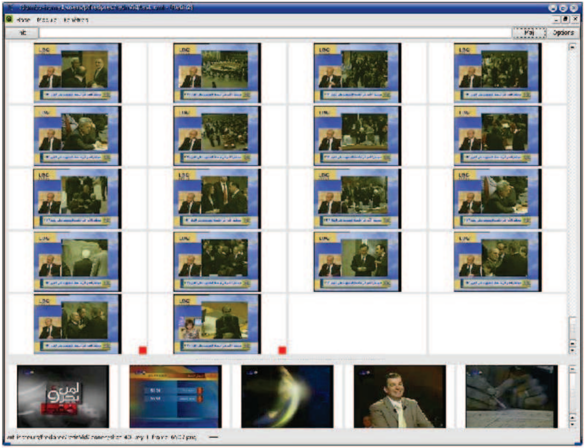


**Figure:** Bottom ranked

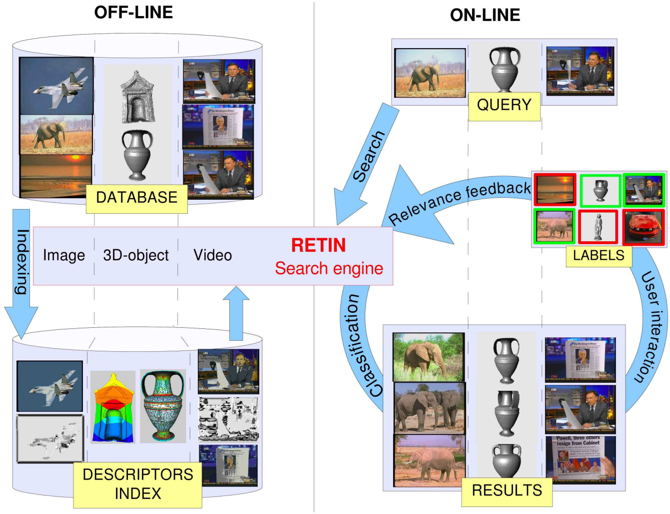
# More complex content-based video shot retrieval ?

Up to now: extraction of 1 Keyframe per shot $\Rightarrow$ Shot retrieval = Image retrieval

***Can we build a more powerful, a more sophisticated content-based video shot retrieval system ?***

1 Shot = more than 1 vector !

# Introduction to Kernel design



# Introduction to Kernel design

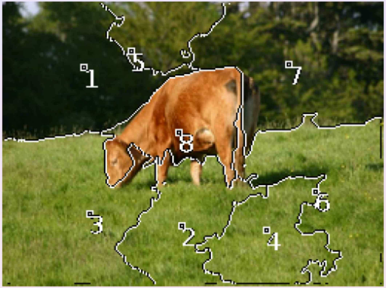
**Key components of the RETIN system**

- Visual data representation
- Similarity as Kernel
- Kernel-based SVM classification, active learning

# Visual data representation

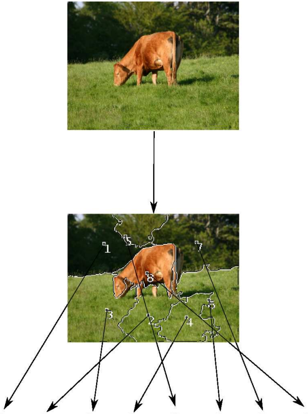
## Image features

- Pixels, Points of Interest, RoI, Regions, Blobs




## Back to indexing process

1. Feature extraction
   $\Rightarrow$ Regions, points of interest, ...
2. Descriptors
   $\Rightarrow$ Colors, textures, SIFT, ...
3. Bags of feature $B_i = \{\mathbf{b}_{ri}\}_r$
   $\Rightarrow$ $\mathbf{b}_{ri}$ : region/poi $r$ of image $i$
4. Similarity $S(B_i, B_j)$



$$\underbrace{\left\{\begin{array}{cccccccc} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \mathbf{b}_4 & \mathbf{b}_5 & \mathbf{b}_6 & \mathbf{b}_7 & \mathbf{b}_8 \\ 37 & 37 & 78 & 90 & 19 & 38 & 23 & 77 \\ 83 & 09 & 83 & 12 & 78 & 29 & 36 & 82 \\ 83 & 61 & 38 & 72 & 88 & 99 & 01 & 32 \\ 92 & 43 & 82 & 72 & 66 & 72 & 28 & 39 \end{array}\right\}}$$

Bag $B = \{\mathbf{b}_r\} \in \mathcal{B}$

## Similarity $S(B_i, B_j)$ using Visual Dictionary

- 2 steps:
  1. Explicit mapping of $B_i$ into a vector space
  2. Similarity on vectors
- Computation of the Visual Dictionary over the database
- Strategies to cluster all the feature data, like k-means

## Image index: distribution on the Visual dictionary

## One step further to track a noodle in a haystack

- Dictionary-based approaches $=>$ Vectors as index
- Other index ? more discriminant ?

## Similarity functions $S(B_i, B_j)$

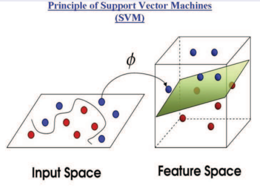
- Alternatives to dictionary-based approaches:
  1. Copy detection approach :
     - Signature $= B_i$ the set of vectors $b_{ik}$
     - Similarity retrieval using NN search and voting systems
  2. Kernels on bags

# Kernel as similarities

## Definition of a kernel function $K : \mathcal{X} x \mathcal{X} \rightarrow \mathcal{R}$

$K$ is a kernel *iff* $\exists \Phi | \forall (x, y), K(x, y) = < \Phi(x), \Phi(y) >$
with $\Phi$ an injection into a Hilbert $\mathcal{H}$ space (explicit or not)

$$\begin{array}{rcl} \Phi : \mathcal{X} & \rightarrow & \mathcal{H} \\ x & \rightarrow & \Phi(x) \end{array}$$




## Advantages:

- Integration with Machine Learning techniques (Neural networks, SVM, . . .)
- Allow to build similarities on non vector input spaces

# Kernel functions for bags of vectors

**Framework:**

Soft maximum kernel function [Shawe-Taylor book02]:

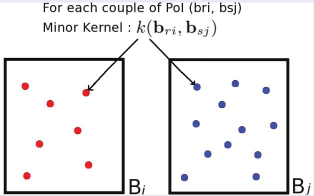
$$K_{softmax}(B_i, B_j) = \sum_{\mathbf{b}_{ri} \in B_i} \sum_{\mathbf{b}_{sj} \in B_j} k(\mathbf{b}_{ri}, \mathbf{b}_{sj})$$

- Nice property:
  $k$ is a kernel function $\Rightarrow K_{softmax}$ is a kernel function
- Not enough discriminant ?

