UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CONTENT-BASED VIDEO COPY DETECTION

TESIS PARA OPTAR AL GRADO DE DOCTOR EN CIENCIAS, MENCIÓN COMPUTACIÓN

JUAN MANUEL BARRIOS NÚÑEZ

PROFESOR GUÍA:
BENJAMÍN BUSTOS CÁRDENAS

MIEMBROS DE LA COMISIÓN:
GONZALO NAVARRO BADINO
JORGE PÉREZ ROJAS
SHIN'ICHI SATOH

Este trabajo ha sido parcialmente financiado por CONICYT.

SANTIAGO – CHILE
NOVIEMBRE 2013

Resumen

La cantidad y el uso de videos en Internet ha aumentado exponencialmente durante los últimos años. La investigación académica en tópicos de videos se ha desarrollado durante décadas, sin embargo la actual ubicuidad de los videos presiona por el desarrollo de nuevos y mejores algoritmos. Actualmente existen variadas necesidades por satisfacer y muchos problemas abiertos que requieren de investigación científica. En particular, la Detección de Copias de Video (DCV) aborda la necesidad de buscar los videos que son copia de un documento original. El proceso de detección compara el contenido de los videos en forma robusta a diferentes transformaciones audiovisuales. Esta tesis presenta un sistema de DCV llamado P-VCD, el cual utiliza algoritmos y técnicas novedosas para lograr alta efectividad y eficiencia.

Esta tesis se divide en dos partes. La primera parte se enfoca en el estado del arte, donde se revisan técnicas comunes de procesamiento de imágenes y búsqueda por similitud, se analiza la definición y alcance de la DCV, y se presentan técnicas actuales para resolver este problema. La segunda parte de esta tesis detalla el trabajo realizado y sus contribuciones al estado del arte, analizando cada una de las tareas que componen esta solución, a saber: preprocesamiento de videos, segmentación de videos, extracción de características, búsqueda por similitud y localización de copias.

En relación a la efectividad, se desarrollan las ideas de normalización de calidad de videos, descripción múltiple de contenidos, combinación de distancias, y uso de distancias métricas versus no-métricas. Como resultado se proponen las técnicas de creación automática de descriptores espacio-temporales a partir de descriptores de fotogramas, descriptores de audio combinables con descriptores visuales, selección automática de pesos, y distancia espacio-temporal para combinación de descriptores.

En relación a la eficiencia, se desarrollan los enfoques de espacios métricos y tabla de pivotes para acelerar las búsquedas. Como resultado se proponen una búsqueda aproximada utilizando objetos pivotes para estimar y descartar distancias, búsquedas multimodales en grandes colecciones, y un índice que explota la similitud entre objetos de consulta consecutivos.

Esta tesis ha sido evaluada usando la colección MUSCLE-VCD-2007 y participando en las evaluaciones TRECVID 2010 y 2011. El desempeño logrado en estas evaluaciones es satisfactorio. En el caso de MUSCLE-VCD-2007 se supera el mejor resultado publicado para esa colección, logrando la máxima efectividad posible, mientras que en el caso de TRECVID se obtiene una performance competitiva con otros sistemas del estado del arte.

Abstract

The amount of digital videos on Internet has grown exponentially over the last few years. Academic research on video topics has developed over many decades, however the current ubiquity of videos pushes the development of new and better algorithms. There are currently many unsatisfied needs and open problems that demand scientific research. In particular, Content-Based Video Copy Detection (CBVCD) addresses the need to retrieve videos in a collection that are copies of some original document. The copy detection process relies exclusively on the audiovisual content, and must be robust to common audiovisual transformations. This thesis details a novel CBVCD system, called P-VCD. The system is founded on the metric space approach and develops many novel algorithms and techniques in order to achieve high effectiveness and efficiency.

The thesis is divided into two parts. The first part gives an overview of the area, presents different techniques from image processing and similarity search, analyzes the definition and scope of the CBVCD problem, and summarizes the state-of-the-art on the CBVCD topic. The second part details our approach for CBVCD, following the tasks of the detection process, namely: video preprocessing, video segmentation, feature extraction, similarity search, and copy localization.

Regarding effectiveness, we explore the ideas of video quality normalization, multiple content description, combination of distances, and metric versus non-metric distances. In particular, we propose techniques for the automatic creation of spatio-temporal descriptors using frame-based global descriptors, an acoustic descriptor that can be combined with global descriptors, automatic weight selection, and spatio-temporal distance to combine descriptors.

Regarding efficiency, we explore the ideas of metric access methods and pivot tables in order to reduce the amount of distance computations. In particular, we propose a novel approximate search that uses pivot objects in order to estimate and discard distance evaluations, a multimodal search in large datasets, and a novel index structure that exploits the similarity between consecutive query objects.

This thesis has been evaluated by using the MUSCLE-VCD-2007 dataset and by participating in TRECVID 2010 and 2011. We are very pleased with the performance achieved in both evaluations. In the case of MUSCLE-VCD-2007, the system outperforms the best published result for that dataset, achieving the maximum detection effectiveness, whereas in the case of TRECVID it shows competitive performance with other state-of-the-art systems.

Dedicado a Solange, Daniel y Elizabeth.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

The creation, distribution, storage, and broadcasting of digital videos has proliferated enormously in the last decade. In fact, a regular study on internet traffic shows that more than 50% of the current traffic corresponds to streams and downloads of videos for viewing on a PC screen [Cisco Systems Inc., 2012]. This traffic includes user generated videos (i.e., YouTube-like videos), online movies (i.e., Netflix-like videos), television broadcasts, webcam viewing, and mobile phone videos. Moreover, the study projects that the video traffic will represent more than 80% of the global consumer traffic on Internet by 2016. This is even more impressive if we acknowledge that video traffic made up less than 5% by 2005 [Cisco Systems Inc., 2008] (see Figure 1.1). The exponential growth of videos on the Internet can mostly be explained by the fall in the prices of digital cameras and video capture devices, the increase in network bandwidth, the emergence of devices showing online content, and the spread of mobile phones with embedded cameras.

Academic research on video topics has developed over many decades, however the current ubiquity of videos pushes the development of new and better algorithms and techniques. Regarding video topics, there are currently many open problems to address, novel applications to develop, and unsatisfied needs to fulfill. Most users would be interested in enhancing their possibilities to manage personal video collections, analyze large videos datasets, extract information in realtime from streams of videos, automatize processes using digital cameras, etc.

Multimedia Information Retrieval (MIR) is the research area that aims at searching and retrieving semantic information from multimedia documents [Lew et al., 2006]. A multimedia document can be understood as any repository of information, structured or not. It includes image,

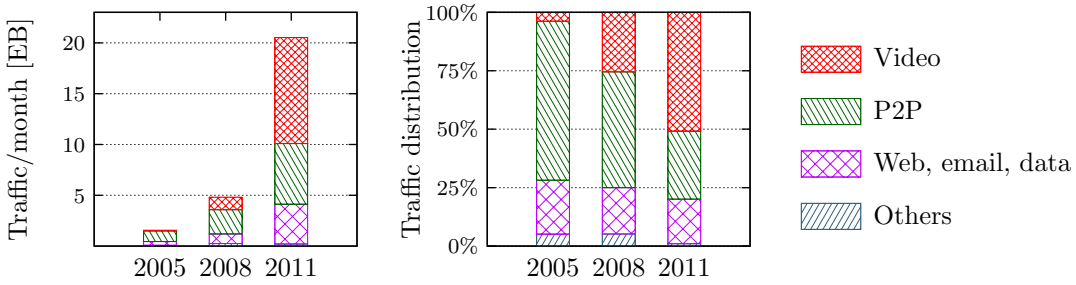


Figure 1.1 – Internet users data traffic per month for 2005, 2008 and 2011 [Cisco Systems Inc., 2008, 2012].

audio and videos, but also comprises other sources of information like time series, webpages, graphs, XML documents, DNA sequences, etc. In general, MIR systems follow two steps: a process for summarizing the content of the documents into *descriptors*, and a process for analyzing the descriptors in order to create knowledge.

Two classes of descriptors can be distinguished in MIR systems, namely high-level and low-level descriptors. High-level descriptors represent semantic features of documents, usually in the form of metadata. These descriptors usually have a human basis, i.e., they are produced by a person describing the content of a document, either directly (by assigning concepts, tags, text annotations, or other metadata) or indirectly (by derivations of direct metadata or by capturing the user behavior on the document). On the other hand, low-level descriptors represent features of the content itself, usually in the form of statistics and patterns. These descriptors are usually produced automatically by computers following different algorithms.

Content-Based MIR (CBMIR) aims to fulfill the objectives of MIR but relying exclusively on the content of the documents, i.e., by analyzing and producing semantic information from low-level descriptors and limiting the use of user generated metadata. The content-based techniques enable the use of MIR in cases when large datasets lack high-level descriptors, or when it is unfeasible to ask users to create descriptors. There is a broad field of research that addresses the problem of automatically creating high-level descriptors from low-level descriptors, i.e., to predict the concepts that a person would assign to a document given its content. The lack of coincidence between high-level descriptors that a user would create for a document and the high-level descriptors that a computer automatically produce is known as *Semantic Gap* [Smeulders et al., 2000]. Even in cases where enough semantic descriptors exist, content-based techniques can improve the effectiveness of MIR systems by allowing them to analyze an additional source of information [Lew et al., 2006].

Content-Based Video Retrieval (CBVR) aims at searching and retrieving videos in a collection based on their audiovisual content. Common use cases for CBVR systems are the organization and browsing of video collections, the retrieval of videos that "look similar" to a query video, and the localization of videos with specific features. As many image processing and image analysis techniques are needed, CBVR might be seen as a natural extension of content-based image retrieval. However, large differences are produced by the inclusion of the temporal dimension (like video segmentation and tracking of objects), the existence of acoustic and visual tracks (like fusing information from different sources), and the usually huge amount of data stored in video collections.

Content-Based Video Copy Detection (CBVCD) is a problem from CBVR that aims at detecting and retrieving video documents that are copies of some original document. The copy detection process must rely exclusively on the audiovisual content, ignoring any metadata associated with videos. Depending on the definition of what a copy is, CBVCD systems must detect either semantically identical videos or audio/visual derivatives. A derivative is a video that uses any part of the original, including edition processes such as extracting scenes, video-montages, and chroma key composition (green-screens), therefore a copy does not necessarily look similar to the original.

In general, the videos retrieved by CBVR systems may differ from videos retrieved by CBVCD systems. For example, assume we have a clip with the best point from a tennis game. By giving that clip to a common CBVR system, it may retrieve videos containing other points of the game, other games of the tournament, or any other tennis game. In turn, a CBVCD system will retrieve videos containing that specific point, like a news program reviewing the game, fan videos with the best tennis points of the history, or maybe no videos at all (if it did not locate any duplicate).

The detection performed by a CBVCD system should achieve both high effectiveness and

high efficiency. *Effectiveness* measures the quality of the answer of the system. *Efficiency* measures the resources used by the system to produce the answer. A CBVCD system with high effectiveness will retrieve all the videos that indeed are copies, without including *false alarms* (i.e., those videos that are not copies). A CBVCD system with high efficiency will require less memory and disk storage to run, and will take less time to produce the answer.

When dealing with large datasets, the resources required to achieve the highest effectiveness usually produce very poor efficiency, i.e., comparing every possible video following every possible criteria commonly leads to unreasonable search time and storage needs. Hence, a common issue that CBVCD must address is the *effectiveness-versus-efficiency* trade-off. Moreover, even in the hypothetical case of unlimited resources, there are no guarantees to achieve the maximum effectiveness because the definition of copy is defined by the human perception. Therefore, another issue that CBVCD must address is to bridge the semantic gap between low-level content descriptors and what a user would state as a copy.

Finally, TRECVID is an evaluation and a workshop organized yearly by the National Institute of Standards and Technology (NIST), which promotes the research in video information retrieval [Smeaton et al., 2006]. Many teams from different universities and private companies around the world participate regularly in the proposed tasks. A specific evaluation for CBVCD systems was held at TRECVID between 2008 and 2011. This evaluation provided common datasets and evaluation measures to benchmark different CBVCD systems.

1.1 Thesis Overview

This thesis details a novel CBVCD system called P-VCD, which achieves competitive performance with other state-of-the-art systems. The system is founded on the metric space approach and develops many novel algorithms and techniques in order to achieve high effectiveness and efficiency.

P-VCD is divided into five main tasks, namely Preprocessing, Video Segmentation, Feature Extraction, Similarity Search, and Copy Localization. In particular, the Similarity Search task uses metric spaces to achieve both high effectiveness by defining smart similarity measures to compare objects, and high efficiency by using mathematical properties to reduce the number of comparisons required to perform a search. To the best of our knowledge, this is the first system which applies techniques from the metric spaces to address the CBVCD problem.

The system is evaluated in detail using the MUSCLE-VCD-2007 dataset [Law-To et al., 2007b], a medium-size dataset which provides a ground-truth for CBVCD. It contains a collection of original videos, a collection of query videos, and the expected answer for each query video. In addition to our own evaluations, we participated in the CBVCD evaluation at TRECVID between 2010 and 2011 as the "PRISMA" team, hence, the performance of our system has also been compared with other CBVCD systems.

This thesis comprises twelve chapters and three appendixes. It is divided in two parts:

Part I: Background and Related Work. This part introduces the areas of image processing, image analysis and metric spaces and reviews the state-of-the-art in the CBVCD topic.

- In Chapter 2 we briefly review the techniques of image processing and image analysis that we

use in this thesis.

- In Chapter 3 we give the background for the similarity search process, and present some general techniques to improve effectiveness and efficiency in a search.
- In Chapter 4 we analyze the definition and scope of the CBVCD problem and common applications, and we summarize the related work on the CBVCD topic.

Part II: CBVCD and the Metric Space Approach. This part details our approach for CBVCD following each of its tasks.

- In Chapter 5 we give a general overview of the P-VCD system and its five main tasks. We also present the MUSCLE-VCD-2007 dataset used for the evaluation of the following chapters.
- In Chapter 6 we detail the Preprocessing task. We define a process for quality normalization of videos, and a process for detection and reversion of content transformations.
- In Chapter 7 we detail the Video Segmentation and the Feature Extraction tasks. We show the low-level descriptors used by the system, and we perform a basal evaluation of their effectiveness. In particular, we present six different global descriptors, an approach for converting them into spatio-temporal descriptors, and a novel acoustic descriptor. The most important conclusions from the experiments in this chapter are:
  - The quality normalization at the preprocessing task improves the effectiveness of all the evaluated global descriptors.
  - The spatio-temporal description improves the effectiveness of all the evaluated global descriptors.
  - The descriptor based on edge orientations achieves higher effectiveness than the other evaluated global descriptors, when using both quality normalization and spatio-temporal description. Moreover, the achieved effectiveness outperforms the state-of-the-art systems evaluated on the MUSCLE-VCD-2007.
  - The proposed acoustic descriptor achieves high detection effectiveness. It also outperforms the state-of-the-art systems evaluated on the MUSCLE-VCD-2007.
- In Chapter 8 we detail the developed techniques for improving the effectiveness of the Similarity Search task. We analyze the weighted combination of distances, the spatio-temporal combination of distances, and the use of non-metric distances. In this chapter we propose three novel techniques to automatize the selection of weights in a combination: the α-normalization, which scales different distances in order use them in a weighted combination of distances, the weighting by max-ρ algorithm, which automatically allots weights that maximizes the intrinsic dimensionality of the combination, and the weighting by max-τ algorithm automatically, which automatically allots weights that maximizes the value that α-normalizes the combination. The most important conclusions from the experiments in this chapter are:
  - Using the α-normalization, weighting by max-ρ, and weighting by max-τ algorithms, it is possible to automatically locate a set of weights that achieve high effectiveness without requiring the use of training data.
  - The spatio-temporal combined distance can improve the effectiveness. In fact, the temporal weighted combination of global and acoustic descriptors can achieve the maximum effectiveness for MUSCLE-VCD-2007, i.e., to detect all the copies without giving any false alarm. This result outperforms the state-of-the-art systems evaluated on this dataset.

  - The use of some common non-metric distances does not produce a noticeable improvement in effectiveness.

- In Chapter 9 we detail the developed techniques for improving the efficiency of the Similarity Search task. In particular, we focus on the use of static and dynamic pivots. In this chapter we propose the Approximate Search with Pivots, which uses a fast estimator to discard most of the irrelevant objects of the search; the Approximate Search with Pivots for Local Descriptors, which is an adaptation of previous search to local descriptors; the Two-step search, which divides a combined search into an approximate search and an exact search; and the Snake Table, which is a pivot table using dynamic pivots. The most important conclusions from the experiments in this chapter are:
  - The Approximate Search with Pivots shows a satisfactory *effectiveness-versus-efficiency* trade-off. The search time can be drastically reduced at a small cost in the effectiveness.
  - The Approximate Search with Pivots for Local Descriptors can increase both effectiveness and efficiency of local descriptors when it is compared with the exact search.
  - The Snake Table can improve the efficiency without decreasing the effectiveness by using previous query objects as dynamic pivots.

- In Chapter 10 we detail the techniques used at the Copy Localization task. We determine the boundaries of a copy by using a voting algorithm between the objects retrieved by the similarity search. The most important conclusions from the experiments in this chapter are:
  - The voting algorithm can improve its effectiveness when it considers more than one similar object by weighting the votes according to rank positions.
  - The inclusion of similarity values in the voting algorithm has almost no impact on the effectiveness.

- In Chapter 11 we review our participation in the CBVCD evaluation at TRECVID 2010 and 2011. Here, we tested most of the proposed algorithms. The most important conclusions from the experiments in this chapter are:
  - P-VCD achieves competitive performance with other state-of-the-art CBVCD systems, especially in detecting copies without false alarms.
  - The Preprocessing task has a high impact on increasing the effectiveness of detection.
  - The Two-step search can improve the effectiveness, especially for multimodal searches.
  - The metric space approach can successfully be applied to CBVCD systems: it can improve the effectiveness by enabling a novel method to combine multimodal information at the search time, and it can improve the efficiency by enabling the use of mathematical properties to accelerate searches.

Finally, in Chapter 12 we conclude this thesis by summarizing our main contributions, and offering some final thoughts regarding this work. Additionally, we outline different research trends we plan to develop in future work.

The thesis also includes three appendixes: Appendix A presents the ground-truth for the MUSCLE-VCD-2007 dataset, and Appendix B and Appendix C publish some data provided by NIST for the CBVCD evaluation at TRECVID 2010 and TRECVID 2011, respectively. We provide these data for academic purposes, in order to give insight into this thesis and its contributions.

P-VCD has been released as an Open Source Project under the GNU General Public License version 3.0 (GPLv3). Most of the source code used in this thesis can be freely downloaded from its website[1].

1.2 Thesis Publications

The main contributions in this thesis have been published in the following research papers:

[**Barrios and Bustos, 2009**] This work, developed during the preparation of this thesis, presents a content-based image retrieval system that combines low-level features with high-level features following the metric space approach.

[**Barrios, 2009**] It is the proposal of this thesis. This work reviews the related work at the CBVCD topic and proposes to use the metric space approach to address the problem.

[**Barrios and Bustos, 2010**] This work summarizes the first participation of the PRISMA team in the CBVCD evaluation at TRECVID 2010, described in Chapter 11.

[**Barrios and Bustos, 2011a**] This work presents the Approximate Search with Pivots algorithm and the Copy Localization algorithm, described in Chapter 9 and Chapter 10, respectively.

[**Barrios and Bustos, 2011b**] This work presents the weighting by max-ρ algorithm, described in Chapter 8.

[**Barrios, Bustos, and Anguera, 2011**] This work summarizes the second participation of the PRISMA team in the CBVCD evaluation at TRECVID 2011, described in Chapter 11.

[**Barrios and Bustos, 2013**] Journal paper. This work details the achieved results at TRECVID 2010 and it presents the weighting by max-τ algorithm, described in Chapter 8.

[**Barrios, Bustos, and Skopal, 2012**] This work presents the Snake Table, described in Chapter 9.

[**Barrios, Bustos, and Skopal, 2013**] Journal paper. This work presents the Snake Table and gives new insights about the analysis and indexing of query sets.

This thesis also considers a joint work with Telefonica Research. The results of that collaboration have been published in:

[**Anguera, Adamek, Xu, and Barrios, 2011a**] This work summarizes the participation of Telefonica Research team in the CBVCD evaluation at TRECVID 2011. That participation included a joint submission between the Telefonica team and PRISMA team, described in Chapter 11.

[1] P-VCD: `http://sourceforge.net/projects/p-vcd/`

[Anguera, Barrios, Adamek, and Oliver, 2011b] This work presents a late fusion algorithm for multimodal detection in CBVCD systems, described in Chapter 4.

This thesis also considers some unpublished results.

Part I

Background and Related Work

Chapter 2

Image Processing and Analysis

In this chapter we briefly review the image processing and image analysis techniques that are used in this thesis. The content of this chapter is mainly extracted from the books Gonzalez and Woods [2007], Bradski and Kaehler [2008], and Szeliski [2010].

2.1 Concepts

Image In this thesis, we define *image* as a two-dimensional function $I(x, y)$, where x and y are called the spatial coordinates. The coordinates are bounded integer values $x \in [0, width - 1]$ and $y \in [0, height - 1]$, hence the image consists in $width \times height$ picture elements or *pixels*.

Gray images In the case of gray images, the value of $I(x, y)$ is a single value, representing the *gray level* or *intensity* for each pixel. The *depth* of the image is the number of bits used to indicate the value for each pixel. Commonly used depths are 8-*bits*, where each value is an integer between 0 (black) and 255 (white), and 32-*bits*, where each value is a floating point number between 0 (black) and 1 (white). Additionally, the 16-*bits-signed* depth, where each value is an integer between -32,768 and 32,767, is commonly used for storing intermediate values during the processing of 8-bits depth images.

Color images In the case of color images, a color space must be first selected. The *color space* (also called *color model* or *color system*) specifies a multidimensional space, a coordinate system, and a region in that space in which a color corresponds to a single multidimensional point. Each coordinate of the color space is called a *color channel*. Most color spaces define three coordinates, thus the value of $I(x, y)$ is a triplet (a, b, c) containing the value for each of the three channels. Alternatively, a color image can be seen as a combination of three gray images I_a, I_b, I_c each one containing the value for each channel, therefore $I(x, y) = (I_a(x, y), I_b(x, y), I_c(x, y))$.

RGB color space The most common color space is RGB, which represents colors as a uniform combination of primary colors *red*, *green* and *blue*. In this case, the value of each pixel is a triplet containing the value for each channel, i.e., $I(x, y)=(r, g, b)$. The region containing RGB colors is usually represented by a unitary cube (see Figure 2.1). In this cube, the gray intensities lie in the

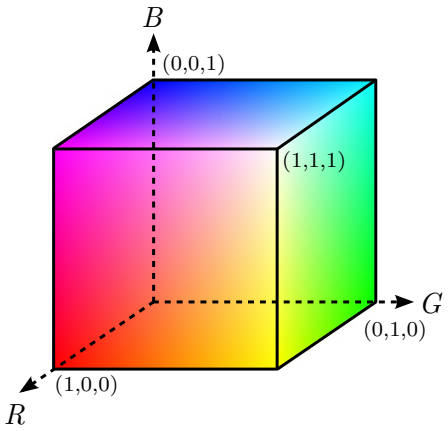


Figure 2.1 – The RGB unitary cube.

main diagonal with r=g=b. The RGB space is usually used to display and store images due to its closeness with displaying hardware. Frequently, RGB images are stored using 24-*bits* color depth, which corresponds to use 8-bits depth to each of the three channels. This color depth, also known as *full-color* or *true-color*, can represent up to $(2^8)^3$=16,777,216 colors.

HSV color space The HSV color space represents colors as a combination of three characteristics: *Hue*, *Saturation*, and *Value*. Hue is a value that represents a pure color (e.g., yellow, red, etc.). Saturation measures the colorfulness of the color, i.e., the degree in which a color is pure (full saturation) or is diluted with white light (no saturation). Value measures the brightness of the color between black and its pure color. The region containing HSV colors is usually represented by a cylinder, where Hue is a circular coordinate, Saturation is a radial coordinate, and Value is a vertical coordinate. The HSV space is created by a geometrical transformation of the RGB space. The colors in HSV space with medium Value are a circular mapping of the colors in RGB space that lie in the main triangle perpendicular to the gray diagonal in the RGB cube. The HSV space intends to represent colors in a space closer to the way humans describe and represent colors, therefore it is usually used for color manipulation and image representation. An image stored with RGB space needs to be converted into the HSV space following a mathematical conversion between both color spaces.

Histograms Given n observations, the *histogram* counts the number of observations that fall into each of k disjoint categories (known as *bins*)). Given a gray image I of size $W \times H$ with n gray levels (e.g., n=256 for 8-bits depth), the normalized gray histogram is a function h defined as:

$$h(k) = \frac{n_k}{WH}$$

where n_k is the number of pixels of I containing the k^{th} gray level. The gray histogram (or intensity histogram) represents the distribution of gray levels in the image, discarding the spatial locations. It provides useful information about the global content of the image that can be used either for image processing or for image analysis.

Analogously, the color histogram represents the distribution of colors in the image, discarding the spatial locations. In the case of full-color images, a common approach to reduce the number of bins consists in partitioning the RGB color space following a regular grid producing $n_r \times n_g \times n_b$

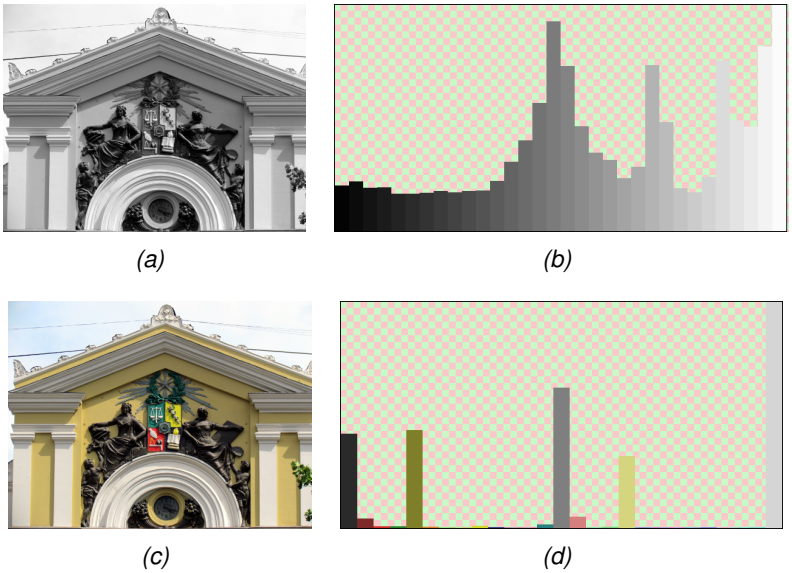


Figure 2.2 – Gray and Color Histogram. (a) Sample gray image. (b) 32-bins gray histogram. (c) Sample color image. (d) 27-bins RGB histogram (regular grid n_r=n_g=n_b=3).

bins, where n_r, n_g and n_b are the number of ranges in which each dimension of the RGB cube is divided (see Figure 2.2).

2.2 Image Processing

Point operations The point operators modify each pixel of the image independently of the others. Given an input image I and a point operator f, the output image G is defined as $G(x, y) = f(I(x, y))$. Some examples of these operators are: the conversion between color spaces, the gamma correction (defined by the function $f(x) = x^\gamma$), the adjustment of brightness and contrast (defined by $f(x) = ax + b$ with constants $a > 0$ and b), the binarization or thresholding ($f(x) = 0$ iff $x < t$, $f(x) = 1$ otherwise, for some predefined threshold t), and the histogram equalization which takes the histogram of the image and defines a function $f(x)$ that produces a flatter histogram.

Convolution In image processing, the *convolution* is a linear operator (denoted by $*$) that filters an image using second smaller image (known as *filter*, *mask*, or *kernel*). It is also known as *linear filtering* or *spatial filtering*. Given a gray image I of size $W \times H$ and a filter w of size $a \times b$ the convolution produces an image G of size $W \times H$ defined as:

$$G(x, y) = (I * w)(x, y) = \sum_{i=0}^{a-1} \sum_{j=0}^{b-1} I(x - \lfloor a/2 \rfloor + i, y - \lfloor a/2 \rfloor + j) \cdot w(i, j)$$

There are several way to handle the convolution in the boundaries of I. In this thesis, we extend the first and last column and row of I to contain the whole filter, i.e. $I(x, y) = I(\max\{0, \min\{x, W - 1\}\}, \max\{0, \min\{y, H - 1\}\})$. In order to have a precise definition of the center of the filter, the size of the filter is usually restricted to odd numbers.

$\frac{1}{9}\times$

1	1	1
1	1	1
1	1	1

(a) 3×3 Average.

$\frac{1}{16}\times$

1	2	1
2	4	2
1	2	1

(b) 3×3 Gaussian.

Figure 2.3 – Blurring filters.

The convolution for color images is usually performed by separating the image into its color channels, convolving each channel with the filter, and combining the filtered channels into the final image, i.e.:

$$G_{rgb}(x,y) = (I_{rgb} * w)(x,y) = ((I_r * w)(x,y), (I_g * w)(x,y), (I_b * w)(x,y))$$

Blur The blurring of images is frequently used to remove small details and filter noise. It is usually performed by the convolution of the image with an average filter or Gaussian filter. The average filter is a square matrix with all values equal summing to 1. The Gaussian filter is created by adjusting a 2D Gaussian function into the center of the filter and adjusting the standard deviation according to the size of the filter. The values in the Gaussian filter are normalized to sum 1. Given a standard deviation σ, the 2D Gaussian function is defined as:

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

One alternative to control the amount of blur in the image, is to perform a convolution with a larger filter, e.g., a 5×5 average filter will produce more blur than a 3×3 average filter. Another alternative is to leave the filter unmodified and perform many successive convolutions with the image.