# Kernel on Bags of Features

## Improvement [LyuCVPR 05, CordCIVR 07]

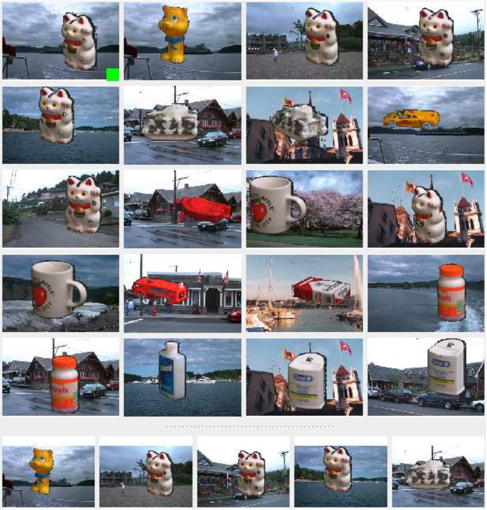
$$K(B_i, B_j) \triangleq \left( \sum_r \sum_s \left( k(\mathbf{b}_{ri}, \mathbf{b}_{sj}) \right)^q \right)^{\frac{1}{q}}$$



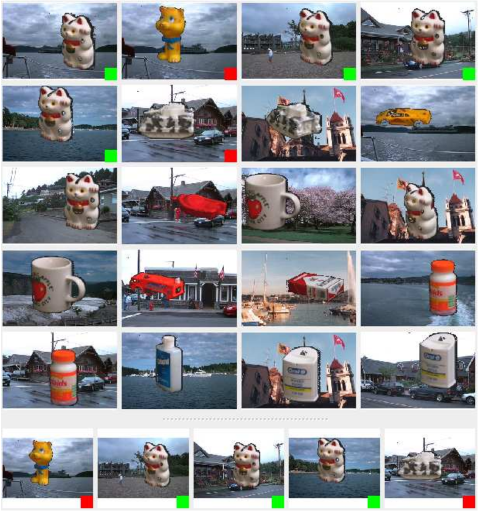

Good Results

BUT Complexity of kernel evaluation high $=>$ fast scheme

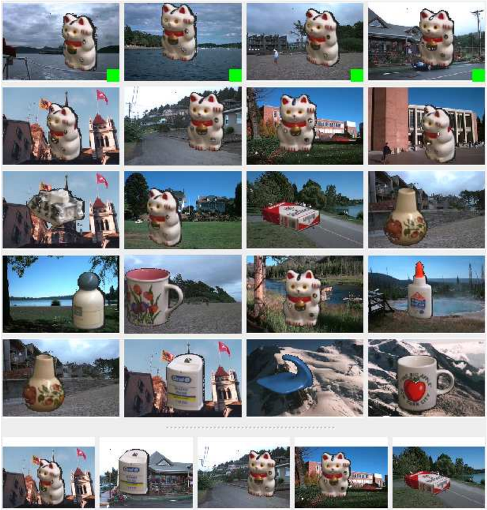
# Example of retrieval session



# Example of retrieval session

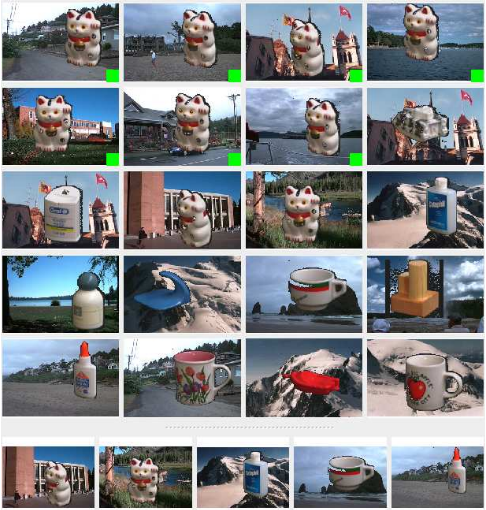


# Example of retrieval session

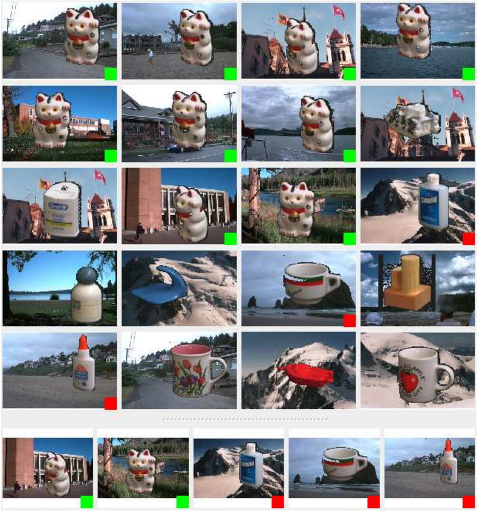


# Example of retrieval session

# Example of retrieval session



# Example of retrieval session



# Example of retrieval session

## Any Extension ?

## Extension : integration of spatial constraints

## Extension : integration of spatial constraints

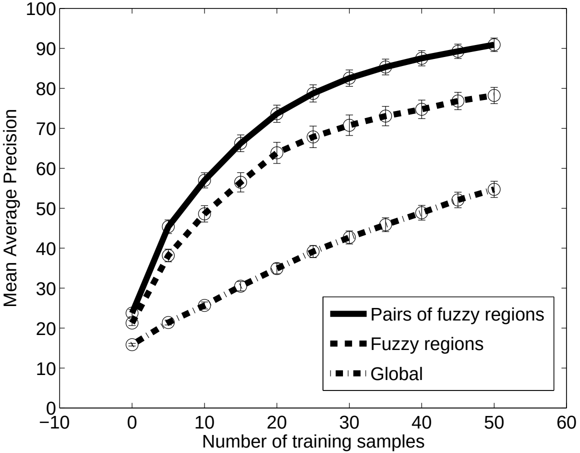
## Extension : integration of spatial constraints


Kernel function on bag $\mathcal{P}_i$ of bags of pairs $P_{vi}$:

$$K_{pairs}(\mathcal{P}_i, \mathcal{P}_j) = \left( \sum_{P_{vi} \in \mathcal{P}_i} \sum_{P_{wj} \in \mathcal{P}_j} K_{single}(P_{vi}, P_{wj})^q \right)^{\frac{1}{q}}$$

For each region $\mathbf{b}_{ri}$, we build 3 pairs with its 3 closest regions.
$K_{pairs}$ may be connected to kernel on graphs [Kashima]

# Evaluation




RETIN Active learning with 5 labels/feedback, 10 feedbacks.