Gradient The gradient of a two dimensional function I is defined as:

$$\nabla I = \begin{bmatrix} I_x \\ I_y \end{bmatrix} = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix}$$

The gradient $\nabla I(x,y)$ is a vector which represents the increase rate and the orientation of the maximum increase of I at the point (x,y). The magnitude and the orientation of the gradient are given by:

$$\text{mag}(\nabla I) = \sqrt{I_x^2 + I_y^2} \approx |I_x| + |I_y| \qquad \theta(\nabla I) = \arctan\left(\frac{I_y}{I_x}\right) \tag{2.1}$$

Because images are two dimensional discrete functions, the partial first-order derivatives are defined as:

$$I_x(x,y) = \frac{\partial I}{\partial x}(x,y) = I(x+1,y) - I(x,y)$$
$$I_y(x,y) = \frac{\partial I}{\partial y}(x,y) = I(x,y+1) - I(x,y)$$

-1	0	1
-2	0	2
-1	0	1

(a) Sobel filter for I_x.

-1	-2	-1
0	0	0
1	2	1

(b) Sobel filter for I_y.

Figure 2.4 – Sobel filters.

0	1	0
1	-4	1
0	1	0

Figure 2.5 – Laplacian filter.

The Sobel operators are filters that are frequently used to approximate I_x and I_y. Figure 2.4 shows two commonly used Sobel operators. The gradient for each pixel can be approximated by performing and independent convolution between I and each Sobel operator, and then using Equation 2.1 to combine them.

Laplacian The Laplacian is an operator defined as the sum of the second-order derivatives, hence the Laplacian of a two dimensional discrete function I is:

$$\nabla^2 I = I_{xx} + I_{yy}$$

where the second-order derivatives correspond to:

$$I_{xx}(x,y) = \frac{\partial^2 I}{\partial x^2}(x,y) = I(x+1,y) + I(x-1,y) - 2I(x,y)$$
$$I_{yy}(x,y) = \frac{\partial^2 I}{\partial y^2}(x,y) = I(x,y+1) + I(x,y-1) - 2I(x,y)$$

The Laplacian can be calculated with a convolution of I with the filter shown in Figure 2.5. It is usually used for sharpening images, i.e. to enhance details that have been blurred, and for edge detection, as described next.

Edge Detection A common approach for edge detection is the gradient approach. It is based on two steps: first analyzing the first-order derivative of the image and then analyzing the second-order derivative. It should be noted that an edge is a "local" concept, i.e., the decision of marking a pixel as an edge does not affect the decision for other pixels. The first step calculates the magnitude of the gradient for each pixel, either using the Sobel operator or another technique. A pixel is marked as an edge when the magnitude of the gradient for that point is higher than a predefined threshold t. The second step uses the sum of the second-order derivatives in order to produce thin edges. A pixel is marked as an edge if it was marked by the first step and the Laplacian of the pixel is zero.

This approach was formalized by Canny [1986], who also developed two improvements for the edge selection: to follow the perpendicular direction of the gradient, i.e., to select pixels by

(a) (b) (c)

Figure 2.6 – Edge selection. (a) Sample gray image. (b) Edges according to Canny algorithm. (c) Edges according to DoG algorithm.

"walking" on the edge, and to use an uncertainty threshold $t' < t$ in order to mark edge pixels with gradient magnitude between t' and t when it is next to another already marked pixel. These techniques enable the selection of thin and long edges (see Figure 2.6).

A second approach for edge detection is trough the use of a Difference of Gaussians (DoG). It consists in applying a Gaussian filter to an image and compare the blurred image against the original image. The subtraction between original and blurred images remarks zones with great variation in pixel values, therefore it detects edges as well as noise. In order to reduce the noise, the difference image is computed between two blurred images I_1 and I_2, where I_1 is the result of applying a Gaussian filter with standard deviation σ, and I_2 is the result of applying a Gaussian filter with standard deviation $k\sigma$, for a given step size k. Finally, the edges are detected by thresholding the absolute value of the difference (see Figure 2.6).

After the edge pixels have been determined, a common task is to analyze their distribution in order to locate shapes in the image. In particular, given a set C containing detected edge pixels, the line detection process consists of detecting sets of collinear edge pixels. The set C may contain pixels not belonging to any valid line (i.e., outliers), thus the detection process should be robust to them. The following section reviews two algorithms that are frequently used to locate collinear pixels: RANSAC and Hough Transform.

Random Sample Consensus (RANSAC) It is an randomized algorithm that iteratively tests and corrects many different lines.

Any line is defined by a pair of points, thus RANSAC randomly picks two pixels from C, creates a line $\mathcal{L}$, and calculates the number of inliers to $\mathcal{L}$ in C, i.e., locates the pixels in C that are traversed by $\mathcal{L}$. A correction cycle is usually performed to improve the convergence of RANSAC: between the located inliers to $\mathcal{L}$, the least-squares method is used to calculate a new corrected line $\mathcal{L}'$, and in turn the inliers for $\mathcal{L}'$ are located. The correction cycle ends when there are no new inliers located. In summary, the RANSAC algorithm works as follows:

- Iterate n times:
  1. Select random seed pixels $S \subseteq C$ with $|S| = 2$.
  2. $M \leftarrow S$.
  3. Use least-squares method to fit a line $\mathcal{L}_M$ for all pixels in M.
  4. Calculate the set of inliers $I_M \subseteq C$, which contains the pixels at a distance less or equal than ϵ to $\mathcal{L}_M$.

5. If $|I_M| > |M|$, then $M \leftarrow I_M$ and goto step 3.
6. Otherwise, end current iteration.

- Between all the evaluated lines, choose the line with more inliers, i.e., the $\mathcal{L}_M$ with maximum $|I_M|$.

The performance of RANSAC is controlled by parameters n and ϵ. RANSAC can succeed at detecting a line when there is a reasonable chance of selecting actual inliers as seeds at step 1. That chance increases linearly with the number of iterations, but it decreases quadratically with the number of outliers in C. Therefore, RANSAC is recommendable only when the ratio inliers/outliers is relatively high.

Hough Transform It is an exhaustive and deterministic algorithm that efficiently tests every possible line passing through every pixel in C.

All the lines passing through a pixel with coordinates (x_0, y_0) satisfy the equation:

$$y_0 = ax_0 + b \quad , \tag{2.2}$$

for varying values of a and b. The *parameter space* corresponds to the two-dimensional space where pairs (a, b) reside. Hence, a line $\mathcal{L}$ in the image corresponds to a point in the parameter space. The parameters for every line passing through (x_0, y_0) can be characterized by rewriting Equation 2.2:

$$b = -ax_0 + y_0 \tag{2.3}$$

Using Equation 2.3, the line $\mathcal{L}$ passing through many pixels in C can be detected by drawing a line in the parameter space for each pixel in C, and locating the pair (a', b') with most intersections. The efficiency of the Hough Transform proceeds from discretizing the parameter space into a voting table T with fixed-size cells, where each cell $T[i, j]$ represents a small interval of values $[a_i, a_i + \varepsilon] \times [b_j, b_j + \Delta]$.

However, an implementation issue arises from using Equation 2.3, since a and b may vary between $-\infty$ and $+\infty$. In order to use a bounded parameter space, Equation 2.2 can be replaced by the normal representation of lines. In that representation, a line $\mathcal{L}$ is defined by two parameters (see Figure 2.7): ρ, which is the minimum distance of $\mathcal{L}$ to the origin; and θ, which is the angle between the normal to $\mathcal{L}$ passing through the origin and the x-axis. Using the normal representation, all the lines passing through a pixel (x_0, y_0) satisfy the equation:

$$\rho(\theta) = x_0 \cos\theta + y_0 \sin\theta \tag{2.4}$$

The parameter space defined by θ and ρ is bounded. In fact, because only the lines passing through a pixel in the image need to be represented, θ is bounded to the range $[-\pi/2, \pi]$ and ρ is bounded to the range $[0, D]$, with $D{=}\sqrt{W^2 + H^2}$.

Voting for all the lines passing through (x_0, y_0) corresponds to summing one vote to every cell that satisfies Equation 2.4. A cell receiving k votes means there are k pixels sharing the same line (i.e., k collinear pixels), hence the line detection process consists in running the voting algorithm and selecting the most voted cell. In summary, the Hough Transform algorithm works as follows:

- Allocate table T with $n \times m$ cells, and initialize each cell to zero.

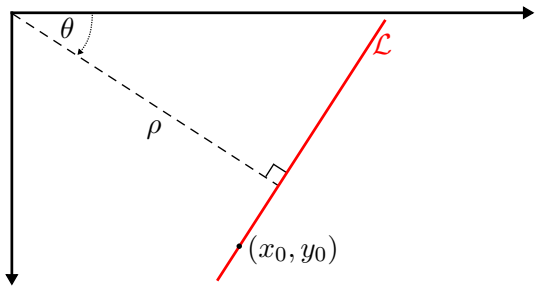


Figure 2.7 – The normal representation of line $\mathcal{L}$.

- Calculate Θ=$\{\theta_0, ..., \theta_{n-1}\}$ by uniform sampling the range $[-\pi/2, \pi]$.
- Calculate P=$\{\rho_0, ..., \rho_{m-1}\}$ by uniform sampling the range $[0, D]$.
- For each pixel $(x_0, y_0) \in C$:
  1. For each $\theta_i \in \Theta$:
     (a) Calculate $\rho(\theta_i)$ according to Equation 2.4.
     (b) If $\rho(\theta_i)$ is out of the range $[0, D]$, end current iteration continuing with θ_{i+1}.
     (c) Locate value $\rho_j \in P$ which is the closest to $\rho(\theta_i)$.
     (d) $T[i, j] \leftarrow T[i, j] + 1$.
- Locate cell $T[r, s]$ with more votes, and choose the line $\mathcal{L}$ determined by parameters θ_r and ρ_s.

The performance of the Hough Transform is controlled by parameters n and m. Their values present a tradeoff between accuracy and computational cost. Once n and m have been fixed, the processing time of the Hough Transform decreases linearly with the number of points (independently of the ratio inliers/outliers), therefore it is a convenient approach when there are a high amount of outliers.

RANSAC and Hough Transform are two methods for detecting lines in a set of pixels. Besides the line detection, these two algorithms can be generalized to detect subsets satisfying a given condition. For instance, the edge pixels lying in a circumference can be located either using RANSAC (randomly picking three pixels, generating a candidate model and locating inliers to the model) and Hough Transform (using a tridimensional parameter space comprising the center and radius). Moreover, these two algorithms can also be applied to object localization as described in next section.

2.3 Content Description

The content description consists in computing one or many *descriptors* (also called *feature vectors* or *fingerprints*) to represent some visual characteristic or feature of the image (like the distribution of colors, orientation of edges, etc.). The extracted descriptors usually correspond to multidimensional vectors or binary sequences.

The descriptors can be either *global* or *local*. A global descriptor represents the content of the whole image. A local descriptor represents the content of only a small zone of the image, hence

the whole image is represented by many local descriptors. The following subsections review some basic feature extraction methods for images. In Chapter 4 we review feature extraction methods specific to CBVCD.

2.3.1 Global Descriptors

Intensities A simple global description is to represent an image by the intensities of its pixels. Given an image I of size $W \times H$, convert I to a gray image I_Y, and create a vector with $W \cdot H$ dimensions, where each dimension corresponds to the intensity of a pixel in I_Y. If the dataset contains images of different sizes, then each image should be first scaled to a fixed size. A common distance to compare two vectors is the Euclidean distance. This approach is used, for example, in processes such as clustering images or PCA on images. Chapter 7 presents an evaluation of this descriptor using the name **KF**.

Histograms A widely-used global descriptor is to represent the image by its gray or color histogram. As described in Section 2.1, the gray or color histogram is a vector summarizing the gray intensities or the colors in the image, discarding the spatial locations. Chapter 7 presents an evaluation of this descriptor using the names **GH** (gray histogram), **IH** (histogram by channels), and **CH** (color histogram).

MPEG-7 descriptors MPEG-7 is a standard for describing multimedia documents that enables the attachment of high-level data (concepts and full-text description) and low-level data (content-based descriptors) to multimedia files. The low-level descriptions defined in the standard are frequently used in CBMIR systems. Manjunath et al. [2001] provides an overview of color and texture descriptors defined by this standard. The MPEG-7 eXperimentation Model[1] is the reference implementation of these descriptions. In this thesis we use the Edge Histogram, which is described in the following section.

Edge Histogram The Edge Histogram descriptor represents the spatial distribution of edges in the image [Manjunath et al., 2001]. The extraction method converts an image to gray scale, partitions it into 4×4 zones, and for each zone, it calculates a local edge histogram. To calculate the local edge histogram, each zone is scaled and divided in many 2×2 blocks. The main orientation of the edges for each block is determined by measuring the energy for five different orientation kernels (see Figure 2.8). If the kernel with maximum strength exceeds a certain threshold, the block is marked as an edge block. The local edge histogram for a zone is created by accumulating the orientations of its edge blocks. The descriptor is a vector with 80 dimensions ($4 \cdot 4 \cdot 5$), where each dimension is quantized into three bits/dimension. Two descriptors can be compared with the L_1 distance. Chapter 7 presents an evaluation of this descriptor using the name **EH**.

Ordinal Measurement The Ordinal Measurement descriptor captures the relative ordering of intensities in the image [Bhat and Nayar, 1998; Kim, 2003]. The extraction method converts an image into gray scale, and divides it into $n \times m$ zones. For each zone, the average intensity is calculated. The relative ordering between zones is expressed by their ranks. The descriptor is a

[1] MPEG-7 XM. ISO/IEC 15938-6:2003. Information technology – Multimedia content description interface – Part 6: Reference software.

1	-1
1	-1

(a) 180° edge

1	1
-1	-1

(b) 90° edge

$\sqrt{2}$	0
0	$-\sqrt{2}$

(c) 135° edge

0	$\sqrt{2}$
$-\sqrt{2}$	0

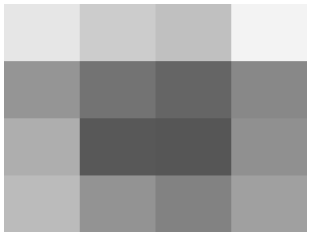
(d) 45° edge

2	-2
-2	2

(e) Isotropic 1

Figure 2.8 – Edge Histogram descriptor. (a-e) Five orientation filters (four directional and one isotropic).

(a)



(b)

230	204	192	243
149	115	101	136
174	88	86	144
187	147	130	160

(c)

15	14	13	16
9	4	3	6
11	2	1	7
12	8	5	10

(d)

Figure 2.9 – Ordinal Measurement 4×4 descriptor. (a) Sample gray image. (b-c) Average intensities for the image divided into 4×4 zones. (d) Rank assigned to each zone.

permutation of $(1, \ldots, n \cdot m)$ where each dimension is the rank for each zone after sorting them in ascending order by intensity (see Figure 2.9). Two descriptors can be compared with the L_1 or the Hamming distance. Chapter 7 presents an evaluation of this descriptor using the name **OM**.

2.3.2 Local Descriptors

The process of calculating local descriptors can be divided in two phases. The detection phase is the process which locates geometrically stable points under different transformations that are likely to be recurrent between different images of the same object, called *keypoints* or *interest points*. The description phase analyzes the neighborhood of a keypoint and represents the content of the image around it with a single descriptor.

Given two images, the comparison method commonly contains three steps: 1) extract local descriptors from each image, 2) match the most similar descriptors between both images, and 3) select a subset of matches with spatial consistency.

Once a subset of matches has been located, the quantification of the similarity between images may consist of counting the number of matched descriptors, or calculating the percentage of local descriptors in the first image that have a matching counterpart in the second image.

Keypoint detection The keypoint detection methods can be classified by its core technique: the Autocorrelation and the Difference of Gaussians. The autocorrelation consists of comparing a zone of the image with a small displacement of itself. The autocorrelation function can be defined for each pixel and displacement as:

$$E(x_0, y_0, \Delta_x, \Delta_y) = \sum_x \sum_y \left(I(x + \Delta_x, y + \Delta_y) - I(x, y)\right)^2 \cdot w(x - x_0, y - y_0)$$

where w is a weighting window that defines the boundaries of the zones. When the autocorrelation for a pixel (x_0, y_0) is high for any displacement, it implies that the pixel can be distinguished from its environment, and will likely be distinguishable even under geometric transformations. Hence, the keypoint detection is performed by analyzing this function. The first-order approximation of I is:

$$I(x + \Delta_x, y + \Delta_y) = I(x, y) + I_x(x, y) \cdot \Delta_x + I_y(x, y) \cdot \Delta_y$$

Replacing that approximation into the autocorrelation function:

$$E(x_0, y_0, \Delta_x, \Delta_y) = \sum_x \sum_y \left(I_x(x, y) \cdot \Delta_x + I_y(x, y) \cdot \Delta_y\right)^2 \cdot w(x - x_0, y - y_0)$$

This function can be analyzed using the autocorrelation matrix (also known as the tensor image):

$$A = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

The pixels that are distinguishable correspond to the points where the eigenvalues of A, λ_0 and λ_1, are high. Instead of directly calculating these eigenvalues, Harris and Stephens [1988] use the indicator:

$$g = \det(A) - \alpha \ \mathrm{trace}(A)^2 = \lambda_0 \lambda_1 - \alpha(\lambda_0 + \lambda_1)^2$$

with α=0.06. The pixels where g exceeds a predefined threshold are marked as keypoints. Shi and Tomasi [1994] propose to calculate the eigenvalues and mark as keypoints the pixels where the lowest eigenvalue is a local maximum.

The Difference of Gaussians (DoG) approach detects keypoints based on the changes that produce a Gaussian filter on the image. In order to achieve invariance to scale, the detection process uses a continuous function of scale, known as scale space:

$$L(x, y, \sigma) = G_\sigma(x, y) * I(x, y)$$

The keypoint detection analyzes a sequence of differences of two consecutive scales separated by a factor k:

$$D_j(x, y) = L(x, y, k^{j+1}\sigma) - L(x, y, k^j \sigma)$$

The scale factor k is fixed to a value $2^{1/s}$, thus, there is an integer number s of differences before doubling σ (which is called "an octave"). Given three consecutive difference images D_{j-1}, D_j, and D_{j+1}, the pixel (x, y) is a keypoint when $D_j(x, y)$ is a local maximum or minimum between 27 neighbor pixels: 9 pixels surrounding $D_{j-1}(x, y)$, 9 pixels surrounding $D_j(x, y)$, and 9 pixels surrounding $D_{j+1}(x, y)$. This approach selects the white circular zones surrounded by black, or viceversa, also known as "blobs". These keypoints are complementarity to the ones detected by the autocorrelation approach.

Finally, a keypoint P is determined by up to four values: the spatial location (P_x, P_y), the size of the neighborhood that represents P_σ (given by the level of the scale space where P was detected), and the orientation P_θ (computed from the neighborhood of P in order to provide invariance to rotations).

Keypoint description The descriptor represents the content of the image around a keypoint with a single vector. The descriptors most commonly used are SIFT [Lowe, 2004] and SURF [Bay et al., 2008].

SIFT is a 128-dimensional vector that represents the orientation of the gradient around the keypoint. It uses σ and θ from the keypoint to define the size and the orientation of the neighborhood of the image to represent. Once the neighborhood has been defined, it is divided into 4×4 zones, and for each zone an 8-bins histogram is calculated with the orientations of gradient. Finally, the SIFT descriptor is the concatenation of the 16 histograms, producing the 128-d vector. A common variation for SIFT is PCA-SIFT [Ke and Sukthankar, 2004], which reduces the dimensionality of the descriptor while improving its effectiveness.

SURF is a 64-dimensional vector representing the orientations of gradients. It computes the integral image (table of summed areas) to efficiently find the sum of any rectangular area and estimate the gradients. Bay et al. [2008] show that SURF requires less computational resources than SIFT while achieving almost the same effectiveness.

Finally, the full image is described by the set of local descriptors, each one containing the keypoint data (i.e., spatial location, size, and orientation) along with the descriptor itself (i.e., the vector describing the neighborhood of the keypoint).

Van de Sande et al. [2008] present a comparison of the performance of different variations of SIFT for the object recognition problem.

Matching local descriptors In order to compare images I_1 and I_2, each descriptor in I_1 is compared with every descriptor in I_2 using a distance function δ. Two descriptors $P \in I_1$ and $Q \in I_2$ create a match (denoted by $P \to Q$) when:

1. Q is the closest object in I_2 to P:

$$\forall X \in I_2 - \{Q\},\ \delta(P,Q) \leq \delta(P,X)$$

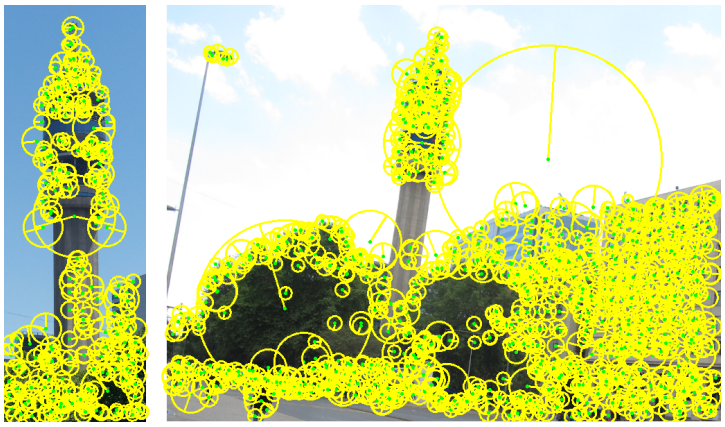
2. The ratio between the distance to Q and the distance to the second closest object in I_2 is lower than s:

$$\forall X \in I_2 - \{Q\},\ \frac{\delta(P,Q)}{\delta(P,X)} \leq s$$

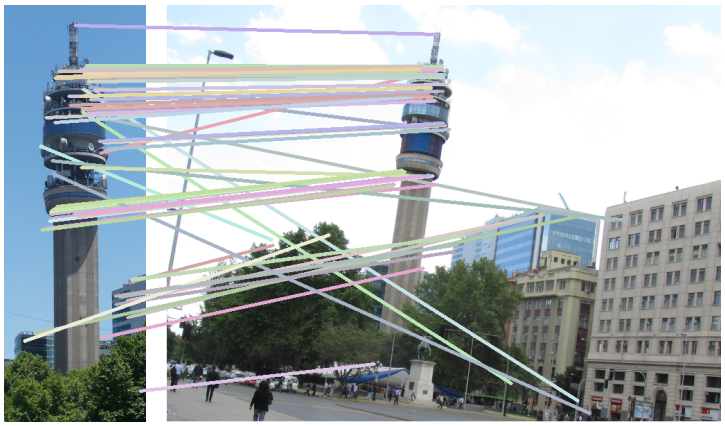
In the case of SIFT descriptors, δ is the Euclidean distance in the 128-dimensional space, and s is commonly fixed to 0.8 [Lowe, 2004]. Figure 2.10b shows matching descriptors between two images following this criterium.

Spatial consistency of matches Given the set of matches C=$\{\{P^1 \to Q^1\}, \{P^2 \to Q^2\}, ...\}$, the spatial consistency determines the best spatial transformation that is satisfied by most of them, i.e., it is an algorithm that computes S and $\mathcal{T}$, where $S \subseteq C$ and $\forall\{P^j \to Q^j\} \in S,\ \mathcal{T}(P^j) = Q^j$. The algorithm must first decide the model of transformation that $\mathcal{T}$ will satisfy, for example:

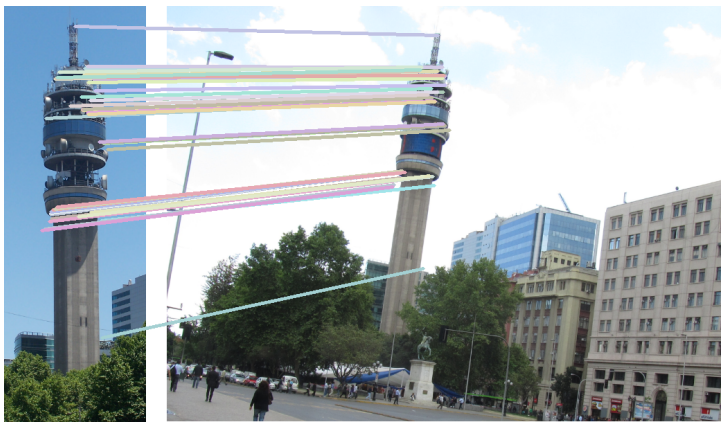
- Translation. $\mathcal{T}$ can only move a set of points in I_1 to match a set of points in I_2, i.e., $\mathcal{T}$



(a) SIFT descriptors for two images. Image on the left has 327 descriptors. Image on the right has 1180 descriptors.



(b) Pairs whose ratio between distance to 1st-NN and 2nd-NN is smaller than s=0.8 (70 pairs).



(c) Subset of pairs satisfying a homographic transformation (45 pairs).

Figure 2.10 – Extracting and matching local descriptors between two images.

corresponds to values (t_x, t_y) which $\forall\{P^j \rightarrow Q^j\} \in S$:

$$\mathcal{T}(P^j) = Q^j \;\Leftrightarrow\; \begin{bmatrix} P_x^j \\ P_y^j \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} Q_x^j \\ Q_y^j \end{bmatrix}$$

- Scale+Translation (ST). $\mathcal{T}$ can scale and move a set of points, i.e., $\mathcal{T}$ corresponds to values $(t_x, t_y, \sigma_x, \sigma_y)$ which $\forall\{P^j \rightarrow Q^j\} \in S$:

$$\mathcal{T}(P^j) = Q^j \;\Leftrightarrow\; \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} P_x^j \\ P_y^j \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} Q_x^j \\ Q_y^j \end{bmatrix}$$

  Sometimes, the scale is restricted to σ_x=σ_y.

- Rotation+Scale+Translation (RST). $\mathcal{T}$ can rotate in the plane, scale and move a set of points, i.e., $\mathcal{T}$ corresponds to values $(t_x, t_y, \sigma_x, \sigma_y, \theta)$ which $\forall\{P^j \rightarrow Q^j\} \in S$:

$$\mathcal{T}(P^j) = Q^j \;\Leftrightarrow\; \begin{bmatrix} \sigma_x & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} P_x^j \\ P_y^j \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} Q_x^j \\ Q_y^j \end{bmatrix}$$

- Affine transformation. $\mathcal{T}$ can apply a linear transformation maintaining the parallelism between lines, including translation, scale, rotation and shear. $\mathcal{T}$ corresponds to values (t_x, t_y, a, b, c, d) which $\forall\{P^j \rightarrow Q^j\} \in S$:

$$\mathcal{T}(P^j) = Q^j \;\Leftrightarrow\; \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} P_x^j \\ P_y^j \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} Q_x^j \\ Q_y^j \end{bmatrix}$$

- Homographic or Projective transformation. $\mathcal{T}$ simulates the change of the point of view of the observer, including Affine transformation and the rotation outside the plane. $\mathcal{T}$ corresponds to values $(a, b, c, d, e, f, g, h, i)$ which $\forall\{P^j \rightarrow Q^j\} \in S$:

$$\mathcal{T}(P^j) = Q^j \;\Leftrightarrow\; \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} P_x^j \\ P_y^j \\ 1 \end{bmatrix} = \begin{bmatrix} wQ_x^j \\ wQ_y^j \\ w \end{bmatrix}$$

Depending on the model, an amount of matches are needed to determine a transformation. For instance, the Translation transformations are determined by one match, the ST transformations by two, the RST and Affine transformations by three, and the Projective transformations are determined by four matches. Once the model of transformation is decided, the algorithms commonly used to determine the best $\mathcal{T}$ and S are RANSAC and Hough Transform (both already introduced in Section 2.2).

In the case of RANSAC, the algorithm follows these steps: a set of seed matches are randomly sampled from C (as many as required by the model of transformation) and a first $\mathcal{T}$ candidate is computed; the set S is created by locating inliers in C for $\mathcal{T}$; and a corrected model $\mathcal{T}$ is computed using the whole set of inliers S. The algorithm is performed many times using different seeds, the $\mathcal{T}$ that collected more inliers in S is selected.

In the case of Hough Transform, a single match is used to fix some variables in $\mathcal{T}$, and the remaining free variables are used to fill the parameter space. In order to decrease the degrees of freedom in $\mathcal{T}$, the scale and orientation in the keypoint can also be used: A single match is able to determine a full RST transformation by using rotation θ=$Q_\theta^i - P_\theta^i$ and scale σ=Q_σ^i / P_σ^i, and therefore $\mathcal{T}$ becomes a single point in the parameter space. The Hough Transform algorithm takes each match in C, computes $\mathcal{T}$, and increments the corresponding accumulator cell in the parameter space. The most voted cell defines $\mathcal{T}$ and the set of voter matches define S.

2.3.3 Bag of Visual Words

The Bag-of-Visual-Words (BOVW) approach, initially described by Sivic and Zisserman [2003], is a global description which summarizes the local descriptors computed from an image. The computation of BOVW descriptors follows three main steps:

1. The local descriptors for the whole collection of images are computed.
2. The "codebook" or "visual vocabulary" is determined. Usually, the codebook is obtained by running the k-means algorithm on a representative sample of local descriptors. The k-means algorithm is chosen because it is fast and can deal with large sets of vectors. Each centroid corresponds to a "visual word" and the set of k centroids corresponds to the codebook.
3. The codebook is used to compute a global descriptor for each image. The most simple approach consists in quantizing each local descriptor to its nearest visual word. Thereafter, the BOW descriptor is computed by following the *tf-idf* weighting [Baeza-Yates and Ribeiro-Neto, 1999]:
   - *tf*: The frequency of occurrence of each visual word in the image.
   - *idf*: Each frequency is weighted by the logarithm of the amount of images in the collection that contains that visual word.