# Extension (2): application to video actor retrieval

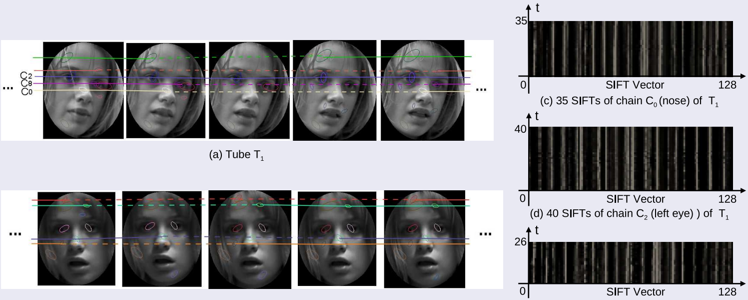
## Video object extraction and description

- RoI = face tubes
  - Frame face detection
  - Face region grouping in shots

## Example of a tube:

## Data representation

Temporal stability of SIFT points: Intra-tube chain tracking




(a) Tube $T_1$

(b) Tube $T_2$

(c) 35 SIFTs of chain $C_0$ (nose) of $T_1$

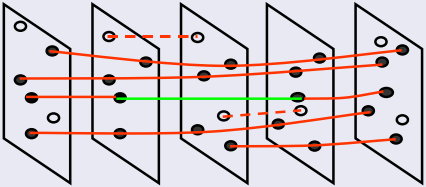
(d) 40 SIFTs of chain $C_2$ (left eye) ) of $T_1$

(e) 26 SIFTs of chain $C_8$ (right eye) ) of $T_1$

SIFT points along the same chain in same color (scale and orientation of ellipses representing the scale and orientation of SIFT)

## Representation optimization

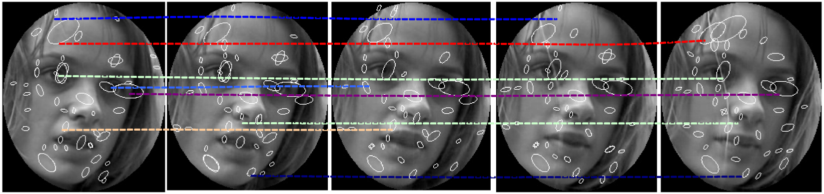
- Intra-tube chain tracking
- Consistent chain extraction:



Solid lines: consistent chains, dash lines: noise, green lines: linking two short chains

- Tube $T_i$: a set of chains $C_{ri}$ of SIFT descriptors:
- $T_i = \{C_{1i}, \ldots, C_{ki}\}$ and $C_{ri} = \{SIFT_{1ri}, \ldots, SIFT_{pri}\}$

# Kernel design for actor retrieval



The major kernel on tubes is then defined as:

$$K'_{pow}(T_i, T_j) = \left( \sum_r \sum_s \frac{|C_{ri}|}{\sqrt{|T_i|}} \frac{|C_{sj}|}{\sqrt{|T_j|}} k'(C_{ri}, C_{sj})^q \right)^{\frac{1}{q}} \quad (1)$$

with the following minor kernel on chains:

$$k'(C_{ri}, C_{sj}) = exp\left( -\frac{1}{2\sigma^2} \chi^2 \left( \overline{C}_{ri}, \overline{C}_{sj} \right) \right) e^{-\frac{\left(\overline{x}_{ri} - \overline{x}_{sj}\right)^2 + \left(\overline{y}_{ri} - \overline{y}_{sj}\right)^2}{2\sigma_2^2}}$$

**Kernel design**

And so what ?

**Kernel design**

And so what ? Actually, all the work is done !

## Kernel design

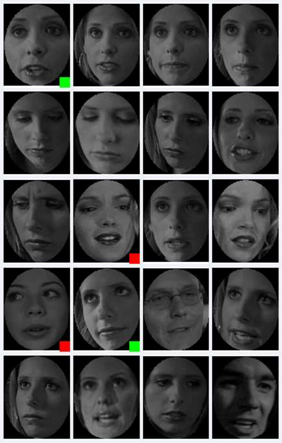
And so what ? Actually, all the work is done !
It is now RETIN compatible: online actor retrieval

# Experiments on a french movie ”L’esquive”

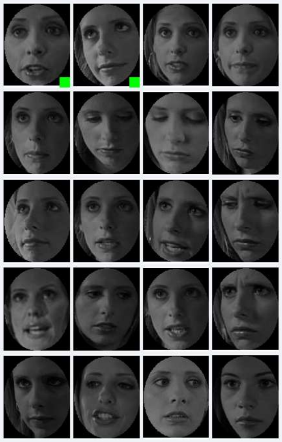
## Experiments on a french movie ”L’esquive”

## Experiments on a french movie ”L’esquive”

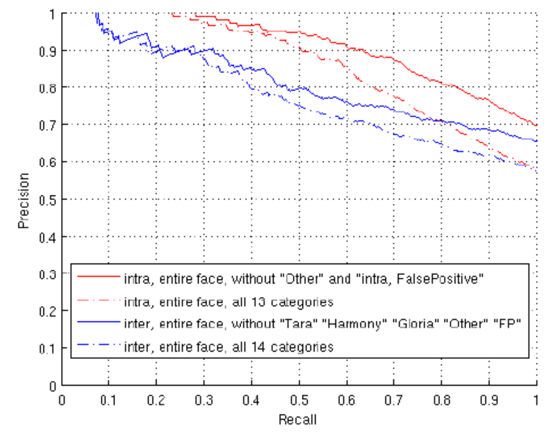
## Experiments for multi-class actor retrieval on videos ”Buffy” [Zisserman&Sivic database]