Following this approach, every image in the collection is described by k values corresponding to the relevance of each visual word in the image. Two descriptors are compared using the cosine similarity:

$$sim(\vec{x}, \vec{y}) = \frac{\sum_{i=1}^{k} x_i y_i}{\sqrt{\sum_{i=1}^{k} x_i^2}\sqrt{\sum_{i=1}^{k} y_i^2}}$$

The combination of codebook and cosine similarity enable to achieve high efficiency by using an inverted index to resolve similarity searches (more details are given in Chapter 3). On the other hand, two main issues arise when following this approach: the high computational cost required by the codebook creation, and the loss of information due to vector quantization. During the last years, many techniques have been developed in order to improve the performance of the codebook computation and/or increase the information extracted from the local descriptors in an image.

Van Gemert et al. [2008] address two issues produced by the hard assignment to the nearest visual word: *uncertainty*, defined as the problem produced when many visual words are equally close to the descriptor, and *implausibility*, defined as the problem produced when no visual word is actually close to the descriptor. These problems are overcome by estimating density functions for each visual word, and replacing the hard assignment by a soft assignment based on the distance of the descriptor to each visual word.

The Spatial Pyramids [Lazebnik et al., 2006] focuses on increasing the spatial information stored by BOW descriptors. It partitions the image into increasingly finer regions and computing histograms of local features found inside each region. Another alternative to reduce the loss of information due to quantization consists in increasing the vocabulary size, e.g. Le et al. [2011] use hierarchical k-means to compute a large vocabulary tree with one million leaves and obtain a BOW representation that achieves high effectiveness.

In other work, the Hamming Embedding [Jégou et al., 2008] extends the information in the BOW with a binary signature. The signature stores the relative position of each descriptor with

respect to the associated visual word: for each n-dimensional descriptor the Hamming Embedding is an n-bits sequence, where the i^{th} bit is 1 if the i^{th} dimension of the descriptor was higher than the i^{th} dimension of its visual word.

Similarly, BOSSA [Avila et al., 2011] extends the information in the BOW by including a histogram of distances of the descriptors assigned to each visual word.

The BOW approach has been successfully used in image retrieval [Chum et al., 2007; Jégou et al., 2009], image classification [van Gemert et al., 2008], object localization [Le et al., 2011], and other related problems.

2.4 Summary

In this chapter we have reviewed some techniques of image processing and image analysis that are relevant to understand this thesis.

In the case of image processing, we reviewed different techniques for image filtering and edge detection. In particular, the RANSAC and Hough Transform algorithms enable to fit a model from data samples, which can be used to address the line detection problem and to determine the spatial consistency of local descriptors.

In the case of image analysis, we focused on techniques to describe the content of an image. These techniques can broadly be divided in global description and local description. Once the content has been described, the comparison of any two images relies on the comparison of their descriptors.

The efficient comparison of descriptors is a key problem in CBMIR systems. The next chapter reviews different approaches to address this issue, focusing on a specific approach to compare objects known as Metric Spaces.

Chapter 3

Similarity Search

In this chapter we briefly review different techniques to perform a similarity search. In the context of CBMIR, the similarity search is the algorithm that locates within the database the objects that closely match a given query object. Because databases may be very large, it is essential to use appropriate indexing algorithms to efficiently perform the searches.

Some indexing and search algorithms are designed for specific kinds of descriptors, other more generic algorithms are designed for any descriptor modeled as a vector (vector spaces). There are even more generic algorithms designed for any kind of descriptor while the comparison between descriptors satisfies some minimum properties (metric spaces).

3.1 Concepts

Let $\mathcal{D}$ be the descriptor space (the domain), let $\mathcal{R} \subseteq \mathcal{D}$ be a collection of descriptors (the search space), let $q \in \mathcal{D}$ be the query object (the query), and let $d : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ be a function that compares descriptors (the distance). Typically, d is a dissimilarity function between descriptors, i.e., $d(a, b)$ is a small value (near zero) when two descriptors a and b are similar, and $d(a, b)$ is a high value when they are dissimilar.

The *range search* returns all the objects in $\mathcal{R}$ that are closer than a distance threshold ϵ to q. The *k nearest neighbor search* (k-NN) returns the k closest objects to q in $\mathcal{R}$, and the *NN+range search* returns the intersection between a k-NN search and a range search.

The linear scan or sequential scan is an algorithm that resolves the similarity search by sequentially comparing q to every descriptor in $\mathcal{R}$ and retrieving those that are the closest. The linear scan essentially is a brute-force approach that sets a baseline for high effectiveness and low efficiency.

We should stress that once the query objects, distance function, and search space have been fixed, it is highly improbable to outperform the effectiveness achieved by the linear scan. Hence, in order to increase the effectiveness, the search input must be changed in some semantically meaningful way, like using a different feature extraction and/or using a different distance function.

Indexing and exact search

In order to improve the efficiency, an index structure can be built to reduce the computational effort needed to resolve the search. An *exact search* is any search algorithm that retrieves the same result than the linear scan presumably with higher efficiency.

A system is commonly divided in two phases: the *offline* phase corresponds to the bulk processing of the whole collection of multimedia documents, including the feature extraction and the creation of the index structure, and the *online* phase corresponds to processing queries and resolving similarity searches.

Most of the indexes are designed to be created during the offline phase, that is, a time-expensive process creates the index structure prior to resolving any search. It is expected that the index will resolve many similarity searches, amortizing its creation time, but no information is a priori known about the queries that will be resolved afterwards. In the online phase, the index efficiently receives and resolves any similarity search that may proceed from different sources. All the searches share the same index, and the index should achieve good performance at every search.

The efficiency gain that an index structure can provide is related to: 1) the cost of creating the index during the offline phase; 2) the amount of objects that are discarded during the resolution of similarity searches; 3) the internal cost for deciding whether each object can be discarded or not. A similarity search will be faster than a linear scan when the saved time, due to discarded computation, is greater than the overtime spent due to index creation and internal cost.

Indexes can be classified as static or dynamic depending on how they manage the insertion or deletion of objects in $\mathcal{R}$ during the online phase. A dynamic index can efficiently update its structure to add or remove any object while it is in the online phase, hence it can resolve searches even for growing collections. A static index cannot manage large updates in its structures, therefore after many modifications of $\mathcal{R}$ the whole indexing structure must be rebuilt.

Approximate search

An *approximate search* is an algorithm that admits differences in the answer compared to an exact search. The possibility of returning incorrect nearest neighbors permits greater gains in efficiency at the cost of decreasing effectiveness. However, it should be noted that the loss in effectiveness due to search approximation can even be negligible when compared to the inherent subjectiveness of content similarity and imprecision of content description.

Zezula et al. [2005] classify approximate search approaches into two broad categories:

1. Reduction of the data to be examined, analyzing less data than is technically needed.
2. Transformation of the search space, replacing either the objects or the distance to reduce the search cost.

A more general classification, presented by Patella and Ciaccia [2009], identifies four characteristics from approximate searches and classifies many known algorithms according to them. These characteristics are:

1. Type of space the approach applies to: for instance, vector spaces or metric spaces.
2. Approximation type: for instance, transformation of the search space (to modify either the distance or the objects) or reduction of the distance evaluations (by smart pruning of less promising regions or by just an early termination of the search).
3. Quality guarantees: for instance, deterministic guarantees, probabilistic guarantees, or no guarantees at all.
4. User interaction: for instance, static or interactive.

In Chapter 9 we present an approximate search algorithm that applies to metric spaces. The approximation consists in replacing a time-expensive distance by a fast estimator. Then, it performs an exact search using the original distance only for the objects with most promising estimations.

3.2 Vector Spaces

A *vector* is an n-tuple of real numbers $\vec{x} = (x_1, ..., x_n)$, each individual number x_i being referred to as *coordinates* or *components*. A vector space is a mathematical structure formed by n-dimensional vectors and two operations: vector addition and scalar multiplication. These operations satisfy properties including associativity, commutativity, distributivity, identity element for addition, inverse element, and identity element for multiplication.

The length of a vector can be measured by defining a *norm* (notation $||\vec{v}||$). The angle between any two vectors can be measured by defining a *dot product* (notation $\langle \vec{x}, \vec{y} \rangle$). A common definition for these functions is given by the Euclidean space (also known as L^2-space):

$$||\vec{x}|| = \sqrt{\sum_{i=1}^{n} x_i^2}$$

$$\langle \vec{x}, \vec{y} \rangle = \cos(\angle(\vec{x}, \vec{y})) \cdot ||\vec{x}|| \cdot ||\vec{y}|| = \sum_{i=1}^{n} x_i y_i$$

A common extension to these definitions is given by the family of vector spaces L^p, which are based on the p-norm:

$$||\vec{x}||_p = \sqrt[p]{\sum_{i=1}^{n} x_i^p} \quad \text{with } p \geq 1$$

The descriptors computed by feature extraction methods commonly correspond to high-dimensional vectors, therefore the use of indexes for vector spaces appear as a natural alternative for improving the efficiency of similarity searches. In the following we review some indexes using data organization, space division, and random projections multidimensional hashing.

3.2.1 R-tree

The R-tree [Guttman, 1984] is a balanced tree that resembles the well-known B-Tree. The internal nodes store minimum bounding rectangles (MBR) and tree leaves store groups of data

vectors. It is a hierarchical organization of data vectors that dynamically supports insertions and deletes. When a new vector is added to a previously-full node, the node is split and the overflow is iteratively added to the parents following a backtracking algorithm similar to B-trees. Different criteria can be used to decide the node splitting, the author recommends an heuristic to minimize the total volume of the resulting division. The search algorithm uses a depth-first traversal where a branch is pruned if the MBR does not overlap the query ball. The R-tree was designed for spatial data (i.e., 2-d and 3-d vectors), but can also be used with high dimensional vectors. However, in high dimensional spaces it is highly probable that most of the MBRs overlap, in that case, the R-Tree will not provide any efficiency gain compared to a linear scan.

Hjaltason and Samet [1995] propose a search algorithm that does not use a recursive search in spatial indexes. Instead, a priority queue sorts the tree nodes according to their distance from the query object. The search adds the tree root to the queue and then iteratively removes and examines the node in the head of the queue. If the node is a leaf, its vectors are examined and the relevant vectors are reported. If the node is not a leaf, its child nodes are added to the queue (and sorted according to their distance to the query). Böhm et al. [2001] demonstrate the optimality of this algorithm to retrieve the nearest neighbor in hierarchical spatial indexes.

3.2.2 Kd-tree

The kd-tree [Friedman et al., 1977] is a binary tree which is created by recursively splitting the dataset according to a dimension i and a coordinate value m. The splitting algorithm analyzes the coordinate values for the subset, and selects as the splitting dimension i the one whose coordinates show the highest variance, and as the splitting value m the median value for coordinate values. A node is created containing i, m and one child nodes for each half. Each node is recursively split, producing a balanced binary tree with height $\lceil \log n \rceil$.

The original search algorithm for the kd-tree uses a depth-first recursive search that first locates the node that would contain the query object and compares all the elements in that node. Then, it compares the neighbor nodes by backtracking the recursion and discarding nodes that do not intersect with the query ball.

Beis and Lowe [1997] propose an approximate search for the kd-tree called BBF search (Best Bin First), which uses a priority queue similar to Hjaltason and Samet [1995] search algorithm. The approximation is incorporated by stopping the search when the number of examined nodes has exceeded some user-defined threshold.

Muja and Lowe [2009] propose to compute several independent kd-trees using the same dataset, each tree selecting a random dimension between the top D dimensions with greater variance. The search algorithm traverses all the randomized kd-trees at the same time by examining nodes using a unique priority queue. The search starts by adding all the roots to the priority queue, iteratively examines the node in the head of the queue and adds its child nodes to the queue, and stops the search according to the number of nodes that has been examined.

3.2.3 K-means tree

The k-means is a widely used clustering algorithm for vector spaces. It is based on selecting k seed vectors (either by random selection or using some heuristic), and iteratively corrected them to

minimize the sum of the squared error [Tan et al., 2009]. The k centroids induce a Voronoi diagram that partitions the space into k cells, hence the search algorithm locates the cells that overlap the query ball, and examines the objects assigned to them.

The hierarchical k-means is a k-ary tree which is created by recursively computing k centroids and subdividing the dataset. The recursion ends when a cell contains no more than k data vectors. This produces a tree with height $\lceil \log_k n \rceil$. Muja and Lowe [2009] adapted the approximate search for kd-tree to hierarchical k-means tree by using a priority queue that sorts nodes according to the distance from the their centroid to the query.

The inverted index is frequently used to improve the efficiency when a codebook has been computed (see Section 2.3.3). The inverted index is a table with one entry for each visual word, each entry contains a list with all the images that contain some descriptor assigned to that visual word (i.e., the vectors containing a greater-than-zero weight for that dimension). The similarity search uses the inverted index to locate all the images that share one or more visual words with the query image in "immediate run-time" [Sivic and Zisserman, 2003].

3.2.4 LSH

Locality-Sensitive Hashing (LSH) is a randomize algorithm which uses several hash functions to perform approximate searches [Andoni and Indyk, 2008]. It converts a multidimensional vector into a sequence of n bits, and indexes them in a hash table. The Manhattan distance between objects can be approximated by the Hamming distance between the corresponding binary strings. LSH is designed to perform approximate searches. The effective-versus-efficiency tradeoff is controlled by the number n of bits used to quantize vectors. A query vector is processed by the hash functions and all the vectors that are assigned to the same bin are selected as candidates. The final result is created by performing a linear search through candidate vectors.

3.3 Metric Spaces

A *metric space* is defined by the pair $(\mathcal{D}, d)$, where d satisfies the following properties:

$$\begin{array}{lll} \text{reflexivity} & \forall x \in \mathcal{D}, & d(x,x) = 0 \\ \text{non-negativity} & \forall x, y \in \mathcal{D}, & d(x,y) \geq 0 \\ \text{symmetry} & \forall x, y \in \mathcal{D}, & d(x,y) = d(y,x) \\ \text{triangle inequality} & \forall x, y, z \in \mathcal{D}, & d(x,z) \leq d(x,y) + d(y,z) \end{array}$$

For comparing descriptors, the Minkowski distances ($\mathsf{L_p}$) are an example of widely used metrics. For a n-dimensional space, the $\mathsf{L_p}$ metric is defined as:

$$\mathsf{L_p}(\vec{x}, \vec{y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}} \qquad \text{with } p \geq 1$$

In general, the metric properties represent a tradeoff between efficiency and effectiveness for similarity searches. On one hand, the metric properties enable the use of well studied index

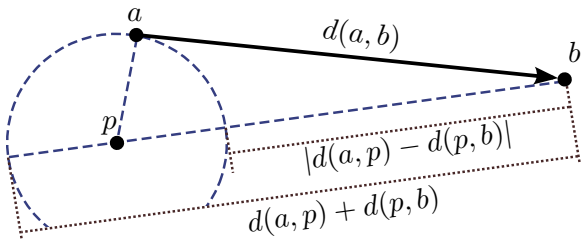


Figure 3.1 – The object-pivot distance constraint for distance $d(a, b)$ using the pivot object p.

structures, accelerating searches by discarding groups of objects (as we review below). On the other hand, the metric properties restrict the similarity model that can be used for comparing two objects [Skopal, 2007; Skopal and Bustos, 2011].

3.3.1 Efficiency in Metric Spaces

In order to improve efficiency in metric spaces, the *Metric Access Methods* (MAMs) [Chávez et al., 2001] are index structures designed to efficiently perform similarity search queries. MAMs avoid a linear scan over the whole database by using the metric properties to save distance computations, usually at the cost of storing some previously computed distances. Given the metric space $(\mathcal{D}, d)$, the object-pivot distance constraint [Zezula et al., 2005] guarantees that:

$$\forall a, b, p \in \mathcal{D}, \quad |d(a, p) - d(b, p)| \leq d(a, b) \leq d(a, p) + d(b, p) \tag{3.1}$$

This constraint implies that for any two objects a and b, a lower bound and an upper bound for $d(a, b)$ can be calculated using a third object p, which is called a *pivot* (see Figure 3.1). If $d(a, p)$ and $d(b, p)$ are precalculated, then these bounds can be efficiently computed.

MAMs resolve similarity searches by grouping objects in the database according to some criteria. Then Equation 3.1 is used to discard groups of objects, thus saving distance computations and search time. MAMs differ in their methods for grouping objects and for selecting pivots.

The GNAT [Brin, 1995] is a n-ary tree where that recursively subdivides a dataset according to n split objects. It is analogous to the hierarchical k-means tree from vector spaces, with the difference that the split objects belong to the dataset. In fact, the split objects can be selected at random, but the author recommends to select objects that are far away from each other. The search algorithm uses the split point as a pivot to discard tree branches.

The M-tree [Ciaccia et al., 1997] is a balanced tree where each node stores a representative object and a covering radius, like the GNAT. The M-tree is a dynamic structure that manages object inserts and deletes in a similar manner as the R-tree: every new object is added to a leaf, the full nodes are divided in two groups, and the overflow is recursively added to the parent nodes up to the root.

Pivot Tables

Given a collection of objects $\mathcal{R}$ and a set of pivot objects $\mathcal{P}$ (which may or may not be a subset of $\mathcal{R}$), a pivot table is a $|\mathcal{R}|\times|\mathcal{P}|$ matrix that stores the distance from each pivot to every

object in the collection. The similarity search for a query object q (not necessarily in $\mathcal{R}$) evaluates the distance $d(q,p)$ for each pivot $p \in \mathcal{P}$, and then sequentially scans each object $r \in \mathcal{R}$, calculating a lower bound for $d(q,r)$:

$$\max_{p \in \mathcal{P}} \{|d(q,p) - d(r,p)|\} \leq d(q,r)$$

This lower bound can be evaluated efficiently because $d(q,p)$ is already calculated and $d(r,p)$ is stored in the pivot table. In the case of range searches, if the lower bound is greater than ϵ then r can be safely discarded because r cannot be part of the search result. In the case of k-NN searches, if the lower bound is greater than the current k^{th} nearest neighbor candidate, then r can be safely discarded. If r could not be discarded, the actual distance $d(q,r)$ must be evaluated to decide whether or not r is part of the search result.

The Approximating and Eliminating Search Algorithm (AESA) [Vidal, 1994] computes the distance between every pair of objects in $\mathcal{R}$ and stores them in a $|\mathcal{R}|\times|\mathcal{R}|$ matrix, i.e., $\mathcal{P}$ is the whole set $\mathcal{R}$. Actually, due to the metric properties, only $(|\mathcal{R}|-1)\cdot(|\mathcal{R}|-2)$ evaluations of d are needed. AESA can resolve nearest neighbor searches with an average constant number of distance computations, at the cost of requiring quadratic space for storing the pivot table. The Linear AESA (LAESA) [Micó et al., 1994] overcomes this issue by selecting a set of pivots $\mathcal{P} \subseteq \mathcal{R}$. LAESA can reduce the space for storing the pivot table compared to AESA, however it requires an algorithm for selecting a good set of pivots.

One approach for selecting pivots is to randomly select objects in $\mathcal{R}$. However, as can be inferred from Figure 3.1, a good pivot should be either close to the query or to the object in $\mathcal{R}$. Additionally, a key property for selecting good sets of pivots is that each pivot in the set should be also far away from each other [Zezula et al., 2005].

The Sparse Spatial Selection (SSS) [Bustos et al., 2008] is an algorithm that selects pivots far away from each other: given a distance threshold t, it traverses $\mathcal{R}$ and it chooses an object $x \in \mathcal{R}$ to be added to $\mathcal{P}$ when its distance to each previously selected pivot is higher than t. If many candidate sets of pivots have been selected, either by random selection or by the SSS algorithm, they must be evaluated and compared. The evaluation consists of calculating $\mu_{\mathcal{P}}$, which is the average value of lower bounds [Bustos et al., 2003]. The set of pivots $\mathcal{P}$ with higher $\mu_{\mathcal{P}}$ is finally selected while the other sets are discarded. The evaluation prefers sets with high lower bounds, because they will probably discard more distances in the search.

LAESA can manage the insertion or deletion of objects and pivots by adding or removing rows or columns from the pivot table [Micó and Oncina, 2010]. However, LAESA is mainly a static index because the actual implementation of the pivot table may not support dynamic updates. In that case, a new table is created and the old table is discarded. Also, after many modifications in $\mathcal{R}$ the set of pivots can begin to perform poorly and a new set of pivots should be selected.

Intrinsic Dimensionality

In order to analyze the efficiency that any MAM can achieve in a metric space $(\mathcal{R}, d)$ with $\mathcal{R} \subseteq \mathcal{D}$, Chávez et al. [2001] propose to analyze the histogram of distances of d. A histogram of distances is constructed by evaluating $d(a,b)$ for a random sample of objects $a, b \in \mathcal{R}$. The histogram of distances reveals information about the distribution of objects in the collection, see Figure 3.2.

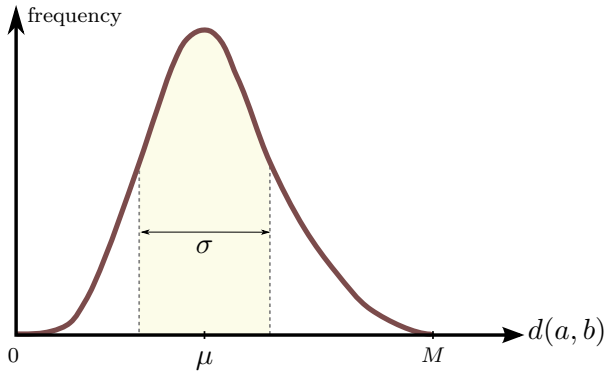


Figure 3.2 – Histogram of distances with median μ, variance σ^2, and maximum distance M for some metric space $(\mathcal{R}, d)$.

Given a histogram of distances of a metric space, the *intrinsic dimensionality* is defined as [Chávez et al., 2001]:

$$\rho = \frac{\mu^2}{2\sigma^2} \ , \tag{3.2}$$

where μ and σ^2 are the mean and the variance of the histogram of distances.

The intrinsic dimensionality estimates the efficiency that any MAM can achieve in $(\mathcal{R}, d)$, therefore it tries to quantify the difficulty in indexing a metric space. A histogram of distances with small variance (i.e., a high value of ρ) means that the distance between any two objects $d(a, b)$ with high probability will be near μ, thus the difference between any two distances with high probability will be a small value. In that case, for most of the pivots the lower bound from Equation 3.1 will probably become useless at discarding objects.

In the case of LAESA, the internal cost for the similarity search comprises: the evaluation of the distance between the query object and each pivot at the beginning of the search, and the evaluation of the lower bound between the query object and each object in the collection. Increasing the number of pivots will improve the value of the lower bounds, thus more objects may be discarded. However, the cost for evaluating the lower bounds increases linearly with the number of pivots. Hence, the number of pivots presents a tradeoff between the amount of distances discarded and the cost of evaluating each lower bound. The optimal solution for this tradeoff mainly depends on the intrinsic dimensionality of the search space and quality of the set of pivots.

Indexing the query set

In some domains the query objects may have some special properties that can be exploited to improve the performance of the index. In particular, content-based video retrieval systems usually extract many keyframes from a query video, and similarity searches are performed for those keyframes. Because keyframes proceed from the frame video, it may be expected that two consecutive query objects will frequently be similar. In interactive content-based systems, the user starts a search with a text or an example, the system performs a k-NN search and the results are presented to the user. The user iteratively selects new query objects between those presented objects, and a new search is performed to refine the results. Because the new queries are selected from the answers of a previous search, it may be expected that two consecutive query objects will be similar.

The D-file [Skopal et al., 2012] is a dynamic MAM that is created online while processing a stream of queries. The D-file is the database file which is accompanied by a main-memory structure, called the D-cache. The D-cache stores the evaluated distances $d(q_i, o_j)$ while processing the queries in the stream. When the n^{th} query in the stream is processed, the D-cache calculates a lower-bound distance for $d(q_n, o_j)$ evaluating the distance from q_n to previous q_i, thus treating previous queries as pivots. D-cache content is modeled as a sparse dynamic pivot table which stores all the evaluated distances. If some distances have been discarded, then some rows may be incomplete. Using the stored distances, the D-cache tries to discard objects using the same approach as pivot tables. Because the D-cache is built during query processing, the D-file does not need an offline indexing step. As the D-cache uses the previously processed queries as dynamic pivots, the authors recommend that previous queries should be as close to the current query as possible.

The D-cache is implemented with: 1) a fixed-size hash table that stores triplets $(q_i, o_j, d(q_i, o_j))$; 2) a hash function $h(q_i, o_j)$ for accessing the bucket where a triplet is stored; 3) a collision interval, for searching a near available bucket when some triplet is mapped into an already used bucket; and 4) a replacement policy, that decides whether or not a new triplet should replace an old triplet when a collision occurs and there is not an available bucket in the collision interval.

In Chapter 9 we analyze the performance of the D-cache, and we show it suffers from high internal complexity. This thesis proposes a new index structure called the Snake Table, which preserves these ideas of dynamic pivots and achieves high performance.

3.3.2 Effectiveness in Metric Spaces

In this section we review two approaches to modify the distance function in order to improve the effectiveness of the search: defining a dissimilarity function as a linear combination of metrics, and raising the restrictions of a metric.

Multi-Metric Approach

Let $\{g_1, ..., g_m\}$ be a set of a feature extraction methods where $g_i : \mathcal{D} \to \mathcal{F}_i$ extracts a global descriptor, let $\{d_1, ..., d_m\}$ be a set of distance functions where $d_i : \mathcal{F}_i \times \mathcal{F}_i \to \mathbb{R}$ is a dissimilarity function that defines the metric space $(\mathcal{F}_i, d_i)$, and let $\{w_1, ..., w_m\}$ be the set of weights $w_i \in \mathbb{R}$, then a multi-metric space $(\mathcal{D}, \gamma)$ is defined where $\gamma : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ calculates the dissimilarity between two objects as:

$$\forall x, y \in \mathcal{D}, \quad \gamma(x, y) = \sum_{i=1}^{m} w_i \cdot d_i(g_i(x), g_i(y))$$

We will call the set $\{d_1, ...d_m\}$ as the underlying metrics of γ. In this thesis, we will focus on convex combinations, i.e., $w_i \in [0, 1]$ and $\sum_{i=1}^{m} w_i$=1.

Bustos and Skopal [2006] proposed a dynamic weighting of metrics, called *entropy impurity*, where each weight changes depending on the query object. This method computes the set of weights prior to each similarity search by analyzing the result of a search on a database already classified. A similar technique is also proposed by Deselaers et al. [2007] under the name of *maximum entropy*. The major problem with dynamic weighting is that it breaks the metric properties, thus general MAMs cannot be used.

If weights are static (i.e., a fixed value for all searches), then γ also satisfies the metric properties [Batko et al., 2009; Bustos, 2006], thus any MAM can be used for indexing the objects. The value assigned to each weight depends on the actual implementation of descriptors (a different weight is required depending on whether a descriptor represents colors or textures), and application specifics (a system that retrieves sport images may use different weights than a system that retrieves hand-made sketches). The set of weights should be fixed subjectively as fine tuning parameters [Batko et al., 2009], or should be fixed in accordance with evaluations of effectiveness. However, the evaluation of the effectiveness of a system is usually difficult because it requires the definition of proper methodology and indicators, and in some cases it requires to hire people to use the system and fill out evaluation forms.

Non-Metric Approach

Another way for improving the effectiveness is to raise the restrictions of a metric. When a distance function does not satisfy some of the metric properties it is known as a *non-metric* [Skopal and Bustos, 2011]. In particular, a *semi-metric* is a non-metric that satisfies reflexivity, non-negativity, and symmetry, but not the triangle inequality.

Even using the more advanced feature extraction method, if the distance function does not correctly model the human notion of similarity, the results will not be satisfactory. Moreover, the effectiveness of the similarity search can only be improved by researching better algorithms, as opposed to efficiency that may be improved by better hardware. Non-metric distances can be useful for creating complex similarity measures that represent more accurately the human notion of similarity without the constraints of the metric properties. However, in this case general MAMs cannot be used.

The *Dynamic Partial Functions* [Li et al., 2002] are a family of semi-metric distances used with remarkable results. It is based on the $\mathsf{L_p}$ distances, but only taking into consideration the subset of the m dimensions with the smallest differences. Let $\vec{x}$ and $\vec{y}$ be two n-dimensional vectors, and $m \in \{1, ..., d\}$ be a parameter, then the DPF distance is defined as:

$$\mathsf{DPF}(\vec{x}, \vec{y}) = \left(\sum_{c_i \in \Delta_m} {c_i}^p \right)^{\frac{1}{p}} \quad \text{with } p \geq 1 \ ,$$

where $c_i = |x_i - y_i|$ is the difference of $\vec{x}$ and $\vec{y}$ in the i-th coordinate, and Δ_m is the subset of the m smallest values of $\{c_1, ..., c_n\}$. Meng et al. [2003] present some criteria for selecting the value of m and make a comparison with Minkowski distances for image copy detection. They showed that for a good selection of m, DPF outperforms the effectiveness of $\mathsf{L_p}$ for retrieving similar images represented by global descriptors.

In other study, Aggarwal et al. [2001] show that the *Fractional distances*, the $\mathsf{L_p}$ distances with $0 < p < 1$, have better results than $\mathsf{L_2}$ for clustering high dimensional spaces. The experiments were performed using the k-means algorithm and showed that for a small p (in particular they tested $\mathsf{L_{0.1}}$) the rate between the minimum and maximum values diverges, thus increasing the discriminability between points. Afterwards, Howarth and Rüger [2005] compared Fractional distances against $\mathsf{L_1}$ and $\mathsf{L_2}$ in an image retrieval system, where each image was represented by a global descriptor. They empirically showed that Fractional distances outperform $\mathsf{L_1}$ and $\mathsf{L_2}$.