## Experiments for multi-class actor retrieval on videos ”Buffy” [Zisserman&Sivic database]



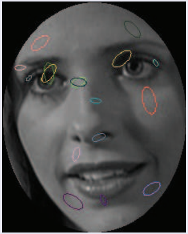
## Experiments for multi-class actor retrieval on videos ”Buffy” [Zisserman&Sivic database]

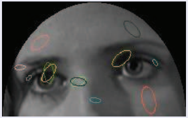


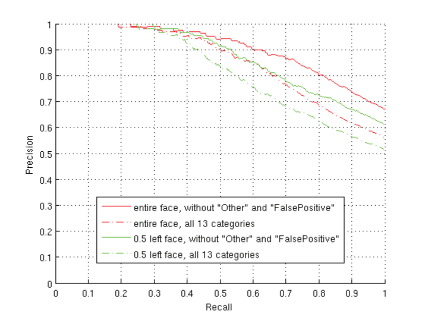
result on Buffy (intra-episode)

result on Buffy (inter-episode)

## Experiments on system robustness:







## Experiments on system generalization:

## Experiments on system generalization:

## Experiments on system generalization:

## Experiments on system generalization:

# Introduction to fast retrieval scheme

## Computation optimization pb

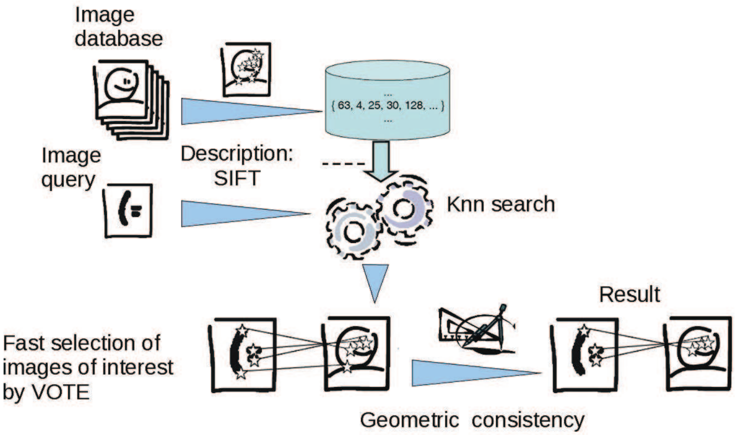
Control of search complexity when the size of the database becomes huge
Problem even more crucial when the number and the size of the descriptors increase

## Computational pb of similarity functions $S(B_i, B_j)$

- All the Alternatives to dictionary-based approaches are time consuming
  1. Copy detection approach :
     - Signature $= B_i$ the set of vectors $b_{ik}$
     - Similarity retrieval using NN search and voting systems
  2. Kernels on bags

$\Rightarrow$ Need optimization scheme !

# Copy Detection scheme [Lowe04]



# Copy Detection scheme

## Optimization scheme

- **Fast NN search (1)** to quickly retrieve near duplicate or most similar images (TOPN) to a given query
- Need to structure the database $\Rightarrow$ indexing scheme

## Database indexing schemes

- Classical indexing schemes fail with high dimensional data
- Approximate search approaches
  - Tree techniques (Kd-tree,. . .)
  - Projections (NV Tree, VA files, Space Filing Curves, **Locality Sensitive Hashing** )

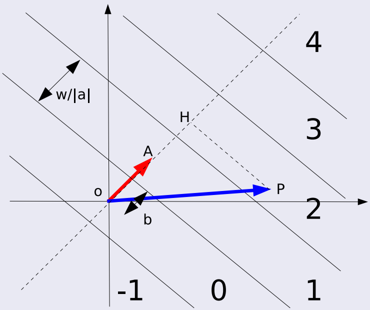
# Implementing Locality Sensitive Hashing

## [datar 2004]

$f_i()$: function of the hash table $i$ and $h_{a,c}()$ the hash function:

$f_i(\mathbf{b}) = \left(h^i_{a_1,c_1}(\mathbf{b}), \ldots, h^i_{a_k,c_k}(\mathbf{b})\right)$

$h_{\mathbf{a},c}(\mathbf{b}) = \left\lfloor \frac{\mathbf{a}.\mathbf{b}+c}{w} \right\rfloor$



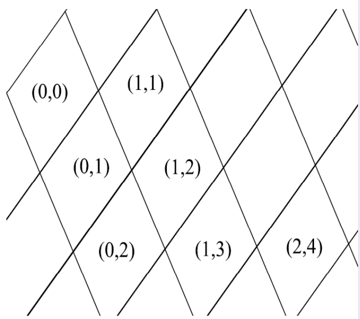
# Implementing Locality Sensitive Hashing

## [datar 2004]

$f_i()$: function of the hash table $i$ and $h_{a,c}()$ the hash function:

$f_i(\mathbf{b}) = \left(h^i_{a_1,c_1}(\mathbf{b}), \ldots, h^i_{a_k,c_k}(\mathbf{b})\right)$

$h_{\mathbf{a},c}(\mathbf{b}) = \left\lfloor \frac{\mathbf{a}.\mathbf{b}+c}{w} \right\rfloor$



# Implementing Locality Sensitive Hashing (2)

## Implementation depending on the representation space

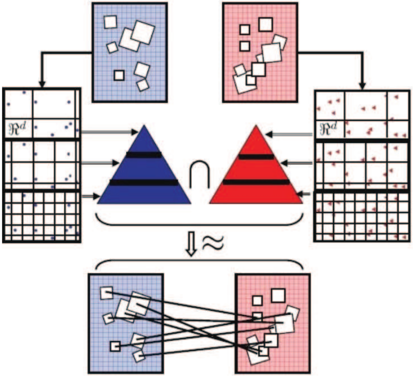
- in Hamming space $\mathcal{H}^d$ or $\mathcal{Z}^d$: LSH random permutation [Indyk98]
- in $\mathcal{R}^d$ normalized: cosine similarity [Charikar02]
- in $\mathcal{R}^d$: distance L2 or L1
  - [Gionis99] projection of $\mathcal{R}^d$ in $\mathcal{H}^d$ + [Indyk98]
  - [Datar04] splitting along 1 dimension
  - [Lv07] (multi-probe) extension of [Datar04]
  - [Andoni06] 24 lattice, [Jegou08] E8 lattice
- Implementation available for a vector representation of images and distances aforementioned
- Extension to other similarities and to non vector spaces ?

# Fast kernel on Bags Pyramid Match Hashing [Grauman07]

- Each image is described by a bag of SIFT
- Injection with a function Φ in a space of high dimension
- The injection is explicit:
  - Projection into SIFT space
  - Multi-scale grid
  - Projection into Hamming space

⇒ Each image becomes a unique Vector

- An explicit induced space allows to use LSH
- The resulting kernel allows to get a similarity from the matching between PoIs (Points of Interest) of the 2 images



# LSH on other kernels ?

- Pyramid Match Hashing $\Rightarrow$ good performances
- BUT cannot be extended to kernels where $\Phi$ is not explicit
- If the class of kernels is different:

$$\text{ex: } K(B_i, B_j) = \left( \sum_r \sum_s \left( k(\mathbf{b}_{ri}, \mathbf{b}_{sj}) \right)^q \right)^{\frac{1}{q}}$$

- Can we speed up the computation?

# Our scheme [ICPR 2008]

- Model:
  - consider each image as a bag of unordered features
  - similarity : class of kernels on bags

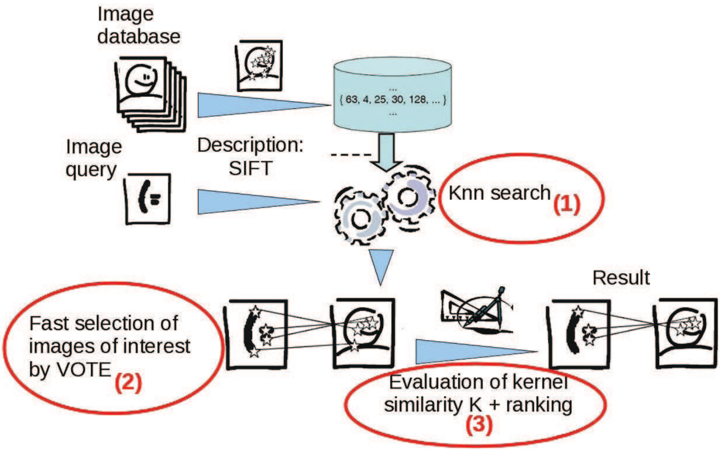
$$K(B_i, B_j) = \left( \sum_r \sum_s \left( k(\mathbf{b}_{ri}, \mathbf{b}_{sj}) \right)^q \right)^{\frac{1}{q}}$$

- Objective:
  - fast computation of the topN from a ranking of the database with similarity kernel K

  $\Rightarrow$ decrease the kernel computational complexity
- Principle (inspired from copy detection):
  - (1) (2) Quick selection of database subset (LSH scheme)
  - (3) Kernel computation only on this relevant subset

  $\Rightarrow$ resulting scheme is an approximation of the exact similarity ranking of the whole database

# Principle for fast retrieval



# Pre-processing: Hashing of the database

For each image $B_i$

For each attribute $b_{si}$

For each hash table $k$

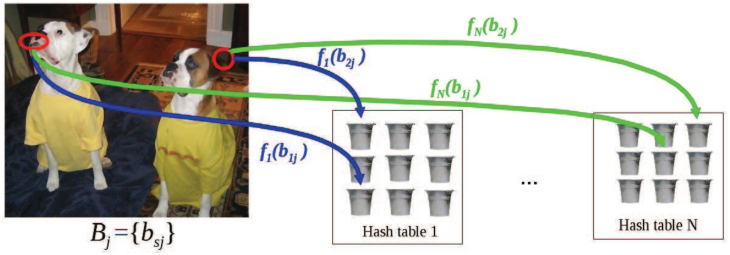
- selection of a bucket with hashing function: $f_k(b_{si})$
- put $b_{si}$ in the selected bucket

Locality Sensitive Hashing [datar 2004]
Notation : $f_i()$: function of the hash table $i$

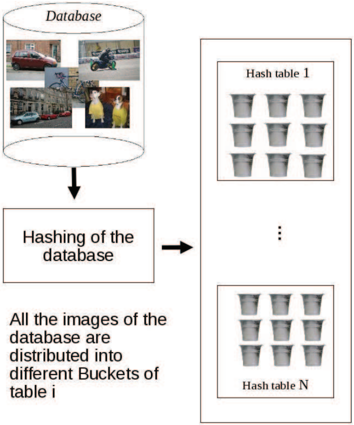
$$f_i(\mathbf{b}) = \left(h^i_{a_1,c_1}(\mathbf{b}), \ldots, h^i_{a_k,c_k}(\mathbf{b})\right)$$

$h_{a,c}()$ : hash function

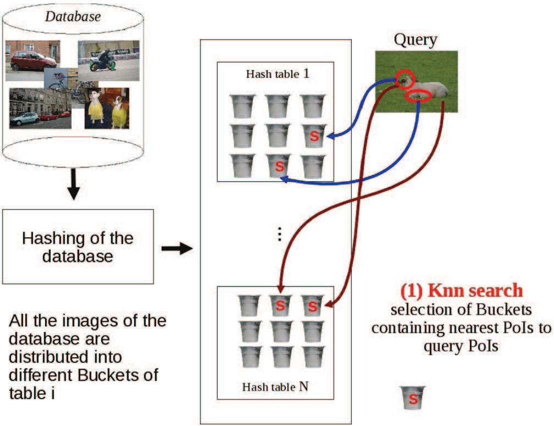
$$h_{\mathbf{a},c}(\mathbf{b}) = \left\lfloor \frac{\mathbf{a}.\mathbf{b}+c}{w} \right\rfloor$$