Rubner et al. [2001] made an empirical comparison of different metrics and non-metrics for

image retrieval. Their experiments showed that with larger sample sizes the non-metric functions outperformed metrics for global color and texture descriptors. In particular, we highlight the performance of the *Chi-squared test statistic*, which achieves a high performance, and it is fast to evaluate. Let $\vec{x}$ and $\vec{y}$ be two n-dimensional vectors, the χ^2 distance is defined as:

$$\chi^2(\vec{x},\vec{y}) = \sum_{i=1}^{n} \frac{(x_i - \bar{m}_i)^2}{\bar{m}_i} \quad , \quad \text{where } \bar{m}_i = \frac{x_i + y_i}{2} \quad .$$

There are few works where non-metric measures have been specifically proposed for content-based video copy detection. The global descriptor FRAS used by Shen et al. [2007] is compared with the Probability-based Edit Distance which is a non-metric. This algorithm is used for short video clips, due to the quadratic time complexity for computing the edit distance.

Cheung and Zakhor [2003] used a L_1 distance for comparing histograms for a video retrieval system and defined a modified L_1 distance by removing a dominant color of each histogram. The modified distance does not comply with the triangle inequality. They defined the modified distance to avoid selecting videos whose background were identical even when the videos were different (the case of videos with slides). The modified distance were used in a postprocessing filtering step for discarding non-relevant objects selected by the metric.

Regarding the efficiency for non-metrics, a direct approach to resolve searches is to directly use MAM and perform exact searches. Because the MAM indexes a non-metric the exact search becomes an approximate search. This approach has the main drawback that it cannot control or even know the amount of approximation in the result. Therefore, some techniques specifically designed for non-metric indexing are needed.

The specific research for non-metric indexing can be broadly classified in two approaches: 1) designing index structures for specific non-metrics; and 2) converting semi-metrics into metrics by correcting the triangle inequality.

In the first case, Aggarwal and Yu [2000] propose an index structure for high dimensional spaces called IGrid. It is inspired in the inverted file, and it intends to index a weighted Minkowski distance computed on multidimensional vectors. The data vectors are organized in ranges for each dimension, and the algorithm selects and retrieves the objects in relevant ranges for the search.

In other work, Goh et al. [2002] propose DynDex, which groups descriptors into clusters in order to discard groups during the search. The clustering algorithm is CLARANS [Ng and Han, 2002] because it compares pairs of in order to compute medoids instead of centroids, thus not requiring the metric properties. The similarity search retrieves the nearest clusters to the query object and performs a linear scan between relevant clusters.

Given a semi-metric δ, a lower bound metric function d (i.e. $\forall x, y\, d(x,y) \leq \delta(x,y)$) can be used for efficiently filtering irrelevant objects. The objects that cannot be discarded by d are compared by δ. Ciaccia and Patella [2002] propose the QIC-M-tree which is an extension of the M-tree that follows this approach. The metric d should be as tight as possible to δ while satisfying triangle inequality for obtaining some efficiency gain. Unfortunately, the definition of d depends on the topological properties of δ, thus it needs specific studies for every semi-metric.

In the more general approach, a semi-metric δ can produce a metric by defining $d(x,y) = \delta(x,y) + c$ with a large enough constant c for satisfying the triangle inequality [Roth et al., 2002]. Even though the triangle inequality is corrected, the intrinsic dimensionality of the new metric space

will be so high that it will ruin any further indexing.

Following this idea, TriGen [Skopal, 2006, 2007] replaces the constant c by concave and convex functions. Using these kind of functions, the amount of triplets satisfying the triangle inequality can be controlled. Skopal and Lokoč [2008] propose NM-Tree, which is an extension of the M-Tree that performs exact and approximate searches on non-metrics. The index controls the effectiveness and efficiency of the search by adjusting the level of correction of the non-metric and the level of approximation of the search.

An alternative approach is Local Constant Embedding Chen and Lian [2008], which partitions the database into multiple groups and computes many different constants c_i for satisfying triangle inequality locally inside each group. The search uses representative objects from each group in order to select the most promising groups to scan in. The main drawback of this technique is the algorithm for generating groups and constants c_i has cubic time complexity.

3.4 Vector spaces versus Metric spaces

Vector spaces and metric spaces are two approaches to resolve similarity searches. A discussion about the benefits and drawbacks for vector spaces and metric spaces are given in the surveys Böhm et al. [2001] and Chávez et al. [2001].

For instance, let $\mathcal{I}$ be the image domain, let $g : \mathcal{I} \to \mathcal{F}$ be a feature extraction method that calculates a global descriptor $g(o)$ for the image o, and let $d_g : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$ be a distance function between global descriptors. Commonly, a vector space is defined for $\mathcal{F}$ and d_g is an L^p distance, hence the distance between two images corresponds to the distance between their global descriptors. A metric space makes no assumptions about a domain as long as the distance satisfies the metric properties, e.g. a metric space for images can be defined using a distance $d : \mathcal{I} \times \mathcal{I} \to \mathbb{R}$, which compare a single global descriptor ($d(x, y) = d_g(g(x), g(y))$), combine global descriptors in a multi-metric, or even compare local descriptors.

On one hand, metric spaces do not have coordinate system nor dimensions, thus they must rely on some rudimentary geometry (based on closeness between objects) to save distance computations. This issue implies that indexes for metric spaces achieve poorer pruning performance (i.e., are less efficient) than indexes for vector spaces [Böhm et al., 2001]. However, we should note this is not always true since some metric indexes can show similar or higher performance than some vector indexes (e.g., M-tree versus R*-tree Ciaccia et al. [1997]).

On the other hand, metric spaces can be applied to any kind of object, whereas the distance function knows how to compare them. It is even possible to define specific functions that compare two images directly without extracting feature vectors at all. In fact, a complex distance function, like the solution of an optimization problem, can achieve higher effectiveness than a simpler distance function [Rubner et al., 2000]. However, we should note that in some scenarios the extra computation time is unaffordable, or the increase in effectiveness is not worth the efficiency cost.

Finally, there are some techniques that map a metric space into a vector space [Faloutsos and Lin, 1995]. The idea of these techniques is to convert a search in a metric space into an approximate search in a vector space. The performance of these techniques highly depends on the characteristics of the involved data and mapped metric space, thus it is not possible to evaluate them in general [Chávez et al., 2001].

In summary, the selection between vector spaces and metric spaces is loosely related to the effectiveness-versus-efficiency tradeoff (described in Chapter 1). The main advantage of vector spaces is the higher efficiency they can achieve. The main advantage of metric spaces is the (potentially) higher effectiveness they may achieve.

3.5 Summary

In this chapter we briefly reviewed the similarity search topic. In particular, we discussed two approaches to resolve similarity searches: vector spaces and metric spaces. The first approach arises naturally when a feature extraction method is used to extract descriptors from multimedia documents. The second approach is a generalization which has as a main advantage the ability to use complex distances to compare objects, at the cost of lower search efficiency compared to vector spaces.

In this thesis, we chose the metric space approach mainly because there is a large unexplored field in the application of metric spaces to the video domain. As we review in next chapter, currently the top-performing CBVCD systems use vector spaces to model similarity, either in the form of tree-based indexes, hash-based tables, or codebooks produced by k-means. To the best of our knowledge, the metric space approach has not successfully been applied to video domains, mainly because the high volume of data forces researchers to choose an approach with higher efficiency. Hence, this research intends to show a valid approach to apply metric spaces in the CBVCD topic, that can achieve competitive performance with state-of-the-art systems.

Because video datasets are usually large, a key aspect of this thesis will be the balance between complexity of the model and the computational effort required to use it. Indeed, in this thesis we tested different similarity functions and we analyze the performance of linear combination of metrics. The efficiency in this thesis is based on pivot tables due to the low internal complexity, and we developed some techniques that profit from video domain properties to reduce the search cost.

The next chapter reviews the related work specific to the content-based video copy detection topic.

Chapter 4

Related Work

In this chapter, we briefly review the related work for the content-based video copy detection topic. First, we discuss different definitions and formalizations for the video copy detection problem. Afterwards, we summarize some applications of video copy detection techniques. Then, we review the related work following four main aspects, namely Content Description, Similarity Search, Temporal Consistency and Multimodal Fusion. Finally, we discuss a complementary approach to content-based video copy detection, called watermarking.

4.1 Definition of Content-Based Video Copy Detection

In this section we analyze and formalize the term *Content-Based Video Copy Detection* (CBVCD). Other terms like *content-based copy retrieval* [Joly et al., 2007], *near-duplicate video detection* [Wu et al., 2007; Shen et al., 2007], *partial near-duplicate video detection* [Tan et al., 2009], *user-centric near-duplicate detection* [Cherubini et al., 2009], *detection of duplicated sequences* [Naturel and Gros, 2005], *video matching* [Basharat et al., 2008], and *video sequence identification* [Iwamoto et al., 2006] are also used for a relatively similar or identical purpose. This section sets out to unify these terms.

A broad definition of Content-Based Copy Detection (CBCD) is the following:

Definition 1 (CBCD) *Let $\mathcal{V}$ be a set of multimedia documents (image, audio, video), and $\mathcal{C}$ be a set of query documents, the CBCD consists in retrieving for every $q \in \mathcal{C}$, all the multimedia documents in $\mathcal{V}$ from which q "is a copy".*

It should be noted that the objective of CBCD is to retrieve multimedia documents instead of just detecting whether or not a query document is a copy. The set $\mathcal{V}$ is known as the "reference collection", and $\mathcal{C}$ is the "query collection".

The specific behavior of a CBVCD system depends on the definition of copy. Depending on the specific objectives of a system, several definitions have been given for what a copy is. In general, we differentiate two approaches for defining a copy: content transformations and semantic similarity.

4.1.1 CBVCD based on content transformations

The approach based on content transformations is mainly related to the copyright infringement problem. Given a collection of original multimedia documents, a copy is any derivation of an original document independent of its semantic meaning. A "content transformation" is the function that produces a derived document by means of adding or removing visual or acoustic information of a source document. In particular, the content transformations that are relevant for CBCD are described by the subjective notion of tolerated transformation [Joly et al., 2007]:

Definition 2 (Tolerated transformation) *Given an original multimedia document, a tolerated transformation is a content transformation that creates a new multimedia document where the original document "remains recognizable".*

The content transformations frequently used in CBVCD can be broadly classified into three categories:

- **Quality transformations**. These transformations usually affect the whole document and sometimes they are unintended consequences of the capture and/or processing of the video. Some examples are: blur, noise, change of brightness or contrast, artifacts due to re-encoding, etc.
- **Postproduction transformations**. These transformations may affect the whole video or some part of it. They are usually produced in the edition process with the purpose of embedding or removing information. Some examples are: insertion of text or logos, cropping, picture in picture, chroma key composition, acoustic effects, and special effects in general.
- **Mash-up transformations**. These transformations affect the length of a video. They are usually produced in the edition process by adding or removing video excerpts. Some examples are: inclusion of opening/endings, embedding into a longer sequence, removal of excerpts, etc.

Additionally, each transformation has an associated intensity. A **low-intensity** transformation slightly modifies the content, hence the transformed video is essentially identical to the original. A **strong-intensity** transformation modifies the document in a way that it is hard to identify the original. Figure 4.1 shows some examples of content transformations. The transformations may affect the visual content and/or the audio content of a video.

Finally, the definition of copy is based on the use of tolerated transformations:

Definition 3 (Copy) *Let $\mathcal{T}$ be a set of tolerated transformations, and u and v be two multimedia documents, v* is a copy of *u if $\exists t \in \mathcal{T},\ t(u) = v$.*

Low-intensity transformations can normally be composed several times, i.e., if t_1 and $t_2 \in \mathcal{T}$, it is common that $t_1 \circ t_2 \in \mathcal{T}$. Thus, given an original document, many copies could be produced by successively applying different transformations, creating a sort of "tree of copies" [Joly et al., 2007]. However, when mash-up transformations or some strong-intensity transformations are used, the relation of copy may become non transitive.

Figure 4.1 – Some examples of content transformations used in the TRECVID datasets.

The tolerated transformations that produce the copies may be a priori known or may be unknown. In the former case, the CBVCD system is able to introduce some special treatment to query videos like detection and reversion of transformations. In the latter, the CBVCD system should be general enough to be robust to any kind of tolerated transformation.

As stated at the start of this section, there are many terms to denominate the copies that a CBVCD system should detect. The following definitions try to unify them with the definition of copy.

Definition 4 (Duplicate) v is a duplicate of u *if* v *is a copy of* u *when* $\mathcal{T}$ *contains only low-intensity quality transformations.*

Definition 5 (Near-Duplicate) v is a near-duplicate of u *if* v *is a copy of* u *when* $\mathcal{T}$ *contains low-intensity quality and postproduction transformations.*

Definition 6 (Partial Near-Duplicate) v is a partial near-duplicate of u *if* v *is a copy of* u *when* $\mathcal{T}$ *contains low-intensity quality, postproduction and mash-up transformations.*

4.1.2 CBVCD based on semantic similarity

The approach based on semantic similarity is mainly related to the problem of reducing the redundancy of search results in video-sharing web sites.

One kind of semantic similarity occurs when two clips present the same scene, but from a different angle and capturing parameters. This issue frequently occurs when different users upload their recording of some public event.

Definition 7 (Scene Duplicate) v is a scene duplicate of u *if* v *presents the same real-world scene in* u *with a different capturing method (i.e., type of camera, capturing viewpoint, camera motions, etc.).*

Satoh et al. [2007] propose an approach for retrieving video footage taking the same scene or event, but from the different viewpoints. Similarly, Basharat et al. [2008] present an approach for

detecting two videos showing the same event but captured from different devices and viewpoints. These are examples of copy detection according to semantic similarity.

A more general semantic similarity can be defined as the videos that "a user would clearly identify as essentially the same" [Wu et al., 2007]. This human perception of duplicates is later analyzed by Cherubini et al. [2009]. Using an online questionnaire, they asked users to rate the similarity of pairs of videos. That study enabled them to present the concept of user-centric duplicates.

Definition 8 (User-Centric Duplicate) *v is a user-centric duplicate of u if v is semantically identical to u, i.e., v provides the same information to the user as u.*

In general, user-centric similarity admits most of the quality transformations and some post-production transformations, however it excludes the transformations that add new information (like insertions of graphs) and mash-up transformations. According to Cherubini et al. [2009], the acoustic content supports stronger transformations without affecting the semantic similarity of the video. For instance, Figure 4.2 shows two clips introducing the same physics experiment, hence from a user point of view those clips are duplicates, regardless of their visual dissimilarity.

Figure 4.2 – Two video clips showing the same physics experiment in similar conditions. These videos correspond to user-centric duplicates despite their visual difference. [Cherubini et al., 2009]

Despite the wide applications that CBVCD based on semantic similarity can provide, this thesis focuses on content transformation mainly because currently there is not a large-scale collection to evaluate CBVCD based on semantic similarity. In fact, TRECVID evaluation provides large datasets for CBVCD where the copies are produced by content transformations.

4.2 Applications of CBVCD

The techniques used at CBVCD can be applied to different problems. Each problem addresses a type of similarity (content transformation or semantic), and may limit the transformations to support. Some examples of applications of CBVCD methods are:

Checking of copyright infringements The owner of the copyright on some material would like to use a CBVCD system to find every derived work from its property. In this case the definition based on content transformations is relevant, where the transformations are created by an adversary that tries to avoid the detection. The main constraint is that transformations should be realistic, i.e., the final video should have enough quality to be watched by a final user. This application may

be preferred by video-sharing web sites and media production companies. An alternative approach focused on this application, called Watermarking, is discussed at the end of this Chapter.

Reduction of redundancy in search results In video search engines, a post-processing phase may use a CBVCD system to determine multiple copies of the same document and group them. In this case, the detection should focus on low-intensity content transformations and semantic similarity.

Identification of known sequences in video streams A CBVCD system can be used to monitor public broadcast with the objective of searching or tracking the emission of previously known commercials. The detection should focus on low-intensity transformations. Naturel and Gros [2005] present an approach to detect duplicated sequences using a lightweight representation and search.

The mining of video databases The analysis of common sequences between videos in a collection may help to discover internal structures in the collection. For example, if two videos share the title sequence, they can be marked as chapters of the same TV series. Poullot et al. [2008] present an approach to mine large archives of TV broadcast creating a graph showing the relationships between videos.

Assignation of semantic descriptions A CBVCD system may be used to add semantic tags to new documents based on the tags assigned to similar content. In this case, a CBVCD system based on semantic similarity is appropriate. Besides, tags can also be assigned based on the frequency of emissions, e.g., a video broadcast from many TV stations in a short period of time may be tagged as a news event, or a video broadcast for a longer period of time may be tagged as an historical event.

Video footage managing system The detection of common shots between unedited material and the final edited version of a film may help to reverse engineer the edition process of a film. In this case, CBVCD based on content transformations (in particular on mash-up transformations) is required.

4.3 Content Description

The Content Description consists in representing the content of the multimedia document with one or more descriptors. The descriptors should fulfill two goals: be invariant to transformations (i.e., be identical or very similar to the descriptors of documents that are copies); and be discriminative between unrelated documents (i.e., be very different to the descriptors of document that are not copies).

Descriptors can be *low-level* or *high-level* depending on the type of information they represent. Low-level descriptors represent some characteristic in the content itself (like colors, edges, shapes, pitch, etc.). They are the result of computational processes on the information stored in the document. High-level descriptors represent semantic features in the video (like the presence of

persons, musical genre, abstract concepts, etc.). Usually, high-level descriptors are generated by users, i.e., a person assigns one or more concepts to a video after having watched it. A broad field of research focuses on the problem of automatically producing high-level features. Machine learning techniques, like grouping low-level features into high-level categories by means of adequately-trained classifiers, are commonly used to this aim.

Depending on the source of information to represent, descriptors can be *visual* or *acoustic*. Additionally, high-level descriptors may represent both sources at the same time, in sorts of audio-visual concepts [Jiang et al., 2009].

In the case of multimedia documents with temporal dimension (audio and video), the description process may represent the entire document by a single description called *signature*, or may divide the document into small units and create independent *fine-grained* descriptors for each one.

Video signatures can be used in a first step for detecting identical videos and for discarding definitely irrelevant documents [Wu et al., 2007]. The signature can be calculated directly from the video content, or be a global summarization of fine-grained descriptions. However, in the case of mash-up transformations or strong postproduction transformations a fine-grained description is required.

The process of selecting independent representative frames from a video is called *keyframe selection*. The process of dividing a video into segments is called *video segmentation*. A *spatial* description represents static visual information from isolated keyframes. A *spatio-temporal* description represents the temporal flow of the visual content in a video segment. The search process for fine-grained descriptors locates similar segments or keyframes between query and reference videos, and then a *temporal consistency* process determines the matched units that joins excerpts between an original with its copy.

A description can be *global* or *local*. A global description represents the content of the whole keyframe or segment with a single descriptor. A local description consists of many descriptors, each one representing only a small zone of the keyframe or segment.

In general, local description can achieve higher detection performance than global description, especially for strong-intensity postproduction transformations [Law-To et al., 2007a]. However, it should be noted that local description presents some difficulties compared to global description: a) it requires more computational time to compute descriptors; b) it needs more disk and memory space to store descriptors; and c) it increases the complexity for the similarity search process. In order to reduce the storage and search time while keeping the high detection performance, a common approach is to create a global description summarizing a local description.

The following subsections detail these definitions and approaches, and review their common usage on CBVCD systems.

4.3.1 Description for an entire video document

The description of an entire video can be calculated by using a content-based hash function or by summarizing fine-grained descriptions.

In the case of content-based hashes, Coskun et al. [2006] propose two hash functions for calculating video signatures that depend on the spatio-temporal position of pixels, and are robust

to low-intensity quality transformations. The detection is performed by locating collisions between the hash values of videos. The first hash function achieves higher detection performance, but does not support attacks, i.e., an adversary is able to produce an unrelated video with the same hash or can produce similar videos with different hashes. The other hash function prevents adversary attacks by using a secret randomly-generated key, but it decreases its detection performance.

In the case of summarizations, Wu et al. [2007] create a video signature by averaging fine-grained global descriptors. The fine-grained descriptors are color histograms of 24 bins calculated on representative keyframes of shots, which are compared with the Euclidean distance. Depending on the distance between them, two videos may be reported as a copy (when signatures are close to one another), no copy (when signatures are far from each other), or otherwise a detailed comparison based on the fine-grained description is performed.

The Bounded Coordinate System (BCS) [Shen et al., 2007; Shao et al., 2008] calculates a statistical summarization of fine-grained descriptors. They calculate PCA over fine-grained global descriptors and transform each descriptor into the reduced space. The video signature is the average descriptor plus the minimum bounding rectangle that encloses the reduced descriptors. Two signatures are compared by summing the Euclidean distance between the averages and the Euclidean distances between the corners of the bounding rectangles.

Video signatures can be understood as a content-aware file hashing algorithm, like a CRC-32 or MD5, that computes its value from video pixels instead of file bytes. Therefore, they can be used to efficiently retrieve videos that visually are almost identical. In fact, they are intended to be used in scenarios where only quality transformations can be present. However, video signatures are not able to detect copies between video excerpts or strong postproduction transformations that may affect many consecutive frames.

4.3.2 Video Segmentation and Keyframe Selection

A video usually has between 20 and 30 frames per second, a movie has more than 100,000 frames, and a large database may contain more than 10^8 frames. However, most consecutive frames in a video are similar to one another. In order to reduce the amount of data, either representative keyframes or shot boundaries are calculated. A video shot is a series of interrelated consecutive frames taken contiguously by a single camera and representing a continuous action in time and space [Hanjalic, 2002]. A keyframe is the frame which can represent the salient content of a shot [Zhuang et al., 1999].

Shot boundaries are commonly located by searching significant changes between consecutive frames. For instance, shot boundaries can be detected by differences of average intensity [Hampapur et al., 1994], difference of gray histograms by zones [Boreczky and Rowe, 1996], difference of "intensity of motion" [Eickeler and Müller, 1999], or motion compensating features with adaptive threshold for hard and gradual transitions [Hanjalic, 2002]. TRECVID performed an evaluation for shot boundary detection between 2003 and 2007. Some systems that achieved good results on that evaluation are Naito et al. [2006], Liu et al. [2007], and Yuan et al. [2007].

Keyframes can be selected either from the whole video or from shots. The common approach for selecting keyframes are motion estimation [Fauvet et al., 2004], unsupervised clustering of histograms [Zhuang et al., 1998], difference between consecutive frames [Gengembre and Berrani, 2008], or just by a regular sampling.

The size of computed segments and/or the number of keyframes extracted by video affects both effectiveness and efficiency of a system. A sparse segmentation or keyframe selection increases efficiency but may ruin the detection effectiveness. On the other hand, a dense segmentation or keyframe selection increases effectiveness but affects the efficiency due to the high number of objects to process. Moreover, a too dense segmentation may also affect the effectiveness due to the high number of redundant and noisy objects.

4.3.3 Visual Global Description

The global description consists in representing the visual content of the video keyframe/segment with a single descriptor. The descriptor may represent only one keyframe in the segment (a spatial descriptor) or the whole segment (a spatio-temporal descriptor).

Computing global descriptors from video frames is a straightforward approach to address the CBVCD problem. Global descriptors are commonly used for detecting video duplicates, i.e., quality transformations, and in some systems, they form the first phase that resolves the easy-to-detect copies, leaving the complex copies to local descriptors.

One contribution of this thesis is to show that global descriptors can achieve satisfactory results in more complex datasets, even outperforming many state-of-the-art systems based on local descriptors. Global descriptors are able to detect many copies without producing false alarms, but they are not able to detect some copies with strong postproduction transformations, i.e., global descriptors can achieve higher precision than local descriptors but lower recall.

Spatial global descriptors

Hampapur and Bolle [2001] compare the effectiveness of different global descriptors extracted from regular-sampled keyframes. The evaluated descriptors are: image difference (the sum of the differences between pixel colors), color histogram, histogram of orientations of gradient, Hausdorff distance between Canny edges, Invariant Moments (spread and slenderness of Canny edges), and Local Edge (centroid of the edge points by zone). In their tests, the Local Edge performed better than the others at CBVCD, followed by the partial Hausdorff distance. This is one of the first works on the CBVCD topic. It focuses on low-intensity quality transformations (reencoding), and shows that the edge pixels contain discriminative information for the CBVCD problem.

In other work, Iwamoto et al. [2006] present a descriptor robust to captions which is very similar to the Edge Histogram descriptor (see Section 2.3.1). Each video frame is converted to gray scale and divided into N blocks. For each block the energy for ten different orientation filters is measured: the five orientations of Edge Histogram plus their mirrored versions (see Figure 4.3). The descriptor is a vector with N values containing the identifier of the dominant orientation for each block or a "no edge" identifier if no orientation exceeded a minimum threshold. The comparison uses a Hamming distance with predefined weights depending on the probability of caption superimposition for each block. This work shows that dividing the frame in zones and computing independent values for each zone produces a descriptor that is robust to localized modifications. In fact, some kinds of postproduction transformations, like insertion of captions or images, modify a few zones and the rest of the frame remains unchanged. Therefore, a robust global description must consider independent values for many zones of frames.

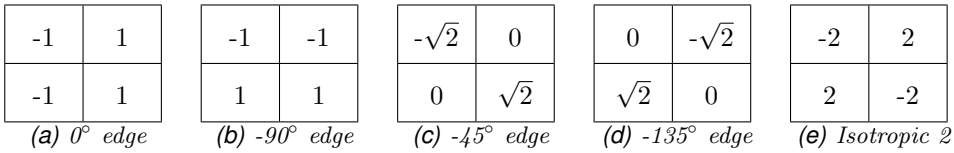


Figure 4.3 – The mirrored versions of the Edge Histogram filters (see Figure 2.8 on page 18).

Lee and Yoo [2008] compute a global descriptor on regular-sampled keyframes. A video frame is scaled, converted to intensity values and divided into N zones, the global descriptor is the centroid of the orientations of the gradient by zone. They show this descriptor outperforms the histogram of orientations of gradient from Hampapur and Bolle [2001]. The work reaffirms that edge pixels are indeed relevant for CBVCD.

The FRAS descriptor [Shen et al., 2007] maps frames to symbols from a dictionary. The symbol dictionary is defined by clustering all the frames in the database and selecting the centroids. A frame is represented by the centroid of the clusters that contain it, or a special symbol if it is not contained by any cluster. The descriptor for a whole video is the string resulting from concatenating all the symbols. The similarity between two descriptors is measured by a probability-based edit distance, which is similar to the Levenshtein distance, but using a probability of replacing symbols given by the number of common frames between clusters. This work shows a non-metric distance that computes similarity between videos. Unfortunately, the cost of distance computation is quadratic in terms of video lengths.

Spatio-temporal global descriptors

The Ordinal Measurement Descriptor (OMD) captures the relative ordering of intensities in an image (see Section 2.3.1). In the case of CBVCD, Kim and Vasudev [2005] extract OMD for each frame and compare sequences of descriptors. First, the spatial distance is defined as the average of the differences between two descriptors. Then, a temporal descriptor is calculated by comparing the changes between consecutive OMDs: the temporal OMD contains for each zone a value 1, -1, or 0, depending on whether the rank for the zone increases, decreases, or does not change, respectively. The temporal distance between two videos is defined as the average of the differences between their temporal OMDs. Finally, the spatio-temporal distance between video sequences is defined as a static weighted combination of spatial and temporal distances. This spatio-temporal distance follows the approach of weighted combinations of distances. Unfortunately, the temporal alignment of videos is determined by linear probing and the OMD descriptor is only robust to quality transformations.

Chen and Lian [2008] create the temporal OMD by sorting the average intensities of the same zone in the frame series. For a subsequence of m frames divided into n spatial zones, the temporal OMD consists in the n permutations (one for each zone) of m intensities (one for each frame). The distance between two temporal OMD is the average of the difference of permutations for each zone. This work improves the temporal localization of OMD, but it is also focused on quality transformations.

4.3.4 Visual Local Description

The local description consists in representing the visual content of the video keyframe/segment with many local descriptors. Each local descriptor represents a fixed zone in a keyframe (spatial descriptors) or a fixed or moving zone in the segment (spatio-temporal descriptors). The use of local descriptors enables the detection of complex postproduction transformations, which would otherwise be undetectable under global descriptions Law-To et al. [2007a]. Therefore, most state-of-the-art CBVCD systems rely on local descriptors to represent visual content. However, local descriptors demand much higher computational resources than global descriptors to store descriptors and perform searches. Additionally, the use of local descriptors usually requires the computation of spatial consistency processes.

The most common local descriptor used by CBVCD systems is SIFT or some variation of it, e.g., Basharat et al. [2008], Liu et al. [2010a], and Bai et al. [2011]. Other local descriptors used in CBVCD are SURF [Roth et al., 2009; Sun et al., 2010], DART [Younessian et al., 2010], CS-LBP [Jégou et al., 2010], and partial derivatives of graylevels [Joly et al., 2003; Poullot et al., 2007].

The temporal dimension can be used to improve the quality of local descriptors. Additionally, in order to improve the efficiency, the use of global summarization of descriptors is commonly used.

Spatio-temporal local descriptors

A common technique to improve the quality of local descriptors in videos is to track them between video frames. This technique also enables discarding unstable regions that usually correspond to noise.

Law-To et al. [2006] propose an algorithm for spatio-temporal local description which classifies persistent keypoints by tracking their trajectories in consecutive keyframes. If the points are static, they represent the background, and if they are moving, they represent objects in motion. This distinction enables the detection of strong postproduction transformations, like background replacement and insertion of new characters in a scene. This work proves the potentialities of local descriptors compared to global descriptors.

Satoh et al. [2007] detect and track interest points in a video using the Kanade-Lucas-Tomasi tracking algorithm [Tomasi and Kanade, 1991]. Following the work by Shechtman and Irani [2005], a trajectory is represented by motion patterns and the comparison between two videos consists in measuring their degree of inconsistency between their patterns. A binary signature is computed for each trajectory by locating the local maximum of inconsistency. This work focuses on detecting copies according to semantic similarity instead of copies based on content transformations. The works reports high precision and low recall, i.e., it reports few false alarms but misses many copies. This technique demands high computational resources to compare videos, therefore it may not be able to scale to common CBVCD datasets.

Similarly, Basharat et al. [2008] detect keypoints in time and compute trajectories. The spatial regions containing trajectories with uniform direction and velocity are used to segment physical volumes in the scene. The volumes are represented by SIFT descriptors, HSV histogram, histogram of gradients, and histogram of motion directions. The similarity is measured by the Earth Mover's Distance using centroids of each descriptor space. Due to the required resources to compute similarity between videos, this technique presumably is not able to scale to common

CBVCD datasets.

Another approach to improve the quality of local descriptors is to directly compute spatio-temporal descriptors. SURF is extended to the temporal dimension by means of a spatio-temporal keypoint detection [Willems et al., 2008a] and a spatio-temporal descriptor [Willems et al., 2008b]. These extensions are tested on a CBVCD system using synthetic quality transformations. Unfortunately, to the best of our knowledge, a comparison of the performance of this spatio-temporal local description with other CBVCD systems does not exist.

4.3.5 Global description based on local descriptors

A common technique to reduce search times is to compute summarizations of local descriptors during the offline phase. This summarization alleviates the need to compare full sets of descriptors during the online phase.

Roth et al. [2009] summarize the distribution of local descriptors in a frame by creating a 16-bytes global descriptor. The global description is the count of local descriptor that belongs to each of the 4×4 quadrants, where the maximum count per quadrant is 255. Two descriptors are compared by just summing up the differences between quadrants. This system did not achieve high detection effectiveness at TRECVID 2009, probably because this kind of aggregation produces a high rate of false alarms.

The Bag-of-Visual-Words approach is widely used for addressing different problems in computer vision area (see Section 2.3.3). In the CBVCD problem, this approach has shown high efficiency improvements and low impact in the effectiveness. The efficiency gains are achieved by the use of lookup tables during the search. It should be noted that the computation of codebooks demands high computational resources, however they are usually not reported because that cost is part of the offline phase, i.e., the cost does not depend on queries to be processed.

Most of the top performing CBVCD systems use some form of BOVW or other codebook-based descriptors in their processes. For instance, Douze et al. [2008] computes a codebook of 200,000 visual words from SIFT descriptors, and additionally they enhance the BOVW by computing the Hamming Embedding signatures. In other work, Jiang et al. [2011] compute a codebook of 800 visual words on dense sampled color SIFT descriptors. They increase the performance by computing spatial pyramids. Similarly, Bai et al. [2011] compute a codebook of 50,000 visual words from SIFT descriptors.

An open issue of codebooks and CBVCD is to determine the extent to which the quantization of descriptors increases or decreases the effectiveness compared to using the local descriptors themselves. In the case of image classification, there is some evidence that using codebooks to produce "mid-level" descriptors can improve effectiveness [Boureau et al., 2010]. However, in the case of CBVCD, particularly for copies based on content transformations, this generalization might just increase the amount of false alarms instead of improving effectiveness.

An alternative approach is to compute the "glocal" descriptor [Poullot et al., 2008; Le et al., 2009] which is a binary sequence that summarizes the local descriptors in a static frame. The space of the local descriptors is hierarchically partitioned into 2^h hyper-rectangular cells. The distribution of descriptor in cells is analyzed in order to confirm that a relatively homogeneous partition is performed. The glocal descriptor is a 2^h-length binary sequence, where the i^{th} bit is set to 1 if at least one local descriptor falls in the i^{th} cell, or 0 otherwise. The system uses h=8,

thus the glocal descriptor is a 32-byte sequence. This brief description enables the processing of very large datasets (in the order of thousands of hours) at the cost of decreasing effectiveness due to false alarms.

4.3.6 Acoustic Description

The acoustic description is usually computed by short-time analysis of the acoustic signal at regular intervals. The analysis is based on converting the signal into the frequency domain (by means of the Fourier transform) and analyzing the energies of the audible frequencies.

Haitsma and Kalker [2002] present an acoustic description based on energy differences in consecutive sub-bands. It consists of a sliding window of 11 ms, and for each window it selects 33 non-overlapping frequency bands in the logarithmic frequency domain. These bands lie in the range from 300 Hz to 2 kHz (the most relevant audible spectral range). A 32 bit binary descriptor is generated by comparing the difference in the energy of consecutive bands for two consecutive windows: 1 means an increase in the difference, i.e., $E(n,m) - E(n,m+1) > E(n-1,m) - E(n-1,m+1)$ for the n^{th} window and m^{th} sub-band; and 0 otherwise. A match between audio segments is performed by calculating a Hamming distance between the binary descriptors.

Saracoğlu et al. [2009] also use the energy differences descriptor with the following differences: window size of 25 ms, frequency range from 300 Hz to 3 kHz, 16 sub-bands, and the m^{th} bit of in the descriptor is 1 when $E(n,m) > E(n,m+1)$ for the n^{th} window and m^{th} sub-band. This configuration is also used by CBVCD systems Gupta et al. [2010] and Younessian et al. [2010].

The Mel-frequency cepstral coefficients (MFCC) are widely used descriptors for speech recognition. Some systems using MFCC descriptors for CBVCD are Liang et al. [2009], Anguera et al. [2009a], Natsev et al. [2010], and Gupta et al. [2011].

4.3.7 High-level Description

Min et al. [2010, 2011] face the problem of semantic similarity by first assigning high-level semantic features to each shot. The semantic vector for each shot is created by extracting low-level features and using trained classifiers for 32 concepts (like park, people, indoor, flowers, and others). The semantic description for each shot is a 32-dimensional binary vector denoting the existence or inexistence of each concept in the shot. A video signature is created by calculating the entropy of the occurrence of each concept in the semantic vectors of the shots. Additionally, some low-level features are extracted from each shot. The distance between two videos is a weighted combination of the distance between semantic descriptors and the distance between global descriptors.

This approach assumes that high-level concepts are able to discriminate between copies and non-copies. In fact, this may be the case in collections containing videos from a broad range of subjects and styles. However, in general this approach will miss some copies with content transformations and also produce a high rate of false alarms. For example, it will miss copies with some picture-in-picture transforms, and will report false alarms between different videos proceeding from the same scene. In fact, the evaluation in that work uses datasets based on content transformations (MUSCLE-VCD-2007 and TRECVID 2008), and the improvement in effectiveness due to semantic descriptors is rather marginal.

4.4 Similarity Search

The similarity search corresponds to the algorithm that locates descriptors within the database that make a close match with the descriptors computed from a query video. In the case of videos signatures, a single search is enough to return similar videos. In the case of fine-grained global or local descriptors, one search should be performed for each descriptor in the query video.

In Chapter 3 we reviewed general aspects of the similarity search and common indexing techniques. In this section we focus on the techniques that are preferred by CBVCD systems. In particular, we distinguish three approaches in current CBVCD systems: linear scan, lookup tables, and space filling curves.

4.4.1 Linear Scan

The linear scan corresponds to sequentially comparing every query descriptor with every reference descriptor and to retrieve those that are the closest.

Kim and Vasudev [2005] perform a linear scan using their spatio-temporal distance function. A copy is detected when the distance between query video and a reference excerpt is lower than a threshold. The main drawback is that it compares the query video with reference videos probing every possible alignment.

Gupta et al. [2010] sequentially compare acoustic descriptors following a linear scan. The linear scan is efficiently implemented on GPU, therefore it is possible to be used on a large collection. A system following this approach achieved high performance in TRECVID [Gupta et al., 2011], as is reviewed in Section 11.6.2.

The linear scan is usually not considered to address large video collections due to the high cost involved in computing the distance between every pair of query and reference descriptors. However, it should be noted that if a system is able to afford the resources required by the linear scan then it can indeed achieve high effectiveness in the similarity search. This fact motivates the research on metric indexes, which present an alternative for improving the efficiency and achieving high effectiveness. More details on this issue are given in Section 9.3.

4.4.2 Lookup Tables

The lookup table approach is a technique based on defining a fixed set of n possible values, and assigning every descriptor to one or more of those values. The lookup table (or inverted index) is an array of n entries, where each entry contains a list with all the descriptors that are assigned to each value. The similarity search is resolved by using the lookup table to retrieve those descriptors that are assigned to the same values than the query objects. The method that assigns a descriptor to one or more of the n values may be a hash function (as in Locality Sensitive Hashing), or may correspond to the relevant codewords from a codebook (as in Bag-of-Visual-Words approach).

The lookup tables can provide a great improvement in search efficiency. However, the quantization error produced by assigning multidimensional vectors to a fixed set of n values may harm the search effectiveness. In order to enhance the discrimination of table entries, the information stored by each entry may be augmented to consider some extra information for each descriptor (as

described in Section 2.3.3). Additionally, in order to reduce the false positives produced by collisions between dissimilar descriptors, a spatio-temporal consistency step is usually required.

Le et al. [2009] organize glocal descriptors of n-bits in a lookup table, where each symbol represents three bit positions (between 0 and $n-1$). A weak geometric consistency step locates triangles that are likely to be stable between copies, which defines a triplet of bit positions. For each triplet, the glocal descriptor is added to the corresponding list in the lookup table. This system participated in TRECVID 2009, achieving high performance in the balanced profile but lower performance in the no false alarms profile.

Liu et al. [2010b] use LSH with a family of m hash functions. Each hash function maps a local descriptor to one of n symbols. In order to resolve a search, for each local descriptor the search uses the m hash functions to retrieve the approximate nearest neighbors. A strong spatial consistency (RANSAC) is performed between query descriptors and retrieved candidates in order to reduce the false alarms rate. According to its participation in TRECVID 2010, this system achieves high performance when a certain amount of false alarms are permitted, however it achieves low performance when false alarms are not allowed (see results in Appendix B).

Bai et al. [2011] present a system that follows the BOVW approach: it computes a codebook with 50,000 visual words and organizes the quantized descriptors in an inverted index. It considers a strong process of spatial consistency (Hough transform). According to its participation in TRECVID 2011, this system achieves high performance when false alarms are permitted, however it decreases its performance when false alarms are not allowed (see results in Appendix C).

4.4.3 Space Filling Curves

This approach partitions the descriptor space into predefined zones, and produces a spatial ordering of zones according to some predefined space-filling curve. For each zone a probability density function is determined, and the search retrieves the zones with high probability of containing a relevant object, and evaluates the distance to all the objects in them.

Joly et al. [2007] propose to partition the space into hyper-rectangular blocks using a Hilbert space-filling curve. The distribution of descriptors for each block is estimated by a normal distribution, with the same standard deviation in all the dimensions, and determined by training data. The search consists of selecting the blocks whose combined probability of containing a query vector is greater than a given threshold. This approach was tested in TRECVID by Joly et al. [2008] achieving high detection performance, however it was outperformed by a system using the BOVW approach [Douze et al., 2008]

Poullot et al. [2007] present an improvement of probabilistic discard, where the space is partitioned into hyper-rectangular blocks following a Z-space filling curve instead of the Hilbert curve. With this change, all the blocks are the same size and orientation, simplifying the calculation of block boundaries. This system was tested in a database with 60,000 hours of video, and in a extension work it was tested with 280,000 hours of video [Poullot et al., 2010]. This system shows good performance at MUSCLE-VCD-2007, unfortunately it has not been evaluated with some TRECVID dataset.

4.5 Temporal Consistency

The temporal consistency consists in analyzing those similar frames or segments returned by the similarity search in order to determine whether there exists one or more copies in a query video and the boundaries of each one.

Joly et al. [2007] use an extension of the spatial consistency between local descriptors (see Section 2.3.2) including the temporal dimension as an additional variable. The RANSAC algorithm is used to determine the best alignment between query and reference video.

Tan et al. [2009] model the temporal consistency problem between a query and reference videos as directed graph. The frames of the videos correspond to vertices, and the similarity between frames corresponds to the weights of the edges in the graph. The copied excerpt is determined by the optimal solution of a network flow problem.

Chapter 10 presents a novel algorithm for temporal consistency that can be used to locate copied excerpts from the results of k-NN searches. Unlike previous works, this algorithm can profit from several nearest neighbors by weighting their ranks and distances.

4.6 Multimodal Fusion

In order to achieve a highly effective detection, CBVCD systems should use many sources of information to perform the detection process. In fact, the top-performing CBVCD systems usually fuse many modalities, i.e., combine different kinds of descriptors to perform a detection process, like global descriptors, local descriptors, and/or acoustic descriptors.

Most of the current CBVCD systems perform a multimodal detection by dividing the system into independent subsystems, where each subsystem performs a detection using a single different modality. For instance, let A and B be two subsystems for a CBVCD system, and assume A uses visual descriptors and B uses audio descriptors. A and B perform an independent copy detection process, each one producing a set of copy candidates C^A and C^B, respectively. The commonly used approach for multimodal fusion creates the final detection list C by combining detections and scores from C^A and C^B.

Snoek et al. [2005] describe two approaches to perform multimodal fusion for the video semantic indexing problem: the "early fusion" and the "late fusion". Adapting those definitions to CBVCD, the multimodal fusion approach described in the previous example corresponds to late fusion, because the multimodal detections are the result of combining unimodal detections. On the other hand, the early fusion combines modalities before determining copy candidates.

4.6.1 Early Fusion

The early fusion combines information between modalities at an earlier stage, rather than at the final decision step. Extending the definition of Snoek et al. [2005], we differentiate three stages for early fusion: content description, similarity search, and temporal consistency.

In the case of early fusion at the content description, a multimodal descriptor is created

by concatenating descriptors from different modalities. Snoek et al. [2005] performs a comparison between early and late fusion for the semantic video indexing using this approach. The results suggest that (in general) late fusion achieves a slightly better performance, while only in specific cases, the early fusion outperforms the late fusion.

In the case of early fusion at the similarity search, a multimodal search is performed by comparing descriptors from different modalities. Barrios and Bustos [2009] present an image retrieval system that combines text-based descriptors with visual-based descriptors at the distance function. Batko et al. [2009] present an image retrieval system that combines many visual-based descriptors using distances with predefined weights.

In the case of early fusion at the temporal consistency, a multimodal detection is created by analyzing the results of similarity searches produced by independent modalities. Anguera et al. [2009b] present a CBVCD system that performs a comparison between query and reference videos using global descriptors and acoustic descriptors, and then produces the copy detections by measuring the correlation between their similarities. Ayari et al. [2011] present a CBVCD system that compares the performance of early and late fusion of local descriptors and acoustic descriptors. The early fusion is performed by analyzing the results of the similarity search of both modalities. The results suggest the early fusion achieves better performance than the late fusion.

In Chapter 8 we test the performance of CBVCD using early fusion at the similarity search by combining global descriptors and acoustic descriptors into a single distance function. Section 9.3.3 compares the performance of early fusion with the late fusion of partial nearest neighbors.

4.6.2 Late Fusion

The late fusion combines the copy detections from subsystems using a single modality. In general, the late fusion processes follow two approaches: a) choose candidates from one subsystem following some heuristic rule depending on prior knowledge of the performance of each modality; or b) merge candidates from all subsystems using a weighted sum of the confidence scores given from each subsystem.

Wu et al. [2007] perform a hierarchical combination of results between video signatures and local descriptors. Depending on the distance between video signatures, the system may decide to: declare is-a-copy (when the distance is near zero), declare is-not-a-copy (when the distance is above a threshold), or otherwise delegate the decision to a subsystem based on local descriptors.

Saracoğlu et al. [2009] combine acoustic descriptors and global descriptors. The fusion consists of choosing the detection with highest confidence that exceeds a certain threshold. Liang et al. [2009] combine acoustic descriptors and local descriptors. The fusion is either the union or intersection of the candidates, depending on whether the final result can or cannot contain false alarms, respectively. Mukai et al. [2010b] combine acoustic descriptors and global descriptors. The fusion joins candidates from both subsystems, but prefers the acoustic-based candidates in case of conflict. Le et al. [2009] combine acoustic descriptors and glocal descriptors. The fusion joins or intersects both detections, and the final confidence score is a weighted sum favoring audio scores.

Anguera et al. [2011b] present a late fusion algorithm specific for CBVCD. It is based on the weighted sum of scores, where the scores are dynamically normalized according to their distribution for each modality. This algorithm can fuse an arbitrary number of modalities and it does not require any specific distribution for the scores, hence it can combine almost any CBVCD result.

In Section 11.6 we compare the performance our system using early fusion to other systems using late fusion. We also analyze the participation of the Telefonica team at TRECVID 2011 [Anguera et al., 2011a], which uses late fusion to combine acoustic descriptors, local descriptors, and our system. We then use these results to show benefits and drawbacks of these two fusion approaches.

4.7 Alternative Approach: Watermarking

Watermarking consists of embedding invisible information, called a *watermark*, into an original video. This process should be performed by the producer or distributor of the content before being broadcast. Then, given a query video from an unknown source, the copy detection system detects and retrieves the watermark from the video, and compares it with its database of watermarks. The result of the search may indicate the owner of the video, its copyrights, the allowed usages, and other information to determine whether the video is a legal copy or not [Swanson et al., 1998; Wolfgang et al., 1999].

The watermarking approach is focused on checking copyright infringements, i.e., a media owner enforces its rights over the distribution of a multimedia content (either an image, audio or video document) by adding invisible information in the content. Another use of watermarking is to check the authenticity of the content, i.e., to verify that the image or video has not been edited after the watermark embedding.

In general, the problem of embedding a secret message into a public communication channel is known as steganography. The watermarking uses steganography to hide a message about the communication channel. Langelaar et al. [2000] present an overview of methods for watermarking, like adding pseudo-random noise pattern, DFT amplitude modulation, and modification of the least significant bits. These methods present some strengths and weaknesses for common content transformations.

The main problems with watermarking are that an early manipulation of the documents is required, and watermarks may not be robust enough to withstand some content transformations (like strong quality or postproduction transformations). Additionally, despite watermarks being usually imperceptible, some domains may require the highest possible image quality (like medical images).

CBVCD systems can be seen as watermark-based copy detection systems where the watermarks are calculated from the content itself. Therefore, CBVCD avoids embedding any extra information into the media, enabling copy detection processes to already broadcasted content.

4.8 Summary

In this chapter we have depicted the main topics that are involved in a CBVCD system. We have also discussed the research work that has been done on these topics.

Despite the improvements that have been shown in recent years, the CBVCD is still a challenging problem. The main issues that researchers are currently facing are the development of new techniques for managing huge sets of local descriptors, the development of different fusion

techniques, and the use of CBVCD in realistic environments.

Part II

CBVCD and the Metric Space Approach

Chapter 5

Overview

This chapter gives a general overview of our CBVCD system, called P-VCD, and its components. The details for each component are given in following chapters.

Let $\mathcal{V}$ be a set of original videos (the reference collection), $\mathcal{C}$ be a set of query videos (the query collection), and $\mathcal{T}$ be the set of content transformations that may have been applied to a reference video to create a query video. In practice, the objective of a CBVCD system is to produce a list of copy detections $(\bar{c}, \bar{v}, s)$, where $\bar{c}$ is an excerpt from query video $c \in \mathcal{C}$, $\bar{v}$ is an excerpt from reference video $v \in \mathcal{V}$, and $s \in \mathbb{R}^+$ is a confidence score for $\exists\, t \in \mathcal{T}$, $t(\bar{v}) = \bar{c}$.

P-VCD is divided into five main tasks: Preprocessing, Video Segmentation, Feature Extraction, Similarity Search, and Copy Localization (see Figure 5.1). These tasks work as a pipeline: each task reads the required input from files in secondary storage, performs the desired work, and then writes back the output to storage (possibly creating new files with some predefined format).

A general overview of the system is the following: 1) the preprocessing task processes every video in $\mathcal{C}$ and $\mathcal{V}$ in order to normalize videos and diminish content transformations effects; 2) the video segmentation task partitions every video into short segments, producing a set of query segments and a set of reference segments; 3) the feature extraction task calculates many descriptions for every segment; 4) the similarity search task performs many NN+range searches to retrieve the most similar reference segments for each query segment; finally 5) the copy localization task uses the similar reference segments to locate chains that belong to the same reference video, producing the final set of detections $(\bar{c}, \bar{v}, s)$.

The main motivation for this design is to clearly isolate the similarity search from the other tasks (specifically, from feature extraction and copy localization). This isolation enables focusing the research on developing techniques for improving effectiveness and efficiency using widely-used descriptors. This is a key aspect for this work: the relevance of the similarity search (including distance definition and multimodal fusion) in the system performance. This marks a difference with some research in CBVCD that focuses on developing novel robust descriptors to different transformations, but keeps the similarity search as linear scans.

Following sections summarize each of these tasks and their relationships. Additionally, we present the dataset and the evaluation we used for the experiments in Chapters 7, 8, 9, and 10.

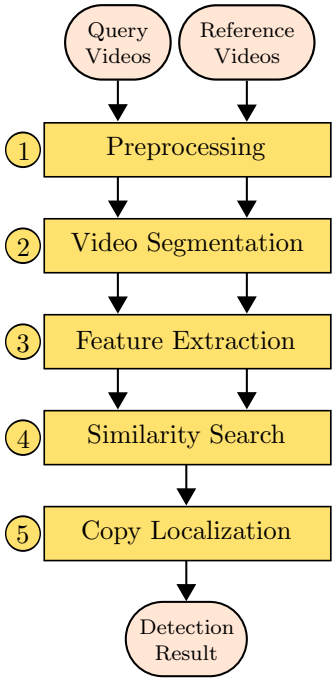


Figure 5.1 – The five main tasks of the P-VCD system.

5.1 Preprocessing Task

The video preprocessing is a common step in CBVCD systems. Its main benefit is the improvement in the detection effectiveness by giving to descriptors invariance or improved robustness to some content transformations. On the other hand, the video preprocessing usually affects the efficiency of the system due to the computational cost needed to analyze and enhance videos. An analysis of the impact of preprocessing on system effectiveness is shown in Section 11.6.3.

Specifically, this task has two objectives: 1) to normalize or enhance the quality of videos; and 2) to diminish the content transformations effect on query videos.

In order to normalize videos, every video is analyzed and the frames without information or too noisy are removed. For each reference video $v \in \mathcal{V}$ its normalized version v' is added to the output set $\mathcal{V}'$, analogously for each query video $c \in \mathcal{C}$ its normalized version c' is added to the output set $\mathcal{C}'$. In order to diminish content transformations effects, new query videos are created by detecting and reverting some specific transformations. The reversion process may create one or more query videos. New query videos are normalized and added to $\mathcal{C}'$.

The output for this task is a new set of normalized reference videos $\mathcal{V}'$ and a new set of normalized and reverted query videos $\mathcal{C}'$. It is expected that $|\mathcal{V}| \simeq |\mathcal{V}'|$ and $|\mathcal{C}| \leq |\mathcal{C}'|$. The following tasks use these new sets to detect copies. The Copy Localization task converts detections between $\mathcal{C}'$ and $\mathcal{V}'$ into detections between $\mathcal{C}$ and $\mathcal{V}$. Chapter 6 details the preprocessing techniques applied by the system to reference and query videos.

5.2 Video Segmentation Task

The objective of this task is to partition every video into short segments. A video segment is just a group of similar consecutive frames $\{f_i, f_{i+1}, ...\}$. The segments do not need to be all the same length.

The segmentation task allows the system to detect copies that may be shorter than the query video. Essentially, the segmentation task divides every video into independent segments, the similarity search task locates similar segments in the collection, and the copy localization task restores the temporal unity obtaining the copy boundaries. Therefore, in order to detect a copy, the original and the duplicated excerpt must have been divided into several segments.

A shot partition usually produces a too coarse segmentation to achieve high detection effectiveness. In fact, some corpus contain copies shorter than a single shot, like TRECVID's dataset where copy excerpts can be just three seconds long. On the other hand, a too fine segmentation directly affects the efficiency due to the high number of segments to process, and also may affect the effectiveness due to noisy segments that may produce false alarms with high score. The impact of segment length on system effectiveness is evaluated in Section 7.6

Depending on the type of descriptor to compute, the system may need to select a keyframe for each segment. In the case of spatial descriptors, each segment s must define a representative keyframe f_s from which the descriptor is computed. In the case of spatio-temporal descriptors, the segments do not need to define keyframes because the descriptor depends on all the frames in the segment.

The output of this task is a set of segments $\{s_1, ..., s_r\}$ for every query and reference video. Chapter 7 details the segmentations used by the system and compares their impact on effectiveness.

5.3 Feature Extraction Task

The objective of this task is to create one or more descriptions (global, local, acoustic) for each video segment.

A feature extraction method is defined by the pair (g, d) where: $g : \mathcal{S} \to \mathcal{F}$ is the extraction function; $d : \mathcal{F} \times \mathcal{F} \to \mathbb{R}$ is the distance function for comparing two descriptions; $\mathcal{S}$ is the set of video segments; $\mathcal{F}$ is the feature space; and $g(s)$ is the description for segment s. In general, the description $g(s)$ can be a global descriptor for the representative frame, a spatio-temporal descriptor for the whole segment, local descriptors for the representative frame, or an acoustic descriptor for the audio track in the segment. Chapter 7 details the descriptions that are used in this thesis and compares their effectiveness using different segmentations.

The task establishes two requirements for feature extraction methods: 1) $g(s)$ should represent the whole segment s; and 2) the feature extraction method should not be severely affected by the content transformations that might have been applied to s. If the descriptor is highly affected by transformations, the inclusion of some normalization or reversion in the preprocessing task should be considered. A third desirable property is that d should satisfy the metric properties. The feature extraction task is restricted to use a unique segmentation for all the feature extraction methods, i.e., to extract different descriptors from the same segments. This restriction enables the early fusion

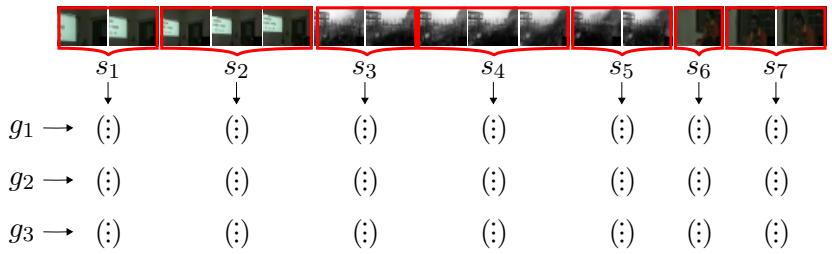


Figure 5.2 – Video Segmentation and Feature Extraction tasks: The video is partitioned into segments by the Video Segmentation task, and for each segment the Feature Extraction task calculates one or more descriptions.

because descriptors represent the same excerpt. On the other hand, the late fusion does not have this restriction because the combination is performed after the similarity search. An analysis and a comparison between early and late fusion is given in Section 11.6.2.

The feature extraction task makes no restriction about the descriptor format, allowing methods to create descriptors as multidimensional vectors, binary signatures, variable length vectors, sets of vectors, or any other format. This independence from the format is possible whereas the distance function is able to compare them and compute a distance value. The main strength of this approach is that it provides two ways for improving effectiveness: 1) to represent and store descriptors in any format that may better represent the data instead of restricting to fixed-length multidimensional vectors; and 2) to compare descriptors using functions that may better implement the concept of similarity instead of restricting to Minkowski distances. On the other hand, this generalization directly affects the efficiency, but (as a general rule) it is the responsibility of the next task (similarity search) to balance the effectiveness-versus-efficiency tradeoff.

The output of this task is a set of m descriptors $\{g_1(s), ..., g_m(s)\}$ for each segment s, and a set of m distance functions $\{d_1, ..., d_m\}$ to compare each type of descriptor. Figure 5.2 depicts the Video Segmentation and Feature Extraction tasks.

5.4 Similarity Search Task

The objective of this task is to perform NN+range searches to efficiently retrieve for each query segment q a list with its k most similar reference segments closer than a distance threshold ϵ. More formally, for every $q \in \mathcal{Q}$ this task calculates the list $N_q = \{(r^1, d(q, r^1)), ..., (r^k, d(q, r^k))\}$, where $r^j \in \mathcal{R}$ is the j^{th} nearest neighbor to q, $d(q, r^j) \leq d(q, r^{j+1})$, and $d(q, r^j) \in [0, \epsilon]$.

Controlling the effectiveness-versus-efficiency tradeoff is a key issue for this task. The similarity search should achieve as high effectiveness as possible (i.e., to retrieve all the segments that are indeed copies) and be as efficient as possible (i.e., to require few resources, including memory and search time). On one hand, effectiveness is usually determined by the quality of descriptors extracted by the previous task. In this thesis we show different techniques that can improve their basal effectiveness at the cost decreasing efficiency. On the other hand, a basal efficiency is achieved by the linear scan, which can be improved with no cost at effectiveness by using some index structure and exact searches. The efficiency can be highly improved by using approximate searches at the cost of decreasing effectiveness.

This task addresses two main issues:

1. To define the distance function d between two segments. In the case of a single descriptor per segment, the distance may be the same d_i defined for the descriptor. In the case of many descriptions per segment, a distance based on $\{d_1, ..., d_m\}$ can be defined. In Chapter 8 we study techniques for defining distances that improve the effectiveness by performing an early fusion of descriptors. We show that very simple descriptors can indeed achieve high effectiveness if they are combined wisely. In particular, we study the effectiveness of linear combination of distances, spatio-temporal distance, and non-metric distances.

2. To perform NN+range searches for every query object using d. In the more general case, the search cannot make assumptions about underlying descriptors, impeding the use of multidimensional indexes. In that case, if d satisfies the metric properties, a metric index can be used, otherwise some non-metric index may be considered (see Chapter 3). In Chapter 9 we focus on techniques for efficiently resolving searches for metric distances d. In particular we study the approximate searches using static pivot tables, and exact searches using dynamic pivot tables.

As discussed in Section 3.4, it is expected that metric indexes achieve lower efficiency than multidimensional indexes. Moreover, if d defines some complex similarity models, it can also be expected it will demand high computational resources, affecting the efficiency. A comparison considering search efficiency between multidimensional indexes and metric indexes is presented in Section 9.3.