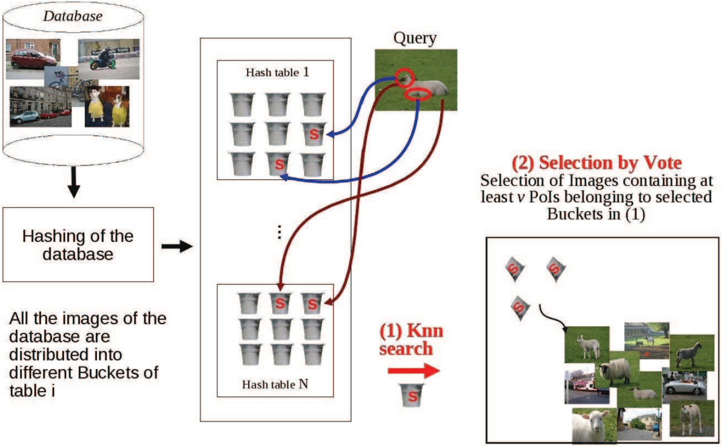
# Retrieval Algorithm



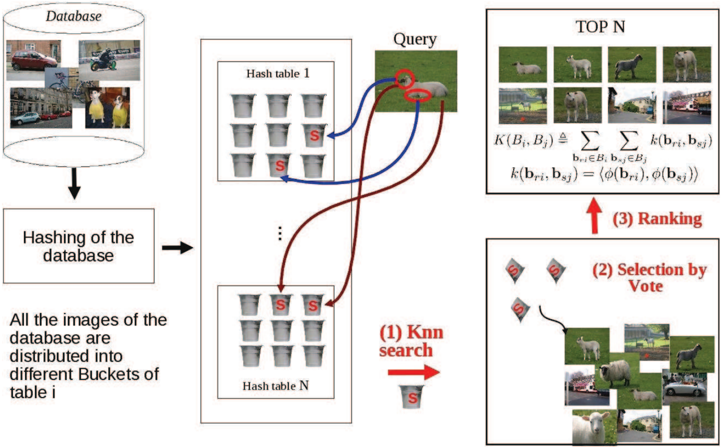
# Retrieval Algorithm



# Retrieval Algorithm



# Retrieval Algorithm

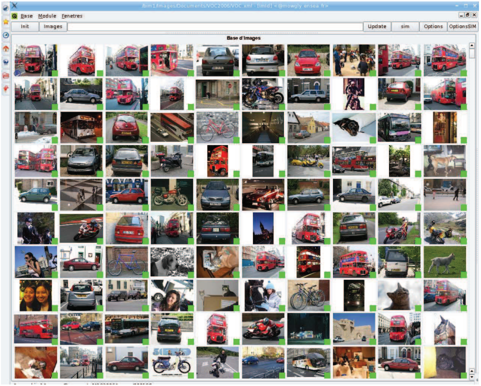


# Experiments

- VOC2006 database : 5,304 images
- Indexing : $\sim$100 PoI per Image
  - MSER region detectors
  - SIFT descriptors
- Variance normalization
- E2LSH parameters
  - radii between 150 and 250 (4.0 and 6.0 after normalization)
  - L = 50 hash tables
  - K = 20 projections
- Image selection VS whole database
  - TOP100 deterioration
  - computational time reduction

# Example

Fast Selection + Ranking by Vote

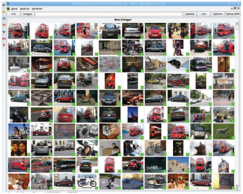



372 / 5304 images (7,1% of the database)

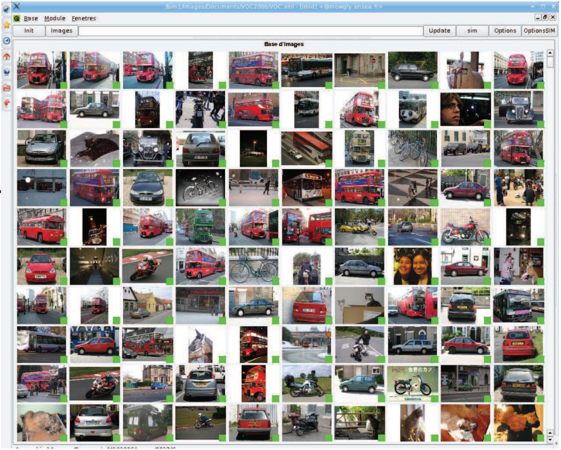
# Example

Ranking of the selection by Similarity K

Fast selection



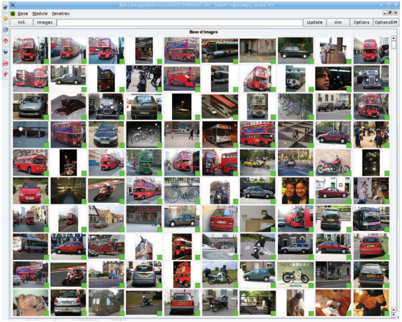
⇒



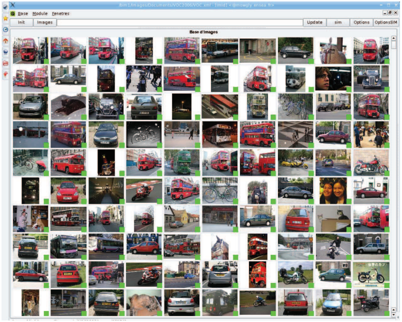

372 / 5304 images (7,1% of the database)

# Example

Selection ranking

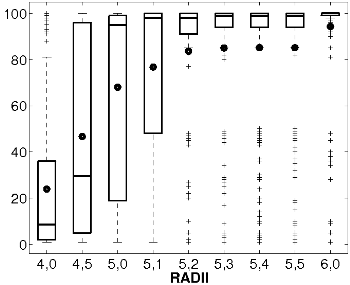



Ground truth for K : Ranking of the whole database

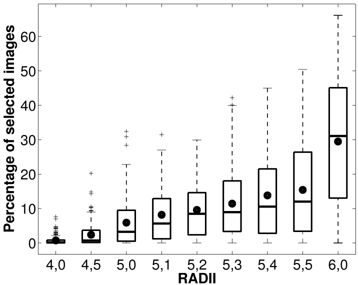



96% of images of TOP100 obtained from our fast selection are identical to TOP100 on the whole database

# Results




Accuracy of TOP100 for various radii of search around query points




Percentage of selected images for various radii

| radius | 4.0 | 5.0 | 5.2 | 6.0 |
|---|---|---|---|---|
| factor | 122.17 | 14.85 | 10.03 | 3.19 |

Speed improvement factor regarding the true search

# Discussion

## Fast similarity scheme

- Fast similarity search working with non explicit kernels and with all fast knn search methods
- Good trade-off between Precision and Speed for R=5.2: 10 time faster and median precision 99%

## But ...

TOPN not good enough for category retrieval

# Discussion

## Fast similarity scheme

- Fast similarity search working with non explicit kernels and with all fast knn search methods
- Good trade-off between Precision and Speed for R=5.2: 10 time faster and median precision 99%

## But ...

TOPN not good enough for category retrieval
Is it RETIN compatible ?

# Discussion

## Fast similarity scheme

- Fast similarity search working with non explicit kernels and with all fast knn search methods
- Good trade-off between Precision and Speed for R=5.2: 10 time faster and median precision 99%

## But ...

TOPN not good enough for category retrieval
Is it RETIN compatible ?
Need adaptation for online category learning

# Scalable active learning

**Introduction to fast online retrieval**

Can we decrease the complexity of Active Learning using similar strategy than ICPR08 ?

# Scalable active learning

### Introduction to fast online retrieval

Can we decrease the complexity of Active Learning using similar strategy than ICPR08 ?
Not straightforward to combined fast similarity schemes with online/Active Learning.

# Scalable active learning

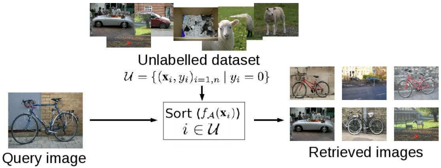
## Introduction to fast online retrieval

Can we decrease the complexity of Active Learning using similar strategy than ICPR08 ?

Not straightforward to combined fast similarity schemes with online/Active Learning.

Active Learning schemes: at least a complexity linear regarding the size of the database. $\Rightarrow$ impracticable for large database.

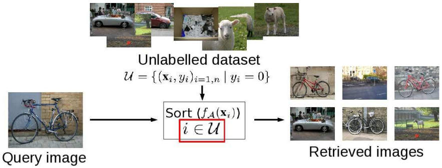
# Scalability Problem




Active Learning have 2 problems of scalability. The database have to be sorted to extract :

- the relevant images for intermediate results.
- the most uncertain images for annotation strategy.

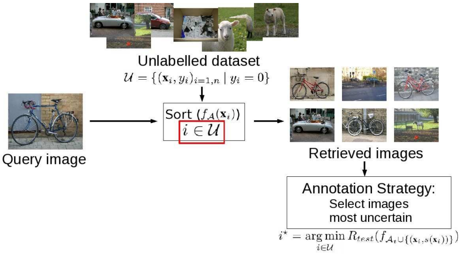
# Scalability Problem




Active Learning have 2 problems of scalability. The database have to be sorted to extract :

- the relevant images for intermediate results.
- the most uncertain images for annotation strategy.

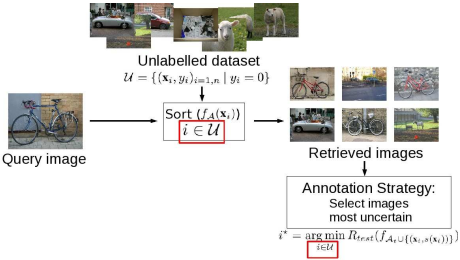
# Scalability Problem




Active Learning have 2 problems of scalability. The database have to be sorted to extract :

- the relevant images for intermediate results.
- the most uncertain images for annotation strategy.

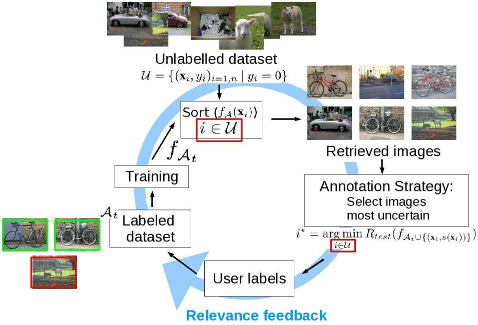
# Scalability Problem




Active Learning have 2 problems of scalability. The database have to be sorted to extract :

- the relevant images for intermediate results.
- the most uncertain images for annotation strategy.

# Scalability Problem

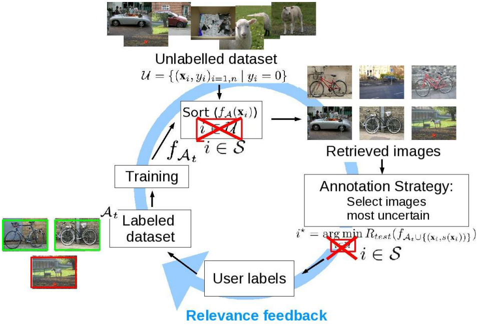



Active Learning have 2 problems of scalability. The database have to be sorted to extract :

- the relevant images for intermediate results.
- the most uncertain images for annotation strategy.

These scalability problems occure at each feedback loop

# Scalability Problem

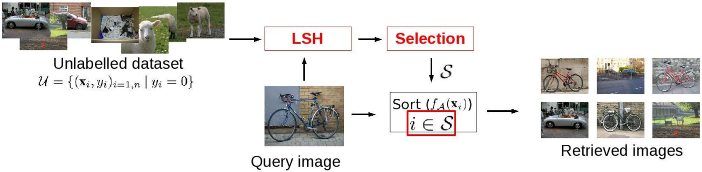



Active Learning have 2 problems of scalability. The database have to be sorted to extract :

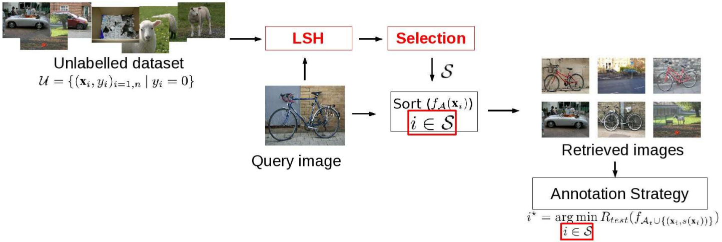
- the relevant images for intermediate results.
- the most uncertain images for annotation strategy.

We tackle these problems by considering only a relevant subset $\mathcal{S}$ instead of $\mathcal{U}$.