The output of this task is a set of similar reference segments (i.e., the nearest neighbors according to d) for every segment in query video.

5.5 Copy Localization Task

The objective of this task is to decide on the existence of a copy by analyzing the result of the similarity search. The input for this task is a query video $c' \in \mathcal{C}'$ partitioned into the segments $\{s_1, ..., s_r\}$ and the output of the similarity search $\{N_{s_1}, ..., N_{s_r}\}$ where N_{s_i} is the list of the nearest neighbors for query segment s_i.

The task traverses the lists of nearest neighbors looking for chains of similar segments belonging to the same reference video. It produces a list of copy candidates $(\bar{c}', \bar{v}', s')$ between the preprocessed sets $\mathcal{C}'$ and $\mathcal{V}'$. Finally, it combines the detections for the query videos created by the Preprocessing task producing the final sets of triplets $(\bar{c}, \bar{v}, s)$.

Chapter 10 details the localization task and its voting algorithm.

5.6 Evaluation of CBVCD

The evaluation compares two aspects: the effectiveness in the detection and the efficiency in the detection. In order to evaluate the effectiveness, the system performs a detection on a dataset with known results (i.e., a ground-truth) and the differences between detected copies and the actual copies are measured. In order to evaluate the efficiency, we compare the computational effort required by linear scans (i.e., searches without indexing at all) with the computational effort required by the proposed algorithms.

Additionally, the system is also evaluated by comparing its effectiveness and efficiency with other state-of-the-art systems. The system participated on the Content-Based Copy Detection task at TRECVID 2010 and 2011. The results of these participations, datasets, and comparison with other systems, are detailed in Chapter 11.

The MUSCLE-VCD-2007 dataset [Law-To et al., 2007b] is a publicly available and widely-used collection for evaluation of CBVCD systems. It was created as the corpus for a CBVCD evaluation performed at CIVR 2007. Between the public databases for video copy detection, we have chosen this dataset because it has a well defined ground-truth (it is possible to suit a frame-to-frame match between query videos and reference videos), and it has an appropriate size for the evaluations: it is big enough to be a challenging database, and exact searches can be resolved in reasonable time, thus it enables us to analyze the real effectiveness of different techniques.

5.6.1 Dataset

The MUSCLE-VCD-2007 dataset consists in one reference collection, called DB-MPEG1, and two query collections, namely ST1 and ST2. Table 5.1 summarizes the sizes of these collections.

Collection	Videos	Hours	Frames	GB
Query	18	3.9	348,321	2.3
ST1	15	3.1	281,052	1.8
ST2	3	0.7	67,269	0.4
Reference	101	58.7	5,285,921	34.3

Table 5.1 – Summary of MUSCLE-VCD-2007 collection.

The reference videos have varying lengths, ranging from 17 seconds for the shortest one, up to 115 minutes for the longest one. The reference videos proceed mainly from TV rips, thus they have reasonably good quality (352×288 and 25 fps).

The query collection was created by applying some content transformation to a reference video. The visual transformations vary between: color adjustment, blur, resize, crop, vertical mirroring, noise, insertion of subtitles, insertion of logo, and manual camcording. The audio transformation for most of the videos is the reencoding of the audio track. For videos with manual camcording the audio was recorded with the camcorder (thus the audio contains ambient noise and low volume), and there is one copy with muted audio track, possibly due to an unintended corruption in the query video. There is no acceleration transformation, thus the copy and reference excerpts are the same length.

The ST1 collection tests full-length copies. It consists in fifteen videos, where ten proceed from the reference collection and the other five proceed from videos not in the reference collection. The query videos are almost identical in length as the corresponding reference videos. The ST2 collection tests partial-length copies. The three videos proceed from videos not in the reference collection, and they contain 21 embedded excerpts extracted from the reference collection.

The details of the ground-truth is shown in Appendix A along with example frames from query videos.

5.6.2 Evaluation measures

In the following chapters, we evaluate different aspects of the CBVCD system. Each experiment first defines some configuration (i.e., a type of preprocessing, a video segmentation, the description for each segment, and the distance between segments), and then it performs the similarity searches. The results of the searches are evaluated according to two aspects: the frame-by-frame matching effectiveness and the copy detection effectiveness.

Frame-to-frame match

These measures evaluate the performance of matching a copied frame with its original frame. Let $\mathcal{Q}$ be the set of segments from query videos and $\mathcal{R}$ be the set of segments from reference videos. Let $\mathcal{Q}_c \subseteq \mathcal{Q}$ be the set of query segments that have a matching original segment in $\mathcal{R}$ according to the ground-truth (i.e., the segments that are copies).

For this evaluation, we stated every segment must have a representative frame. The correct answer for a query segment s with representative frame f_s is the reference segment r that contains the frame R_{f_s}. The frame R_{f_s} is the original frame for f_s, i.e., $\exists\, t \in \mathcal{T},\, t(R_{f_s}) = f_s$. The frame R_{f_s} is determined by the ground-truth. Some examples of frames f_s and corresponding R_{f_s} are given in Appendix A.

Given a configuration and a query object $q \in \mathcal{Q}_c$, the similarity search sorts all the segments in $\mathcal{R}$ from the most similar to the less similar to q. Then, using the ground-truth, we determine the rank of each correct answer. The best value for $rank(R_q)$ is 1 and the worst value is $|\mathcal{R}|$. Finally, we evaluate the frame-to-frame effectiveness with two measures:

- **Mean average precision** (MAP): The average precision for a single query is just the inverse of the rank of R_q (each query has one correct answer), then MAP is defined as:

$$\text{MAP} = \frac{1}{|\mathcal{Q}_c|} \sum_{q \in \mathcal{Q}_c} \frac{1}{rank(R_q)}$$

  MAP values range between 0 and 1, where 1 means best effectiveness.

- **Correct *k*-NN searches** (Nk): The percentage of queries in which the correct answer is retrieved by an exact k-NN search:

$$Nk = \frac{|C_k|}{|\mathcal{Q}_c|} \cdot 100 \quad \text{, where } \; C_k = \{q \in \mathcal{Q}_c,\; rank(R_q) \leq k\}$$

  Nk values range between 0% and 100%, where 100% means best effectiveness.

Correct copy detections

These measures evaluate the performance of detecting the copies in the query collection. Let S=$\{(\bar{c}, \bar{v}, s), ...\}$ be the output of the CBVCD system, and R be the ground-truth, i.e., R defines the actual copies between reference and query videos: R=$\{(\hat{c}, \hat{v}), ...\}$. In order to evaluate S, we calculate Precision (fraction of relevant documents which has been retrieved) and Recall (fraction of

the retrieved documents which is relevant) [Baeza-Yates and Ribeiro-Neto, 1999] using the following steps:

1. Sort triplets in S according to their confidence scores, and determine the minimum and maximum confidence scores s_{min} and s_{max}.
2. For every decision threshold t between s_{min} and s_{max}:
   (a) Define $S(t) = \{(\bar{c}, \bar{v}, s) \in S,\ s \geq t\}$.
   (b) For every triplet $(\bar{c}, \bar{v}, s) \in S(t)$: if $\exists\ (\hat{c}, \hat{v}) \in R$ where $\bar{c} \cap \hat{c} \neq \emptyset$ and $\bar{v} \cap \hat{v} \neq \emptyset$, then add $(\bar{c}, \bar{v}, s)$ to S_a, and add $(\hat{c}, \hat{v})$ to R_a.
   (c) Calculate Precision and Recall at decision threshold t:

$$\text{Precision}(t) = \frac{|S_a|}{|S(t)|} \qquad \text{Recall}(t) = \frac{|R_a|}{|R|}$$

Once precision and recall have been calculated at each threshold, we evaluate S according to:

- **Presicion/Recall curve**: The values for Precision(t) and Recall(t) can be plotted in an x-y plane. A system will outperform another when it achieves higher precision at every recall value. In particular, two relevant values to compare are:
  - **Maximum Recall with Precision 1** (R_{P1}): It compares systems by the amount of correct detections without false alarms.
  - **Maximum Recall with Precision greater of equal than 0.5** ($R_{P.5}$): It compares systems by the amount of correct detections, admitting an equal amount of correct answers and false alarms.
- **Detections without false alarms**. This is the absolute number of correct detections ($|R_a|$) at Precision 1. We prefer this value instead of R_{P1} because MUSCLE-VCD-2007 has a small number of copies (31), thus the absolute number may be more meaningful than its fractions. This value varies between 0 and 31. Table 5.2 shows the relationship between Detections without false alarms and Recall at Precision 1.

Indicator	**Conversion**										
Detections	0	1	2	3	4	5	6	7	8	9	10
Recall	0	0.03	0.06	0.10	0.13	0.16	0.19	0.23	0.26	0.29	0.32
Detections	11	12	13	14	15	16	17	18	19	20	21
Recall	0.35	0.39	0.42	0.45	0.48	0.52	0.55	0.58	0.61	0.65	0.68
Detections	22	23	24	25	26	27	28	29	30	31	
Recall	0.71	0.74	0.77	0.81	0.84	0.87	0.90	0.94	0.97	1	

Table 5.2 – Detections without false alarms at MUSCLE-VCD-2007 dataset and their associated Recall at Precision 1.

To the best of our knowledge, the work that shows the best detection performance on the MUSCLE-VCD-2007 dataset is Poullot et al. [2010], which achieves precision 1 and recall 1 for

ST1, and precision 1 and recall 0.95 for ST2, i.e., a total of 30 out of 31 correct detections without false alarms. Other works achieving high detection performance on this dataset are: Anguera et al. [2009b] achieving precision 1 and recall 1 for ST1, and precision 1 and recall 0.88 for ST2 (but considering 18 instead of 21 copies); Poullot et al. [2008] achieving precision 1 and recall 0.93 for ST1, and precision 1 and recall 0.86 for ST2.

5.7 Summary

In this chapter we have presented the big picture of this thesis and its five main components. This overview shows the relationships between each component and the evaluation procedure for each task.

The following chapters detail each task: Chapter 6 details the preprocessing task, Chapter 7 reviews the segmentation task and feature extraction task, Chapter 8 and Chapter 9 focus on the similarity search task, and Chapter 10 details the localization task.

Finally, Chapter 11 details the participation of this system at TRECVID 2010 and 2011 and its comparison with other state-of-the-art systems.

Chapter 6

Preprocessing

The Preprocessing task normalizes the quality of every video in the dataset and diminishes the effect of content transformations on query videos. This task analyzes the videos and creates a new dataset with the processed versions of the videos.

The new videos produced by this task are not physically created. The creation of a new video means that a source video has one or more filters applied to the actual video file. The parameters for the filters are calculated by this task and stored in a configuration file which is read by the following tasks. This approach avoids reencoding issues and wasting disk space, at the cost of an extra computational effort before reading each frame.

6.1 Quality normalization

The normalization process is performed for every video in the dataset, first in a frame-by-frame basis and then in a global-video basis.

6.1.1 Frame-by-frame normalization

For this normalization, every frame in the video is analyzed. A frame f_i is marked to be skipped if it is plain or it is an outlier. A frame f_i is skipped by duplicating the previous frame f_{i-1} when $i > 0$, or the next non-skipped frame when i=0.

In order to detect plain frames, a frame is converted into gray scale and down-scaled to 20×15 pixels. We state a frame is plain under any of these two conditions: the variance of the intensities is below a minimum value; or the difference between minimum and maximum intensity pixels is smaller than a threshold. These criteria aim to remove frames lacking enough information for being discriminative.

Once the plain frames have been removed, a detection of outlier frames is performed. We state a frame f_i is an outlier when two conditions are met: the previous frame f_{i-1} and the next frame f_{i+1} are similar; and frame f_i is very different from both f_{i-1} and f_{i+1}.

Let $\Delta(x, y)$ be the sums of differences for every pixel between frames x and y. The frames

are first down-scaled to 20×15, and f_i is an outlier when both $\Delta(f_{i-1}, f_i)$ and $\Delta(f_i, f_{i+1})$ are greater than a threshold and $\Delta(f_{i-1}, f_{i+1})$ is smaller than a threshold. This criterium aims to remove frames like flash-lights or other editing effects that may affect the video segmentation or the spatio-temporal feature extraction.

6.1.2 Global normalization

Once every frame has been normalized, the video is processed to remove letter-, pillar-, or window-boxing by cropping the black borders. To do this, the temporal median and variance of the intensity is calculated for each pixel. One row or column is removed iteratively from the borders to the center while the median and variance of the majority of the pixels is smaller than a threshold.

6.2 Detection and reversion of transformations

The detection process performs some test on a query video to decide whether or not the video has some specific transformation to revert. The reversion process creates one or more query videos with the reverted version of the detected transformation. The created query videos are added to the query collection together with the original query video. In this thesis, we focus on the reversion of three transformations: picture-in-picture (PIP), camcording, and vertical flip. The reversion processes have been performed only for TRECVID's query datasets.

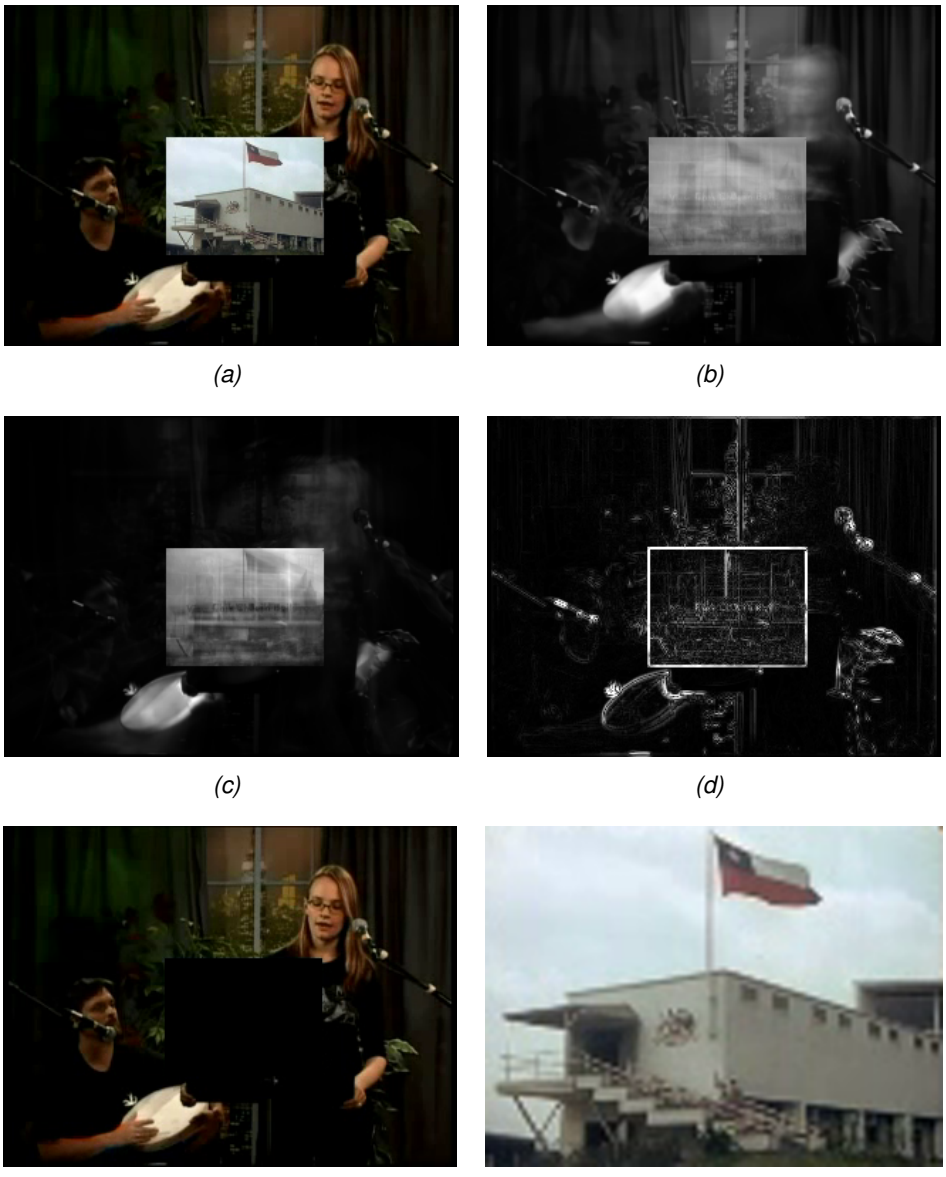
6.2.1 Picture-in-picture

The PIP creation process used in TRECVID's query dataset combines two video files. One video file is used as a background file and the other video file is embedded inside a static rectangle at any of the four corners or the center of the video. The size of the rectangle is fixed for the whole video, but it may vary between query videos. The reversion process looks for that persistent rectangle (see Figure 6.1).

First, all the frames in the video are processed to calculate the average intensity and the variance for every pixel. Because the two combined videos present different sequences, the edges of the PIP rectangle should be visible in either the average frame or the variance frame. The edges in these two frames are detected using the Laplacian kernel. We chose to apply the Laplacian kernel because it produces thinner edges (see Chapter 2). The "mean-edge" frame is the average of the two edges frames. The approach of averaging all the frames in a video is feasible because query videos contain only a few shots (in fact, the average length for query videos in TRECVID is about one minute).

Second, it detects corner candidates in the mean-edge frame through convolution between the frame and different ad-hoc masks (top-left, top-right, bottom-left, and bottom-right corner masks).

Third, it searches for rectangles by joining the detected corner in a valid spatial configuration according to the PIP creation process. The score of a detected rectangle is the sum of the value of its corners. The PIP is detected when the rectangle with the best score is greater than a threshold.



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 6.1 – PIP detection. (a) Query video from TRECVID 2010 dataset. (b) Average for each pixel. (c) Variance for each pixel. (d) The “mean-edge” frame. (e) New query with video at background. (f) New query with video at foreground.

Finally, the PIP reversion creates two new query videos: the foreground video (each frame cropped to the detected rectangle) and the background video (each frame with the detected rectangle filled with black pixels). The two new videos are then normalized.

6.2.2 Camcording

In TRECVID datasets, a simulated camcording transformation may be applying to query videos. This transformation intends to simulate a person recording a projection of the video on a big screen using a handheld camcorder. The transformation is implemented by: modifying a video by altering the gamma correction, applying a projective transform to each frame, and giving some random movements and distortions. Our reversion process searches for a moving wrapping quadrilateral on the query video (see Figure 6.2).

First, each frame in the video is binarized using a near-zero threshold. The center-of-mass is calculated for the frame which will be used as a content-relative reference point. Then, an "outer-edge" frame is created by selecting for each column and row in the binary image the nearest pixels to any border.

Second, with the center-of-mass and edge points, four Hough transforms are performed. Each transform focuses on locating an edge of the quadrilateral (top, bottom, left, and right margins) by restricting the position and slope in the parameter space. The four detected lines are then intersected to find the four vertices of the quadrilateral. The positions of these vertices are stored relative to the center-of-mass of the image.

Third, the four vertices are located for every frame in the video. If they are detected with a small variance in their relative position, then the average position of the four vertices define the wrapping quadrilateral.

Finally, the camcording reversion creates a new query video by calculating the center-of-mass for each frame and mapping the four vertices to the corners of the frame. The new reverted video is then normalized.

6.2.3 Vertical flip

There is no reliable method to detect whether a query video has a flip transformation or not. Thus, the detection process is skipped and the reversion always creates a new query video applying a vertical mirror to each frame. The flip reversion is applied to all the query videos, including the videos created by PIP and camcording reversions.

6.3 Summary

In this chapter we have depicted the processes performed by the Preprocessing task. This task intends to improve the quality of each video in order to increase the effectiveness of the copy detection. For MUSCLE-VCD-2007 dataset, we use the video normalization, while for TRECVID datasets, we use both video normalization and reversion of transformations.

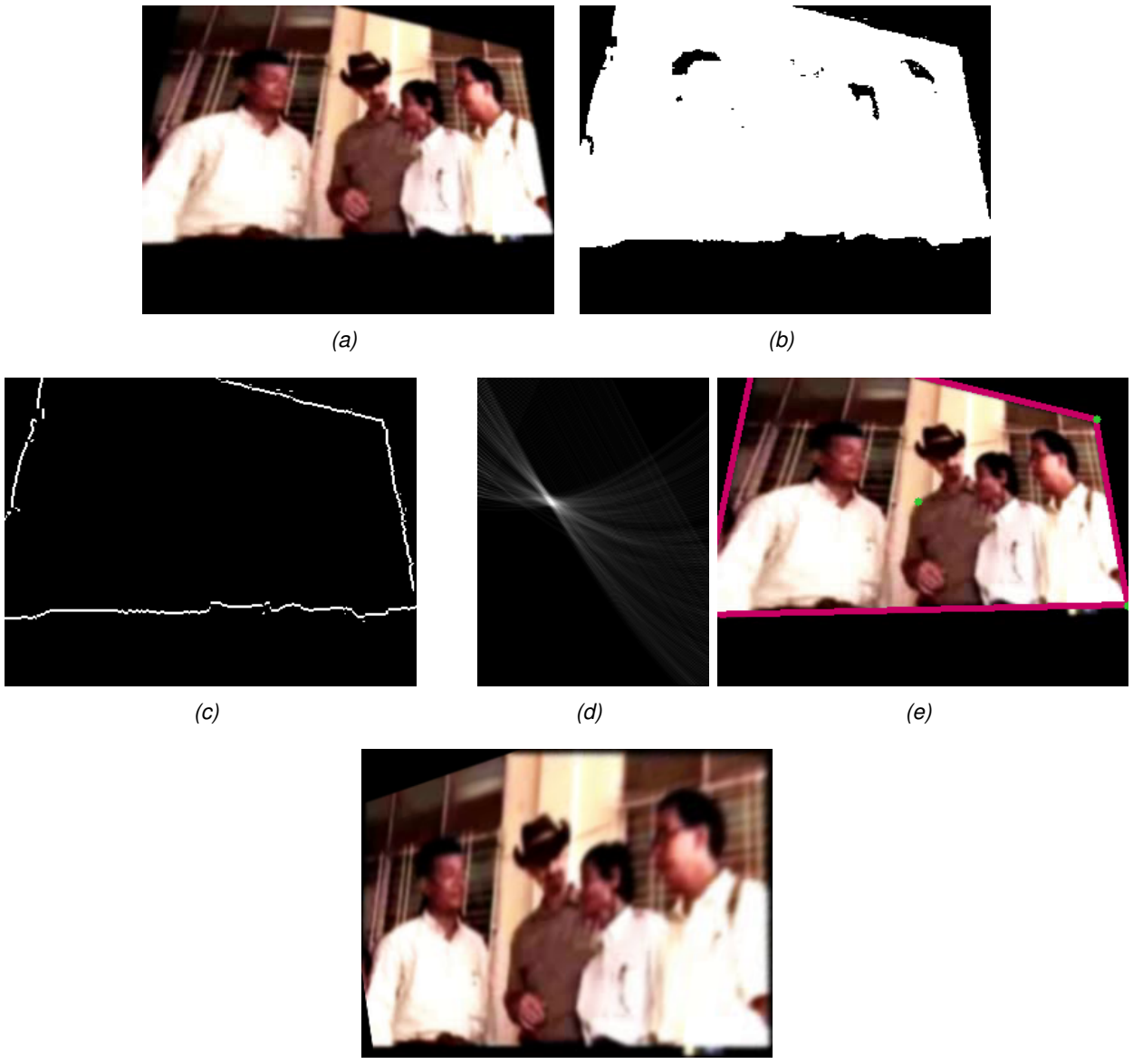


Figure 6.2 – Camcording detection. (a) Query video from TRECVID 2010 dataset. (b) Binarized frame. (c) The "outer-edge" frame. (d) The parameter space for one Hough transform (left edge). (e) Detected quadrilateral and the center-of-mass. (f) New query video.

The following chapter presents the Video Segmentation and Feature Extraction tasks, including an evaluation of the impact of video normalization on the detection effectiveness. Afterwards, in Chapter 11, we analyze the impact of reversion of PIP and camcording on detection effectiveness.

Chapter 7

Video Segmentation and Feature Extraction

The Video Segmentation task creates one or more segments from every video. The Feature Extraction task calculates one or more descriptions (global, local, acoustic) from each video segment. This chapter reviews these two tasks and evaluates the effectiveness of different configurations of segmentations and descriptions.

7.1 Video Segmentation

The Video Segmentation task partitions a video into groups of similar consecutive frames. The segments are not required to be the same length. Each segment may have a representative frame, which is used for extracting spatial descriptors.

Formally, given a video v with n frames $v=\{f_1, ..., f_n\}$, the segmentation task creates the set of segments $S=\{s_1, .., s_r\}$, where each segment s_i is a set of consecutive frames $s_i=\{f_j, f_{j+1}, f_{j+2}, ...\}$ for some starting frame f_j. We restrict segmentations to contain only non-overlapping segments, i.e., $\forall f \in v \; \exists! \, s \in S, \; f \in s$. Additionally, every segment $s \in S$ contains a unique representative frame f_s.

In this thesis we test fixed-length segmentation and variable-length segmentation.

7.1.1 Fixed-length segmentation

The fixed-length segmentation receives a target length t (in seconds), and it produces segments of either $\lfloor t \cdot fps \rfloor$ or $\lceil t \cdot fps \rceil$ consecutive frames, depending on which rounding produces the less accumulative error. For example, the segmentation of $\frac{1}{3}$ seconds on a video with 25 fps produces segments with lengths {8, 9, 8, 8, 9, 8, ...} with the sequence of accumulative errors {-0.$\bar{3}$, 0.$\bar{3}$, 0, -0.$\bar{3}$, 0.$\bar{3}$, 0, ...}. The representative frame for each segment is its middle frame.

In this chapter we evaluate segmentations **S1/5**, **S1/4**, **S1/3**, **S1/2**, **S1**, **S2**, **S3**, **S4**, and **S5**, which are fixed-length segmentations with target length t equal to $\frac{1}{5}$, $\frac{1}{4}$, $\frac{1}{3}$, $\frac{1}{2}$, 1, 2, 3, 4, and 5 seconds, respectively.

7.1.2 Variable-length segmentation

The variable-length segmentation is based on a fixed-length segmentation. It first performs a fixed-length segmentation and then it joins almost identical consecutive segments. The comparison between two segments is performed by global descriptors of their representative frames, thus it uses a feature extraction method (g, d) to determine whether two segments should be joined or not. A segment s_i is joined with previous segment s_{i-1} when the distance $d_{s_i} = d(g(f_{s_i}), g(f_{s_{i-1}}))$ is smaller than a threshold. The representative frame for the new segment is the representative frame of the segment s_i that achieved the minimum d_{s_i}.

Intuitively, this segmentation avoids producing many identical consecutive segments for static scenes, while keeping a dense segmentation for dynamic scenes. Almost identical segments will likely harm the localization of a copy due to their indistinguishable descriptors, hence this segmentation intends to benefit from dense fixed-length segmentations on dynamic content while preventing unnecessary segmentation of static content.

We used this type of partition for our participation at TRECVID 2010.

7.2 Spatial global description

In this thesis, we use six different feature extraction methods with different parameters, producing fourteen different spatial global descriptions. Chapter 2 gives details about the implementation of the extraction methods.

Given a video with segmentation $\{s_1, .., s_r\}$, the spatial-global description is a feature extraction method (g, d) which represents the visual content of a segment s_i by a single vector $g(f_{s_i})$ calculated from each representative frame f_{s_i}.

In order to reduce disk storage needs, if g produces vectors of floating point numbers, we perform a linear quantization to 8 bits to each value, therefore each dimension is an integer value between 0 and 255. During this chapter, d is fixed to L_1. Chapter 8 includes an evaluation of other distance functions.

Edge Histogram per zone (EH) This descriptor captures the orientation of edges. The extraction method converts a frame into gray scale and partitions it into $W \times H$ zones. Each zone is subdivided into $M_w \times M_h$ blocks and the edge orientation of each block is tested. The orientation is determined by calculating the energy of n different filters, and selecting the filter whose maximum exceeds a threshold t.

In this implementation we used M_w=M_h=8, t=5, and linear quantization to 8 bits/bin. We test the usage of five and ten orientation filters (see Figure 4.3 on page 46). In particular, we evaluate descriptors:

- **EH**4x4-10: W=H=4, ten orientation filters. It creates a vector of 160 dimensions. We used this descriptor at TRECVID 2010 and 2011.
- **EH**4x4-5: W=H=4, five orientation filters, the absolute value of the energy is compared against t. It creates a vector of 80 dimensions.

Gray Histograms per zone (GH) This descriptor captures the spatial distribution of intensities. The extraction method converts a frame into gray scale and divides it into $W \times H$ zones. For each zone a histogram of m bins is calculated. We test four descriptors:

- **GH**1x1-180: W=H=1, m=180. It creates a vector of 180 dimensions, invariant to vertical mirroring.
- **GH**1x4-32: W=1, H=4, m=32. It creates a vector of 128 dimensions, invariant to vertical mirroring.
- **GH**3x3-20: W=H=3, m=20. It creates a vector of 180 dimensions. We used this descriptor at TRECVID 2010.
- **GH**4x4-12: W=H=4, m=12,. It creates a vector of 192 dimensions. We used this descriptor at TRECVID 2011.

Independent-Color Histograms per zone (IH) This descriptor captures the spatial distribution of intensities for each channel. The extraction method divides a frame into $W \times H$ zones, and for each zone an independent histogram is calculated for each channel. The descriptor is the concatenation of three 1-d histograms, producing a vector with m_1+m_2+m_3 dimensions. We test three descriptors:

- **IH**1x4-rgb-3x16: RGB color space, W=1, H=4, m_R=m_G=m_B=16. It creates a vector of 192 dimensions, invariant to vertical mirroring.
- **IH**2x2-rgb-3x16: RGB color space, W=H=2, m_R=m_G=m_B=16. It creates a vector of 192 dimensions. We used this descriptor at TRECVID 2010.
- **IH**4x4-rgb-3x4: RGB color space, W=H=4,m_R=m_G=m_B=4. It creates a vector of 192 dimensions. We used this descriptor at TRECVID 2011.