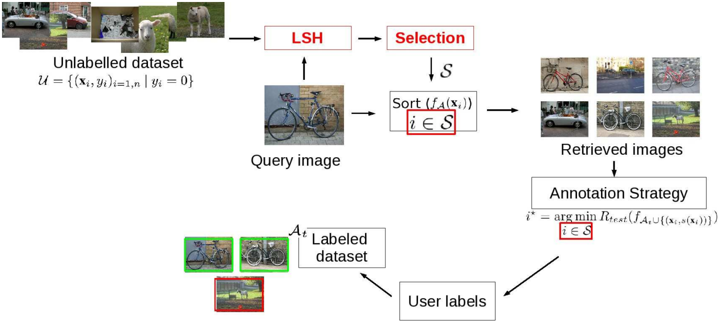
# Our algorithm scheme



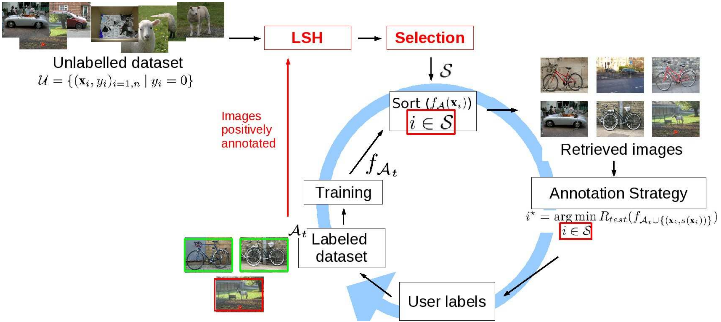
# Our algorithm scheme



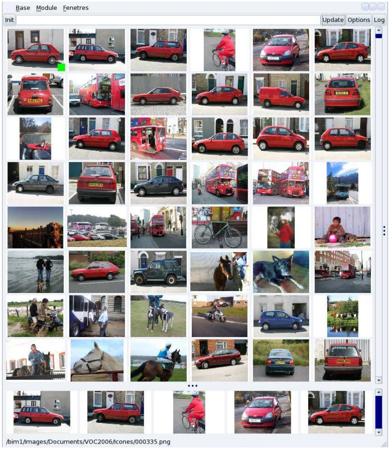
# Our algorithm scheme

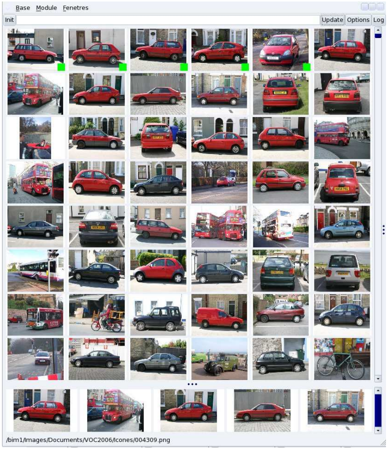


# Our algorithm scheme



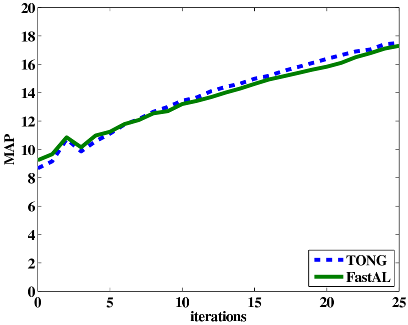
# Example



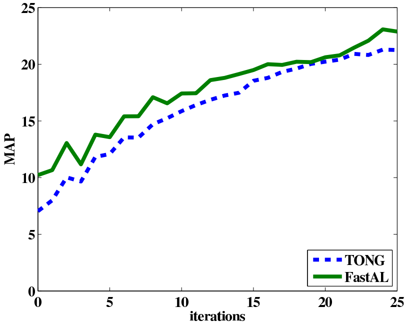




Each image is represented by a 192-dimension vector: 64 chrominances CIE Lab and 2 histograms of 64 textures from Gabor filters.

# Experiments



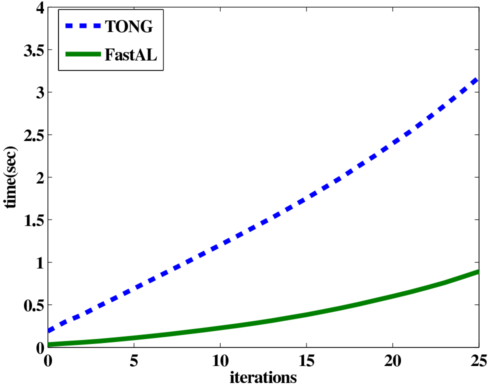

Mean Average Precision(%) of the 10 classes




Mean Average Precision(%) for an interactive search of the class CAR

- Performances are evaluated with Mean Average Precision of the TOP500, i.e., the sum of the Precision/Recall curve for the first 500 images retrieved.
- E2LSH parameters are R = 16.0 and L = 30 hash tables of K = 20 projections.

# Experiments




Average time of an interactive search function of the number of iteration

## Bibliography pieces


- Summarization Sheme Based on Near-duplicate Analysis, D. Gorisse, F. Precioso, M. Cord, S. Philipp-Foliguet, ACM International Conference on Multimedia, page 1–20 - 2008
- Fast Approximate Kernel-based Similarity Search for Image Retrieval Task, D. Gorisse, M. Cord, F. Precioso, S. Philipp-Foliguet, 19th ICPR International Conference on Pattern Recognition - dec 2008
- Actor Retrieval System Based on Kernels on Bags of Bags, S. Zhao, F. Precioso, M. Cord, S. Philipp-Foliguet, 16th European Signal Processing Conference (EUSIPCO) - August 2008
- Spatio-Temporal Tube Kernel for Actor Retrieval, in IEEE International Conference on Image Processing, S. Zhao, F. Precioso, M. Cord, Cairo, Egypt , November 7-11, 2009
- Optimization on Active Learning Strategy for Object Category Retrieval ,D. Gorisse, M. Cord, F. Precioso, Cairo, Egypt , November 7-11, 2009

# Conclusion

Let us switch to a demo !

## People

Matthieu Cord
LIP6, Univ. UPMC-PARIS 6
matthieu.cord@lip6.fr

Frédéric Precioso
ETIS UCP
precioso@ensea.fr

- Shuji Zhao (PhD student),
- David Gorisse (PhD student),
- Guillermo Chavez (PhD work)

All our collegues of LIP6 lab, ETIS team and Arnaldo de Albuquerque from our web page:
http://webia.lip6.fr/∼cord/
http://frederic.precioso.free.fr