Color Histograms per zone (CH) This descriptor captures the spatial distribution of colors. The extraction method divides a frame into $W \times H$ zones, and for each zone a color histogram is calculated using the RGB or HSV color space. The 3-d color space is divided uniformly into $m_1 \times m_2 \times m_3$ blocks, and the 3-d histogram is calculated. We test four descriptors:

- **CH**1x1-hsv-16x4x4: W=H=1, HSV color space, m_H=16, m_S=m_V=4. It creates a vector of 256 dimensions, invariant to vertical mirroring.
- **CH**2x2-hsv-16x2x2: W=H=2, HSV color space, m_H=16, m_S=m_V=2. It creates a vector of 256 dimensions.
- **CH**2x2-rgb-4x4x4: W=H=2, RGB color space, m_R=m_G=m_B=4. It creates a vector of 256 dimensions.

Reduced Keyframe (KF) This descriptor captures the spatial distribution of intensities. The extraction method converts a frame into gray scale and scales it down to $W \times H$ pixels. The descriptor is the vector of $W \cdot H$ dimensions whose values contain the pixel intensities. We test the descriptor:

- **KF**$_{11x9}$: W=11, H=9. It creates a vector of 99 dimensions.

Ordinal Measurement (OM) This descriptor captures the relative ordering of intensities (see Section 2.3.1). The extraction method converts a frame into gray scale and divides it into $W \times H$ zones. The descriptor is a permutation of $(1, ..., W \cdot H)$ with the rank of each zone after sorting them in ascending order by intensity (see Figure 2.9 on page 18). We test the descriptor:

- **OM**$_{9x9}$: W=H=9. It creates a vector of 81 dimensions.

7.3 Spatio-temporal global description

The spatio-temporal (s-t) description represents the whole sequence of frames in a segment instead of only its representative frame. The s-t extraction function uses a spatial extraction function to calculate the average of the spatial descriptors extracted for every frame in the segment. Formally, let (h, d) be a spatial feature extraction method which returns a fixed-length vector (the basal description), the s-t feature extraction method is the pair (g, d) where g is defined as:

$$g(s) = \frac{1}{|s|} \sum_{f \in s} h(f)$$

where s is a segment and $h(f)$ is either the description of frame f or the description of a segment which contains a unique frame f.

As we will show in the experimental section, this s-t descriptor can improve the effectiveness of the similarity search compared with its basal description. The cost for this improvement is higher computational effort at the feature extraction task, but it does not affect the dimensionality, the distance function, nor the disk space, hence it does not affect the similarity search task.

In the evaluation, we test the spatio-temporal descriptor for the fourteen spatial-global descriptors defined in previous section, denoting it with a ^{t} mark. For instance, **OM**$_{9x9}$ is calculated from the representative frame in the segment, while **OM**$^t_{9x9}$ is the average of all the **OM**$_{9x9}$ descriptors calculated from every frame in the segment.

In preliminary experiments we tested the effectiveness of extracting a descriptor from the average frame of a segment. This approach achieved very poor performance, and it was outperformed even by the spatial descriptor of the representative frame. This low performance is mainly due to average frames usually contain grayed colors and blurred edges. Therefore, we preferred the average descriptor extracted from each frame rather than the spatial descriptor extracted from the average frame.

7.4 Acoustic description

The acoustic description is based on Telefonica Research's implementation of the descriptor presented by Haitsma and Kalker [2002]. Originally, the descriptor is calculated with a FFT of the acoustic data every 10 ms over a sliding window of 32 ms. The frequency bins are then converted into a Mel scale of 16 bands, and a 15-bit fingerprint is calculated by comparing the energies of consecutive bands (1=increase, 0=decrease) (see Section 4.3.6).

The extraction method first resamples the audio track with a sampling rate of r-Hz, mono-channel and the descriptor is calculated for each audio segment. The extraction method uses a FFT for a sliding window of size w ms. The FFT coefficients are converted into a Mel scale of m bands, and the sum of energies of all the bands is normalized to 1. The window slides every s ms, and the energies for each band are averaged for every window inside the audio segment. We test the descriptor:

- **AU**160: audio track sampled to 8-kHz, FFT for a window size w=500 ms, shift s=20 ms, m=160 bands. It creates a vector of 160 dimensions 32 bits/dim (float). We used this descriptor at TRECVID 2011.

Unlike the original descriptor, where the search consists of retrieving collisions, the adapted audio descriptor can measure the degree of similarity between any two short audio signals. Note the similitude of this extraction method with the s-t global description. This property enables the distances between audio descriptors to behave in a roughly similar way as global descriptors, i.e., the histograms of distances present an alike shape which eases their combination.

In the experimental section, we test the different parameters of the audio descriptors. In particular, we compare the effectiveness of audio descriptors with window sizes 32 ms, 200 ms and 500 ms; window shifts every 10 ms and 20 ms; and number of bands 16, 80 and 160. This descriptor is compared using L_1 distance. Chapter 8 evaluates other distance functions for this descriptor.

7.5 Spatial Local description

The local description used in this thesis is SIFT. Instead of the SIFT reference implementation [Lowe, 2005], we preferred the implementation provided by the VLFeat library [Vedaldi and Fulkerson, 2008] due to its seamless integration with our system through its C API. We used VLFeat with default parameters which, according to its documentation, produces keypoints and descriptors very similar to the ones produced by the reference implementation.

The local description for a segment is the set of SIFT vectors extracted from the representative frame. To control the amount of vectors, we follow the reference implementation's approach, which scales down an input image to decrease the keypoint detections. This approach makes the tuning of library-dependent parameters unnecessary. We test descriptions:

- **SF**n: SIFT vectors extracted from the representative frame of the segment scaled down by a factor of n, where n varies between 1 and 6.

The local description for each segment is stored as a variable-size array containing vectors of 128 dimensions (1 byte/dim). The distance between any two segments compares their local descriptions and returns a dissimilarity value. Given segments q and r, we use two functions to compare them:

- Matches: It is the number of SIFT vectors in q that match a vector in r. Following the definition of Lowe [2004], a vector in q matches a vector in r if the ratio of the distance between its closest neighbor in r and the distance of its second closest neighbor is less that s. Formally:

$$\mathsf{Matches}_s(q,r) = 1 - |Pairs_s(q,r)|/|\mathbf{SF}(q)|$$

$$Pairs_s(q,r) = \left\{ (x,y) \in \mathbf{SF}(q) \times \mathbf{SF}(r),\ \forall z \in \mathbf{SF}(r) - \{y\}\ \frac{\mathsf{L}_2(x,y)}{\mathsf{L}_2(x,z)} \leq s \right\}$$

- Spatial: It is the size of the maximum subset of vectors in q that match a vector in r that are spatially coherent:

$$\mathsf{Spatial}_s(q,r) = \min_{\sigma,\tau} \{1 - |S_s^{\sigma,\tau}(q,r)|/|\mathbf{SF}(q)|\}$$

$$S_s^{\sigma,\tau}(q,r) = \{(x,y) \in Pairs_s(q,r), \mathsf{L}_2(x \cdot \sigma + \tau, y) \leq \varepsilon\}$$

  We use RANSAC to determine the maximum subset of matching vectors that satisfy the same scale and translation (see Chapter 2).

Lowe [2004] recommends the ratio s=0.8 which selects most of the correct matches and discards most of the incorrect ones.

7.6 Evaluation on MUSCLE-VCD-2007

The evaluation uses the MUSCLE-VCD-2007 dataset, with reference videos from DB-MPEG1 collection and query videos from ST1 and ST2 collections. As stated in Section 5.6, $\mathcal{R}$ is the set of reference segments, $\mathcal{Q}$ is the set of query segments, and $\mathcal{Q}_c$ is the set of query segments that have a correct answer in $\mathcal{R}$. In the following evaluations the similarity search is a linear scan.

7.6.1 Evaluation of global description

The evaluation consists of measuring the frame-to-frame effectiveness and the copy detection effectiveness. The frame-to-frame effectiveness is measured by the MAP and the copy detection effectiveness is measured by the amount of correct detections without false alarms.

The first experiment measures the basal effectiveness of each global descriptor calculated from the representative frame of the segment. These results correspond to the baseline for the following experiments. The second experiment measures the impact of preprocessing videos before calculating descriptors. Finally, the third experiment evaluates the impact of the s-t description on preprocessed videos.

Each query video and reference video is partitioned into segments of fixed length $\{\frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1, 2, 3, 4, 5\}$ seconds, and the effectiveness is measured for each segmentation length. Table 7.1 shows the number

	$\mathbf{S\frac{1}{5}}$	$\mathbf{S\frac{1}{4}}$	$\mathbf{S\frac{1}{3}}$	$\mathbf{S\frac{1}{2}}$	**S1**	**S2**	**S3**	**S4**	**S5**
Frames/segment †	5	6.25	8.33	12.5	25	50	75	100	125
Collection size (thousands of segments)									
$\mathcal{Q}$	69.7	55.7	41.8	27.9	13.9	7.0	4.7	3.5	2.8
$\mathcal{Q}_c$	49.9	39.9	29.9	20.0	10.0	5.0	3.3	2.5	2.0
$\mathcal{R}$	1,057.2	845.8	634.4	422.9	211.5	105.8	70.5	52.9	42.3

Table 7.1 – Number of query and reference segments for each fixed-length segmentation. † Fractional frames are resolved as described in Section 7.1.

of segments produced by each segmentation length. Because all the videos in the MUSCLE-VCD-2007 dataset have 25 fps, each segment contains between 5 frames and 125 frames.

To simplify the comparison we use the same L_1 distance (Manhattan distance) to compare any two descriptors:

$$\mathsf{L}_1((x_1, ..., x_n), (y_1, ..., y_n)) = \sum_{i=1}^{n} |x_i - y_i|$$

Although we could use a different metric for comparing each descriptor, we decided to use only L_1 in order to simplify the analysis. Moreover, L_1 is fast to evaluate, satisfies the metric properties, and achieves satisfactory effectiveness for all tested descriptors. In Chapter 8, we test the behavior of other distance functions.

Between the fourteen evaluated descriptors, only **GH**1x1-180, **GH**1x4-32, **IH**1x4-rgb-3x16, and **CH**1x1-hsv-16x4x4 are invariant to vertical mirroring.

Effectiveness of spatial description

This experiment measures the effectiveness of each global descriptor, when it is calculated from the middle frame of the segment. Figure 7.1 summarizes the effectiveness achieved by the fourteen spatial global descriptors for each segment size. In the case of frame-to-frame effectiveness, the MAP for every descriptor and segmentation varies between 0.2 and 0.5. Top performers are descriptors **GH**1x4-32 at $\mathbf{S\frac{1}{5}}$; **GH**3x3-20 at $\mathbf{S\frac{1}{4}}$, $\mathbf{S\frac{1}{3}}$, $\mathbf{S\frac{1}{2}}$, and **S1**; and **IH**4x4-rgb-3x4 at **S2**, **S3** (overall maximum with MAP=0.49), **S4**, and **S5**. On average, descriptors **GH**3x3-20 and **IH**1x4-rgb-3x16 are the top performers with an average MAP of 0.45 and 0.44 respectively, for the nine tested segmentations. On the other hand, **OM**9x9 is the descriptor that achieves the worst performance for every segmentation.

Analyzing the behavior of segmentation, the frame-to-frame effectiveness achieves a slight increase in coarser segmentations. On one hand, a coarse segmentation reduces the number of objects in $\mathcal{Q}$ and $\mathcal{R}$ which implies faster similarity searches. For instance, doubling the length of segments divides both the number of queries and the reference collection by a half, resulting in searches that can be up to four times faster. On the other hand, a coarse segmentation may affect the performance of the Copy Localization task due to the decrease in the query segments that match their original segment. For example, a copy excerpt of 4 seconds with $\mathbf{S\frac{1}{5}}$ seconds may match up to 20 consecutive query segments with their respective reference segment, but with **S2** it may match at most two query and reference segments.

The copy detection effectiveness shows the performance of the Copy Localization task for the different segmentations and descriptors. The graph shows the number of correct detections before the first false alarm is reported. The copies are determined by locating chains of reference segments between the first nearest neighbor of each query segment. The localization task is described in Chapter 10.

According to the ground-truth, MUSCLE-VCD-2007 has 31 copied excerpts. The top overall performance is achieved by **EH**4x4-10 with up to 27 correct detections without false alarms at **S1/4**, while the worst performers are **GH**1x1-180 (**S1/5**, **S1/4**, **S1/3**, **S1/2**, and **S1**) and **OM**9x9 (**S2**, **S3**, **S4**, and **S5**). The overall results show a clear inverse correlation between segmentation length and detection performance: the average number of detections for the evaluated descriptors at **S1/5** is 20.4, and it gradually decreases to 13.8 for **S5**. Therefore, even though a coarse segmentation may achieve a high frame-to-frame matching, the detection performance decreases due to the reduction of voter segments. Dense segmentations can achieve higher detection performance, however it should be noted that more voters may also produce false alarms with a higher score.

Effectiveness of video normalization

In this experiment we evaluate the influence of the preprocessing task on the effectiveness. Each video on the query and reference datasets is first preprocessed by the quality normalization (described in Section 6.1), i.e., outliers and plain frames are skipped and black borders are cropped. This experiment does not include any reversion, thus the sizes of $\mathcal{Q}$, $\mathcal{Q}_c$, and $\mathcal{R}$ do not vary. Figure 7.2 shows the effectiveness achieved by each descriptor when the videos have been preprocessed.

In the case of frame-to-frame effectiveness, for every descriptor and segmentation, the video normalization increases the effectiveness by an average of 16%. The effect of the quality normalization is bigger at dense segmentations (an average increase of 28% at **S1/5**) than at coarse segmentations (an average increase of 8% at **S4**). Descriptor **KF**11x9 achieves the largest increase (its MAP increased 84% on average for the nine segmentations), while descriptor **CH**2x2-hsv-16x2x2 was almost unaffected (2% average increase). The top performances are now achieved by **EH**4x4-10 at **S1/5**, **S1/4**, **S1/3**, **S1**, **S3** (overall maximum with MAP=0.54), and **S5**; **IH**1x4-rgb-3x16 at **S1/2**, and **S4**; and **IH**4x4-rgb-3x4 at **S2**.

In the case of detection effectiveness, the largest increase is achieved by **KF**11x9 with an average of 20%. The top overall performance is achieved by **EH**4x4-10 with 28 correct detections at **S1/5**, **S1/4** and **S1/3**, while the worst performers are again **GH**1x1-180 and **OM**9x9. In fact, **OM**9x9 is the only descriptor that mostly decreased its detection performance.

In summary, the video preprocessing increases the frame-to-frame effectiveness by an average of 16% for all descriptors and segmentations (ranging from 123% for **KF**11x9 at **S1/5** to -11% for **OM**9x9 at **S4**) and increases the detection effectiveness by an average of 9% (ranging from 45% for **KF**11x9 at **S5** to -38% for **OM**9x9 at **S1**).

Effectiveness of spatio-temporal description

This experiment tests the performance of the s-t descriptor. Figure 7.3 shows the effectiveness achieved by each descriptor when the videos have been normalized and the descriptors are extracted and averaged for every frame in the segment.

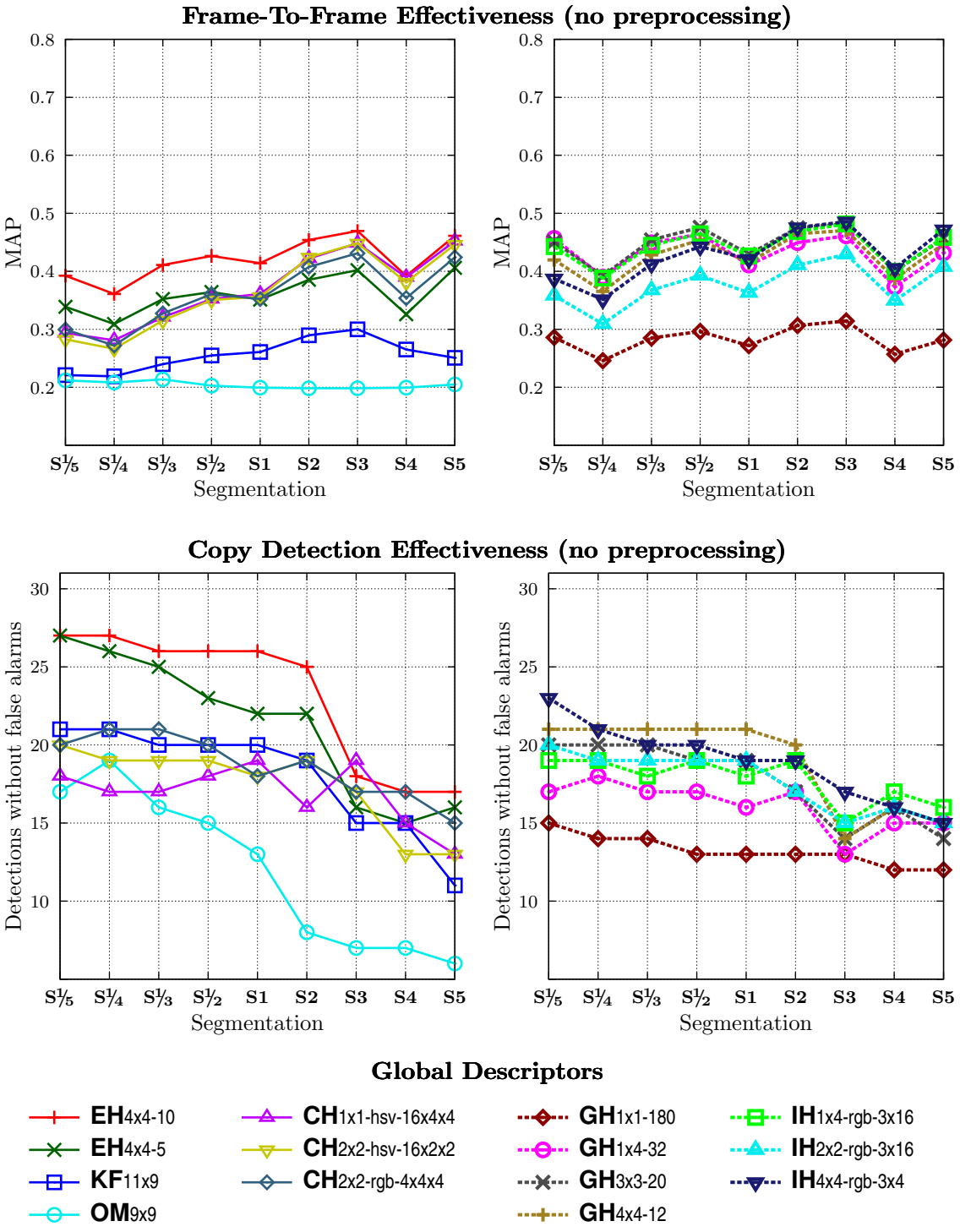


Figure 7.1 – Effectiveness of the fourteen spatial descriptors for different segmentation length without video preprocessing.

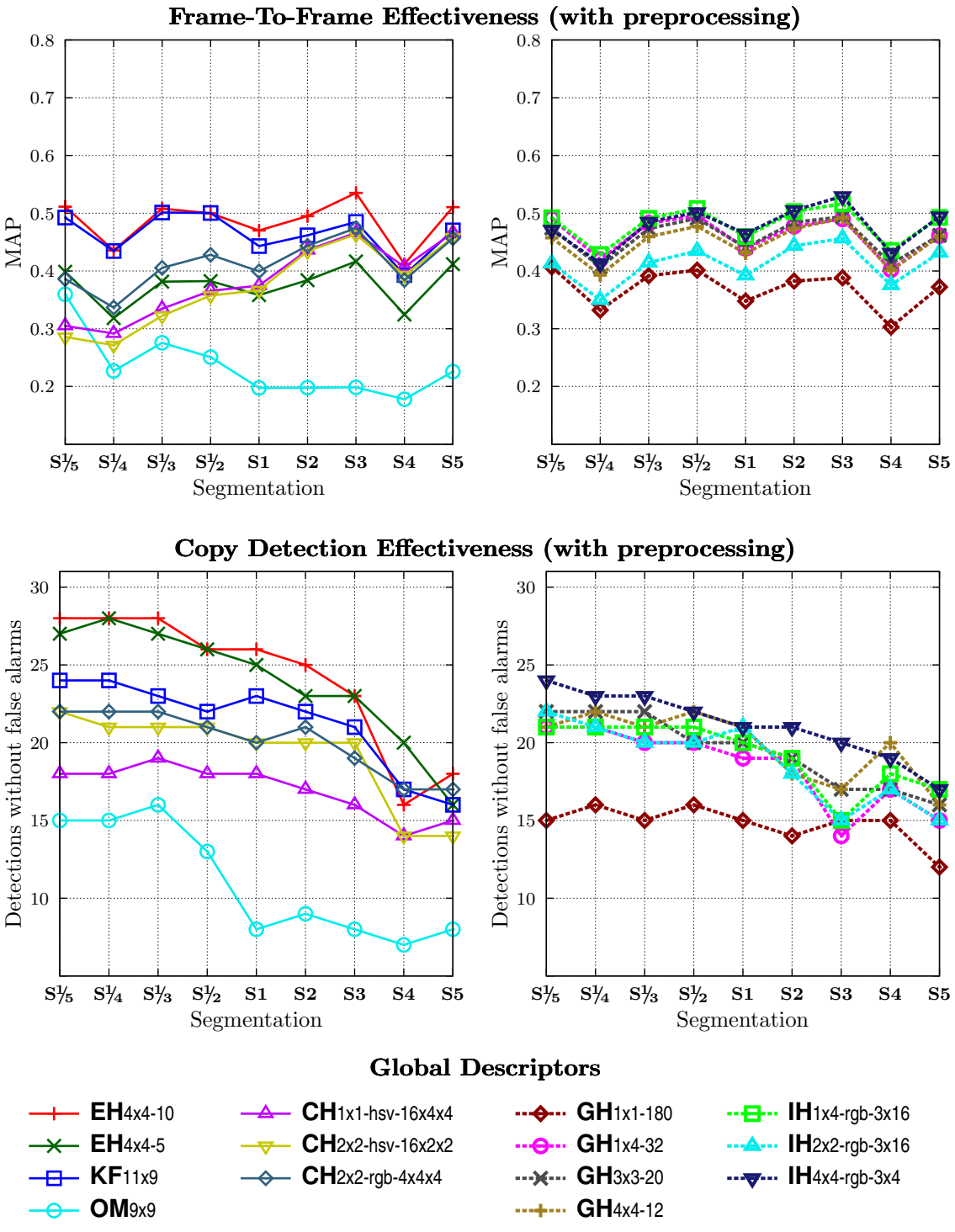


Figure 7.2 – Effectiveness of the fourteen spatial descriptors for preprocessed collections.

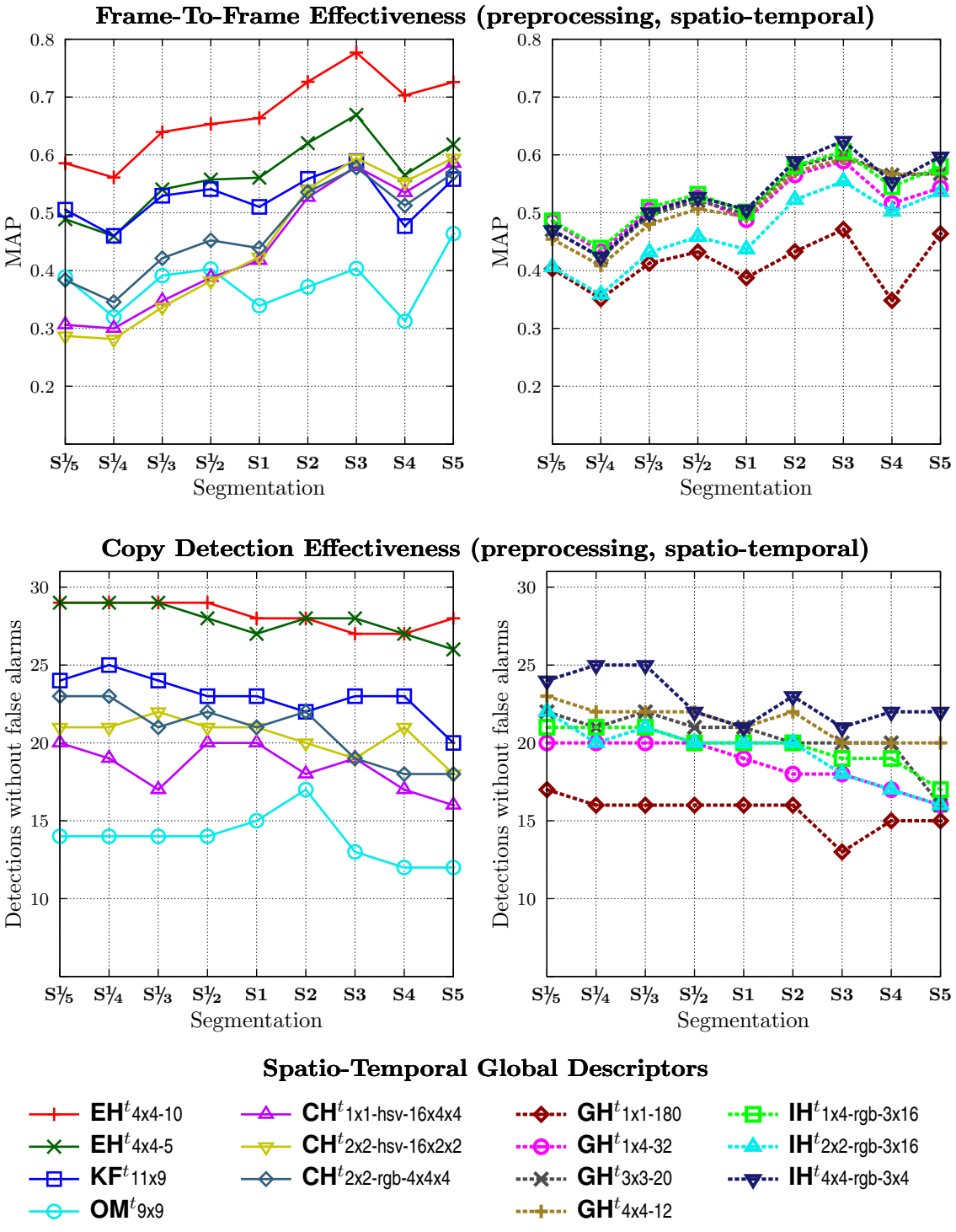


Figure 7.3 – Effectiveness of the fourteen spatio-temporal descriptors for preprocessed collections.

In the case of frame-to-frame effectiveness, the global increase of 22% in MAP compared with the previous experiment shows that the s-t description can improve the effectiveness of spatial descriptors. In particular, the **OM**^{t}9x9 achieves the highest increase in MAP (an average increase of 66% for the nine segmentations). On the other hand, **CH**^{t}2x2-hsv-16x2x2 was almost unaffected (an average increase of 2%). The top performer for every segmentation is the **EH**^{t}4x4-10 with an overall maximum of MAP=0.78 at **S3**. The worst performances are achieved by **OM**^{t}9x9, **CH**^{t}2x2-hsv-16x2x2 and **GH**^{t}1x1-180.

In the case of detection effectiveness, the performance increases by 10% on average for every descriptor and segmentation compared with the previous experiment. The largest average increase is for **OM**^{t}9x9 with a 38%, however it is still the worst performer. The top performers are **EH**^{t}4x4-10 with 29 detections between **S1/5** and **S1/2**, and also **EH**^{t}4x4-5 with 29 detections, but between **S1/5** and **S1/3**.

In summary, the spatio-temporal description increases the frame-to-frame effectiveness by an average of 22% for all descriptors and segmentations (ranging from 106% for **OM**^{t}9x9 at **S5** to -2% for **IH**^{t}2x2-rgb-3x16 at **S1/5**) and increases the detection effectiveness by an average of 10% (ranging from 89% for **OM**^{t}9x9 at **S2** to -13% for **GH**^{t}1x1-180 at **S3**), when they are compared with the previous experiment.

Discussion

Analyzing the results for copy detection at the s-t description, the overall top performers are the two **EH** descriptors, with the 10-orientations histogram outperforming the 5-orientations. This reveals that the orientation of edges is the feature that best represents the visual information for CBVCD, and also that they are more robust to content transformations.

An interesting result is that **EH**^{t}4x4-10 at **S1/2** detects correctly 29 of 31 copies without producing any false alarm, i.e., it achieves recall 0.94 at precision 1. In particular, for ST1 it achieves recall 1, and recall 0.90 for ST2. Despite **EH**^{t}4x4-10 not being invariant to mirroring, the system can detect the flipped copy from ST1 by just matching the almost symmetrical frames. This result outperforms most of the state-of-the-art systems evaluated with MUSCLE-VCD-2007 dataset, like a complex system based on local descriptor [Poullot et al., 2008] and a system combining visual with audio descriptors [Anguera et al., 2009b]. Moreover, **EH**^{t}4x4-5 is the descriptor with lowest dimensionality (80-d) and it also achieves this detection performance at **S1/3**.

The following best performer is **KF**^{t}11x9. In the baseline, this descriptor does not achieve very good results, but the two processes (quality normalization and s-t description) highly improved its detection performance. In fact, the **KF** is a really naive descriptor (it is just a reduction of the frame), thus its good detection performance is somewhat surprising given its low dimensionality (99-d) compared with color histograms.

The remaining best performer is **IH**^{t}4x4-rgb-3x4. This descriptor (with 192-d) outperforms **CH**^{t}2x2-rgb-4x4x4 (with 256-d) mainly because **IH** uses more zones, which in turn is due to **IH** needing less dimensions to represent a zone (just 12-d per zone). This issue is more evident if we compare the results for two descriptors using the same number of zones: **CH**^{t}2x2-rgb-4x4x4 slightly outperforms **IH**^{t}2x2-rgb-3x16. The **IH** descriptor calculates independent histograms for each channel, therefore it does not capture the actual colors in a zone. For instance, an **IH** descriptor with high values for the R channel may be created by zones with colors red, yellow, magenta, or white. This problem can be reduced by using a finer zoning, which turns out to be more relevant for locating duplicates. Note

also that this simplified color model is able to outperform $\mathbf{GH}^{t}_{\text{4x4-12}}$, which uses the same number of dimensions by zone.

Comparing color spaces, the performances of $\mathbf{CH}^{t}_{\text{2x2-rgb-4x4x4}}$ and $\mathbf{CH}^{t}_{\text{2x2-hsv-16x2x2}}$ are mostly similar (both use 64-d per zone) with a slight gain for the RGB color space. Therefore, for locating copies, a color descriptor based on RGB color space can show similar and even better performance than HSV color space for the same number of dimensions.

Between the descriptors invariant to vertical mirroring, $\mathbf{IH}^{t}_{\text{1x4-rgb-3x16}}$ achieves the best performance. Moreover, it also outperforms $\mathbf{IH}^{t}_{\text{2x2-rgb-3x16}}$, showing that the division in horizontal slides can be a better choice for datasets with vertical mirroring. However, it was outperformed by $\mathbf{IH}^{t}_{\text{4x4-rgb-3x4}}$ which have the same dimensionality, thus a good strategy for designing a descriptor is to use more zones, but given a fixed number of zones, the horizontal division is better.

The $\mathbf{OM}^{t}_{\text{9x9}}$ is the descriptor that achieves the worst performance for every segmentation. This result might seem contradictory with the evaluation presented by Kim and Vasudev [2005], however their work focused on detecting low-intensity quality transformations (i.e., video duplicates). Ordinal Measurement is highly affected by partial changes that can convert a dark zone into a bright one (like the insertion of logos), changing the ordering of zones and affecting (and ruining) the whole descriptor. This unsatisfactory performance of **OM** in CBVCD is also verified by Law-To et al. [2007a]. The other evaluated descriptors also divide the frames, but unlike **OM**, a change in one zone does not affect the values for the other zones.

With respect to the two tested improvements, the video normalization increases the frame-to-frame and detection performance by an average of 16% and 9%, respectively, for every descriptor and segmentation. The s-t description also increases them by an average of 22% and 10% compared with the video normalization alone. Compared to the baseline, these two combined processes increase the frame-to-frame effectiveness by 39% on average, and the detection effectiveness by 20%. A synergy exists between these two processes given that the quality normalization removes noisy and outlier frames which may harm the s-t descriptor.

7.6.2 Effectiveness of acoustic description

This experiment measures the effectiveness of the acoustic description using different values for parameters m (number of bands), w (window size), and s (window shift). In particular, we compare the effectiveness of audio descriptors using three numbers of bands (16, 80 and 160), three window sizes (32 ms, 200 ms and 500 ms), and two window shifts (10 ms and 20 ms). As in the previous experiments, we perform the comparison for the frame-to-frame matching and copy detection performance using the nine segmentations ($\mathbf{S1/5}$ to $\mathbf{S5}$). Figure 7.4 shows the results.

In the case of frame-to-frame effectiveness, the performance is highly dependent of the number of bands m. On average, the MAP increases by 24% between 16 and 80 bands, and by 28% between 16 and 160 bands. However, the increase is reduced for the densest segmentation ($\mathbf{S1/5}$): 4% between 16 and 80 bands, and 6% between 16 and 160 bands. A larger window size also increases the effectiveness. On average, the MAP increases by 7% from 32 ms to 200 ms, and by 11% from 32 ms to 500 ms. Finally, the shift s does not have a clear impact as the MAP for 10 ms and 20 ms are almost similar. The overall maximum MAP 0.84 was achieved by m=160 and w=500 ms at $\mathbf{S1/5}$.

In the case of detection effectiveness, a larger m implies more detections. On average, the

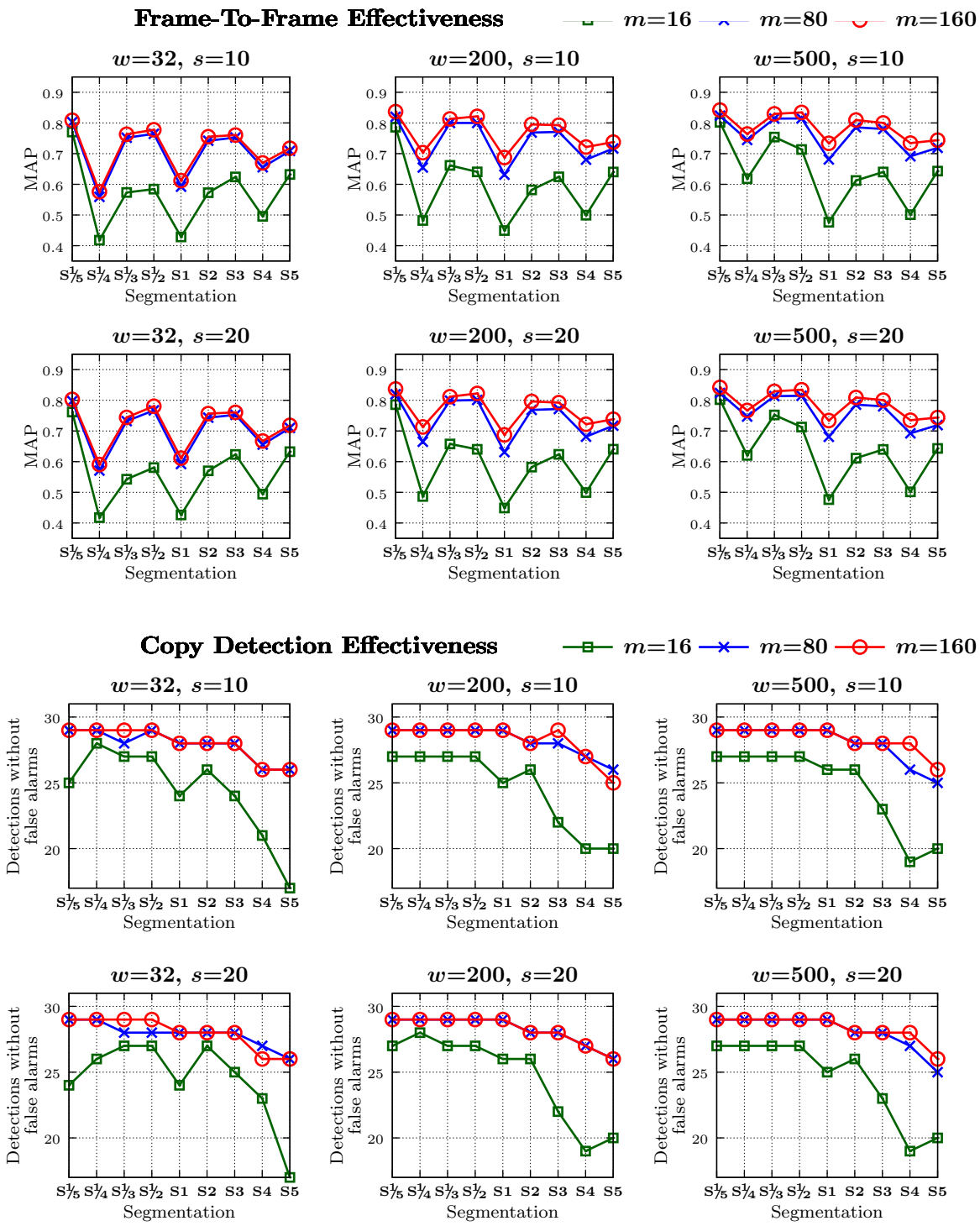


Figure 7.4 – Effectiveness of different parameters for the acoustic descriptor.

	SF1	SF2	SF3	SF4	SF5	SF6
Average vectors/segment						
Query	498.8	151.1	75.5	46.4	31.4	23.1
Reference	579.3	155.4	78.5	48.9	33.3	24.6
Input frame size	352×288	176×144	117×96	88×72	70×58	59×48
Average number of pairs	288,945	23,475	5,927	2,269	1,047	569

Table 7.2 – Average number of vectors per segment, size of frame from which vectors are extracted, and number of pairs between query and reference vectors.

number of detection increases by 15.6% between 16 and 80 bands, and by 16.2% between 16 and 160 bands. However, there is almost no improvement between using 80 and 160 bands. A larger window size increases the effectiveness by 18% for 16 bands between 32 ms and 200 ms, but it has almost no influence for 80 and 160 bands. Finally, the shift s does not have a clear impact on descriptor effectiveness. The overall maximum is 29 correct detections without false alarms (out of 31) achieved between $\mathbf{S}^{\mathbf{1}}\!/_{\mathbf{5}}$ and **S1** for $m \geq 80$ and $w \geq 200$ ms. As discussed in a previous experiment, this detection performance (recall 0.94 at precision 1) outperforms most of the CBVCD systems evaluated with this dataset, in particular a system based on audio descriptors [Anguera et al., 2009b].

The most relevant parameter for the acoustic descriptor is the number of bands: 16 bands are too few, but 80 and 160 are enough to represent the audio. The two copy excerpts that could not be detected are a camcording transformation in which the audio track is almost inaudible, and a copy that completely lacks the audio track. The almost inaudible track may become detectable by increasing the base audio sampling rate (instead of 8-kHz), or with some audio preprocessing (like a volume normalization).

In summary, this experiment proves the metric space approach can also be applied to acoustic descriptors. In fact, the similarity search using the proposed acoustic descriptor shows an alike behavior and performance compared to the similarity search using visual descriptors. Therefore, it seems feasible to design a distance function that successfully combines acoustic and visual descriptors. We explore this idea in the next chapter.

7.6.3 Evaluation of local descriptors

The next experiment measures the base effectiveness of local description. We perform the evaluation for frame-to-frame matching and copy detection performance using different frame resolutions and segmentations in preprocessed videos.

Table 7.2 summarizes the average number of SIFT vectors extracted from the representative frame of each segment. On average, query segments have less vectors than reference segments due to the effect of visual transformations. The number of pairs between query and reference vectors indicates the expected evaluations of L_2 to measure the dissimilarity between any two segments. This number is directly proportional to the overtime spent by a search based on local description compared with a search based on global description (which require just one distance evaluation to compare two segments). For instance, the time spent by a search using **SF1** will be five orders of magnitude slower than the time spent by a search using global description (even not considering any spatial coherence process). This is an unaffordable amount of time, hence in the following experiment we test the effectiveness of **SF4**, **SF5**, **SF6** for segmentations **S2**, **S3**, **S4** and **S5** (the

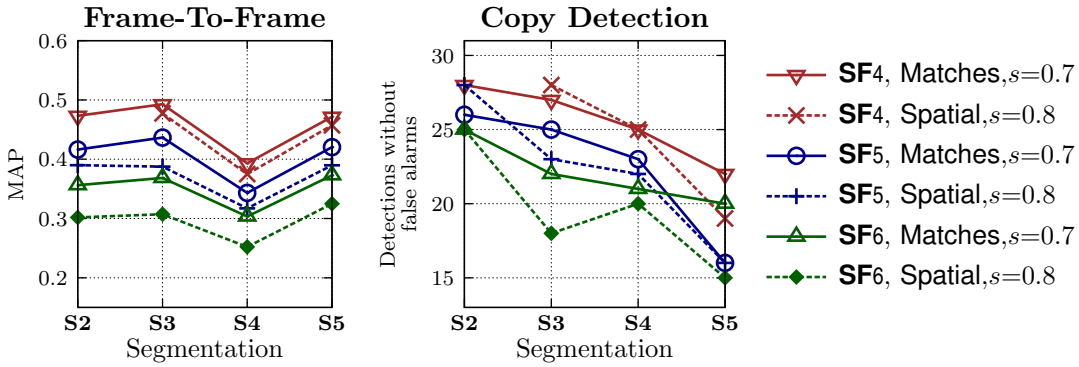


Figure 7.5 – Effectiveness of Matches and Spatial for local descriptions **SF**4, **SF**5, and **SF**6.

number of segments produced by each segmentation is already shown in Table 7.1). The search will be a linear scan comparing each query segment to every reference segment using distances Matches and Spatial, described in Section 7.5.

Figure 7.5 shows the effectiveness achieved by each local description and distance when the videos have been normalized and local description calculated from the middle frame in the segment. We compare distances Spatial with s=0.8, and Matches with a stricter ratio s=0.7, due to its lack of spatial coherence.

The MAP is directly affected by the size of the frames: **SF**4 outperforms **SF**5, which in turn outperforms **SF**6. In the case of distances, Matches consistently outperforms Spatial. This behavior is due to the reduction of the frames: Spatial misses frames when the number of matching vectors is not enough to perform a spatial coherence process. The gap between Matches and Spatial increases when there are less vectors per frames, i.e., the gap using **SF**6 is higher than using **SF**4.

In detection effectiveness, there is also a gap between Matches and Spatial, but it is reduced with denser segmentations and larger frames. In fact, Spatial outperforms Matches from **S3** using **SF**4, and from **S2** using **SF**5.

Comparing these results with the performance of a spatial global description (Figure 7.2), **SF**4 achieves a relatively similar MAP as global descriptions at the same segmentation, but achieves higher detection effectiveness. In fact, **SF**4 detects 28 of 31 copies at **S2**, but **EH**4x4-10 requires **S**$\frac{1}{3}$ or denser to achieve the same result. On the other hand, the search using **SF**4 on **S2** is expected to be about fifty times slower than the search using **EH**4x4-10 on **S**$\frac{1}{3}$.

SIFT vectors capture the orientations of the gradient, which is relatively similar to the information that captures the **EH** descriptor. Therefore, the high detection performance achieved by SIFT vectors is consistent with our previous conclusion that the orientation of edges is the feature that best represents the visual information for CBVCD.

To reduce the search time, the approach of the visual codebook is widely preferred. The codebook approach moves the cost associated with large amount of local vectors from the online phase to the offline phase. The description of each frame only uses a set of representative vectors, which are calculated by clustering local vectors. The search becomes a simple detection of collisions (which is very fast compared to Matches because it does not require any distance evaluation), but it requires a spatial coherence process (like the one in Spatial) to reduce the false positives. In Chapter 9 we present an alternative approach for reducing the search time for local description

based on approximate searches between local vectors.

7.7 Summary

In this chapter we have presented and evaluated the effectiveness of different descriptions under different segmentations.

In the case of global description, we evaluate fourteen different descriptors under nine segmentations. We also test the impact of the video normalization process and the spatio-temporal description. Both processes highly improved the performance of every descriptor. These increases convert rather simple global descriptors into surprisingly well-performing descriptors that can compete with state-of-the-art systems based on local descriptors.

In the case of acoustic description, we present an acoustic descriptor designed for using the metric space approach. We tested its parameters and we showed that it can achieve high detection performance.

In the case of local description, we evaluate SIFT vectors extracted for a reduced frame. The experiments show that local description can achieve a high performance, but its major drawback is the increase in search time. To reduce the search time, the approach of the visual codebook is usually preferred, however in Chapter 9 we present an alternative approach for reducing the search time.

In particular, the configurations that achieved high detection effectiveness in the experimental section are:

- Global descriptions $\mathbf{EH}^{t}_{\text{4x4-10}}$, $\mathbf{KF}^{t}_{\text{11x9}}$, and $\mathbf{IH}^{t}_{\text{4x4-rgb-3x4}}$, with segmentation $\mathbf{S}^{1}\!/_{3}$ or $\mathbf{S}^{1}\!/_{2}$, and distance L_1.
- Acoustic description **AU** with parameters m=80 bands, w=500 ms, and s=20 ms, with segmentation **S1**, and distance L_1.
- Local description $\mathbf{SF}_{4}$, with segmentation **S3**, and distance $\mathsf{Matches}_{s=0.7}$.

In the following chapter we focus on improving the effectiveness of these descriptors using their combination and using different distance functions. Afterwards, in Chapter 9 we focus on improving the efficiency of the search using an index structure and performing approximate searches instead of linear scans.

Chapter 8

Improving Effectiveness in the Similarity Search

The objective of the Similarity Search is to perform NN+range searches to retrieve the k most similar reference segments at a distance threshold ϵ for each segment $q \in \mathcal{Q}$ according to a distance function d.

This chapter focuses on improving the effectiveness of the search by using different definitions of d. In particular, we test: the linear combination of distances, the spatio-temporal combination of distances, and the use of non-metric distances. For these approaches we assume the descriptors are vectors with known dimensionality that represent a whole segment.

For this chapter, the similarity search is implemented as a linear scan. The next chapter focuses on increasing the effectiveness of the search.

8.1 Spatial-Combined Distance

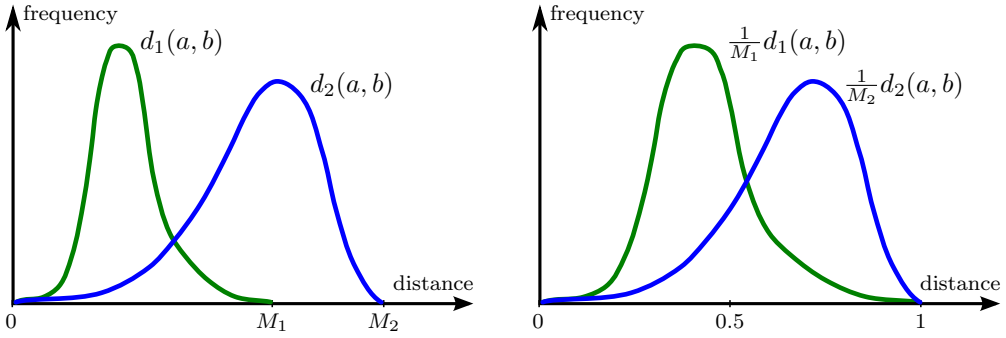
Let m be the number of descriptors for each segment, let $g_i(s)$ be the i^{th} descriptor for segment s, and let d_i be the distance for comparing the i^{th} descriptor, $i \in \{1, ..., m\}$.

Definition 1 (Spatial-Combined Distance) *The spatial-combined distance γ between two segments q and r is defined as a weighted combination of the distances between its descriptors:*

$$\gamma(q, r) = \sum_{i=1}^{m} w_i \cdot d_i(g_i(q), g_i(r)) \tag{8.1}$$

where $w_i \in [0, 1]$ and $\sum_{i=1}^{m} w_i$=1.

If the underlying distances in γ satisfy the metric properties and the weights are static, then the spatial-combined distance will also satisfy the metric properties (see Section 3.3.2).



(a) Maximum distances M_1 and M_2. *(b) Normalization according to M_1 and M_2.*

Figure 8.1 – Normalization by maximum distances on two functions d_1 and d_2.

8.1.1 α-Normalization of distances

The first issue that each weight w_i should solve is that of scaling the values returned by each underlying distance d_i to a range in which they are comparable. These weights are usually fixed to normalize each distance by its maximum value.

The normalization by maximum distance sets weights $w_i{=}\frac{1}{M_i}$, where M_i is the maximum distance that d_i returns. This normalization scales all distances to a bounded value in the range [0,1] in order to enable their combination. However, this approach does not reflect the distribution of values for each function, i.e., for some functions the distance threshold 0.5 could be a very permissive value (a range search selects many objects) while for others it could be very restrictive (a range search selects just a few objects). An example is shown in Figure 8.1.

To address this issue, we set each weight w_i using the histogram of distances of d_i. Because the area of the histogram of distances is normalized to 1, the histogram can be seen as a probability distribution of d_i. Then, we define the cumulative distribution in a similar way as probabilities.

Definition 2 (Cumulative Distribution) *Let d be a distance function, and let H_d be its histogram of distances. The Cumulative Distribution of Distances $F_d : \mathbb{R}^+ \to [0,1]$ is defined as:*

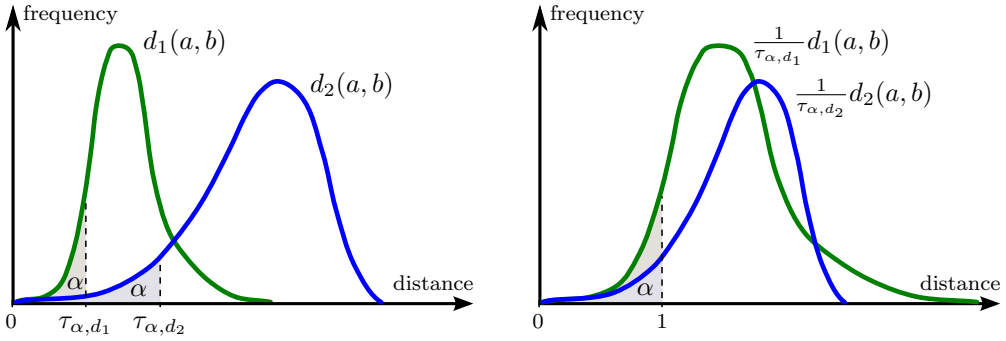
$$F_d(x) = \int_0^x H_d(t)dt$$

Additionally, $F_d^{-1} : [0,1] \to \mathbb{R}^+$ is the inverse function of F_d.

We state that two distance values are comparable when they have the same selectiveness according to their respective distributions, i.e., for two distance functions d_1 and d_2 the values $d_1(x)$ and $d_2(y)$ are comparable when $F_{d_1}(x) \approx F_{d_2}(y)$.

With the objective of scaling distance functions for making their values comparable for the nearest neighbors, we define the α-normalization.

Definition 3 (α-Normalization) *Let d be a distance function, and let α be a number in (0,1], the α-normalization of d defines the normalized distance d^α as:*



(a) Distance threshold τ_α for both distances. *(b) Normalization according to each τ_α.*

Figure 8.2 – The α-normalization of two functions d_1 and d_2.

$$d^\alpha(x,y) = \frac{1}{\tau_{\alpha,d}} d(x,y) \quad , \quad \text{where} \quad \tau_{\alpha,d} = F_d^{-1}(\alpha) \; .$$

Because γ is used to perform NN+range searches, all of its underlying distances should be α-normalized with a small α. Then, the smaller values for the underlying distances will be comparable between them. As a general rule, a value between 0.1 and $\frac{1}{|\mathcal{R}|}$ is usually good enough. Note that α=1 implies the normalization by maximum distance. Note also that α-normalization with the same α for different distance functions should imply a different weight for each one. Figure 8.2 shows an example of the α-normalization of two distances.

The α-normalization of the underlying distances of γ tries to select weights that scale the distributions in order to make the smaller distances comparable between them. Because histograms of distances may have different shapes and slopes, the distances become comparable only in a neighborhood of τ_α. To make underlying distances comparable in their whole range, a variable weight is needed. However, in that case, the triangle inequality will not be satisfied.

Even when underlying distances are comparable, depending on the actual descriptors and the dataset, it may be better to increase or decrease some weight on the combination. Hence, the second issue that weights should support for improving effectiveness is to give more importance to better descriptors. The following sections present two weighting strategies.

8.1.2 Weighting by Max-ρ

The motivation for this algorithm comes from our experimentation regarding efficiency and pivots. In general, we have noticed that very simple descriptors which produce search spaces with low intrinsic dimensionality (i.e., the search space can easily be indexed with a few pivots) usually yield low effectiveness. However, as we use more complex descriptors and/or distances, the increase in the effectiveness is usually bound to an increase in the intrinsic dimensionality (pivots become less efficient in the search). This tradeoff between effectiveness and efficiency motivated us to define a weighting algorithm that tries to improve the effectiveness by decreasing the efficiency.

Definition 4 (Weighting by Max-ρ) *Let γ be a convex combination of the α-normalized distances $\{d_1^\alpha, ..., d_m^\alpha\}$, the Weighting by Max-$\rho$ selects the weights $\{w_1, ..., w_m\}$ that maximizes the intrinsic dimensionality of γ.*

In other words, given the histogram of distances H_γ, the weighting by max-ρ selects the weights that solve:

$$\max_{\{w_1,...,w_m\}} \left\{ \frac{\mu(H_\gamma)^2}{\sigma^2(H_\gamma)} \right\}$$

A search space with high intrinsic dimensionality clearly does not imply a high effectiveness. This can be proved with a distance that returns a constant value: its intrinsic dimensionality will be infinity but its effectiveness will be zero. On the other hand, a perfectly discriminant distance which gives a small distance to correct matches and a large distance to every irrelevant object would achieve the maximum effectiveness and a near-infinite intrinsic dimensionality.

Despite these counterexamples, the weighting by max-ρ assumes that if we start from a point (a set of weights) with medium effectiveness, then a close point that decreases the efficiency should imply that it is a point where effectiveness increases. There may be sets of weights that result in better effectiveness and in lower intrinsic dimensionality, but to recognize them an evaluation of effectiveness should be required.

8.1.3 Weighting by Max-τ

The similarity search retrieves only the nearest objects to the query, discarding the highest distances. The weighting by max-ρ uses an indicator which is influenced by the whole distribution of distances, but the effectiveness is only influenced by the smaller distances. This issue motivates the definition of the weighting by max-τ, which tries to improve effectiveness by using an indicator that depends only on the distribution of the smaller distances.

The α-normalization scales each underlying distance d_i with a weight that for randomly-selected objects x,y, $\mathbb{P}[d_i^\alpha(x,y) \leq 1] = \alpha$. Because γ is a convex combination of α-normalized distances, it might be expected that for randomly-selected objects x,y $\mathbb{P}[\gamma(x,y) \leq 1] = \alpha$. If all the underlying distances were independent, this statement would be true. However, in general, this statement is not true, because the same pair (x,y) chosen to evaluate γ will be used to evaluate each of its underlying distances. If descriptors i and j are highly correlated, then the conditional probability increases $\mathbb{P}[d_j^\alpha(x,y) \leq 1 \mid d_i^\alpha(x,y) \leq 1] > \alpha$. For example, assume d_1^α compares segments by their RGB histogram and d_2^α compares them by their HSV histogram. When two segments x and y are similar according to their RGB histogram ($d_1^\alpha(x,y) \leq 1$), the probability of being similar according to their HSV histogram will increase ($\mathbb{P}[d_2^\alpha(x,y) \leq 1] > \alpha$). In that case, $\gamma(x,y)$ will be less than 1 with a probability higher than α, hence the value of $\tau_{\alpha,\gamma}$ (i.e., the value that α-normalizes γ) will be smaller than 1. Due to this issue, we define the following weighting method.

Definition 5 (Weighting by Max-τ) *Let γ be a convex combination of the α-normalized distances $\{d_1^\alpha, ..., d_m^\alpha\}$, the Weighting by Max-$\tau$ selects the weights $\{w_1, ..., w_m\}$ that maximizes $\tau_{\alpha,\gamma}$, where $\tau_{\alpha,\gamma}$=$F_\gamma^{-1}(\alpha)$.*

In other words, the weighting by max-τ maximizes the value that α-normalizes γ with the objective of favoring uncorrelated descriptors.

8.1.4 Discussion and implementation details

The weighting by max-ρ favors distances that produce values globally closer to μ (in particular the small values). The maximization of τ favors distances whose small values are larger. Graphically, this can be seen as choosing a function whose histogram of distances is sharper in the near zero zone. The intuition behind these two algorithms is to choose functions that increase the isolation of the nearest neighbors by the spreading small distances in a wider range.

For the process of maximization of ρ or $\tau_{\alpha,\gamma}$, the Newton-Raphson method can be used. However, we use a simpler approach: given an indicator (ρ or $\tau_{\alpha,\gamma}$), each weight w_i is initialized to $\frac{1}{m}$ (for m underlying distances), and then iteratively w_i is replaced by $w_i \pm \varepsilon$ if that change increases the desired indicator, and ends when every weight has been tested and none updated.

We should stress that these algorithms depend on the histogram of distances which is created by sampling pairs of objects (not evaluating every possible pair). Depending on the distance function, the evaluation of a statistically-relevant number of distances should take from a few seconds up to a few minutes. Thus, the computational time required for the automatic normalization and weighting processes is practically negligible when it is compared to the computational time required for the similarity searches.

The experimental section shows that the values of ρ and $\tau_{\alpha,\gamma}$ for varying weights produce smooth surfaces, thus the iterative search can have two phases: a first phase, with a large ε and histograms with fewer samples, and a second phase with finer ε and more precise histograms. Because the histogram of distances can be created from any distance function, these algorithms can (a priori) work with any kind of descriptor and distance function. However, there is an assumption of continuity for ρ and $\tau_{\alpha,\gamma}$, thus the underlying distances should produce smooth histograms like the ones produced by the Minkowski distances.

In practice, each underlying distance has two associated weights: an internal weight from its α-normalization, and an external weight from the weighting process of γ. Once the internal and external weight has been selected, both can be multiplied to fix the final weight to each underlying distance.

The two proposed weighting algorithms are just heuristics to select a good set of weights based on statistical indicators. However, these algorithms do not guarantee the quality of the set of weights because they do not perform any effectiveness evaluation. Sometimes the evaluation of the effectiveness is difficult because it requires the definition of proper methodology and indicators, and in some cases, it requires that people be hired to use the system and fill out evaluation forms. In those cases, these algorithms can offer a reasonably good set of weights –at least better than the normalization by maximum distance– to perform a combined search.

8.1.5 Evaluation

The evaluation uses the MUSCLE-VCD-2007 dataset, with reference videos from DB-MPEG1 collection and query videos from ST1 and ST2 collections. As stated in Section 5.6, $\mathcal{R}$ is the set of reference segments, $\mathcal{Q}$ is the set of query segments, and $\mathcal{Q}_c$ is the set of query segments that have a correct answer in $\mathcal{R}$. Each reference and query video was preprocessed and partitioned into segments of one second length (**S1**), thus $|\mathcal{R}|$=211,479, $|\mathcal{Q}|$=13,942 segments and $|\mathcal{Q}_c|$=9,983 segments.

Name	Segm.	d	Descriptor	ρ	MAP
EH	**S1**	L_1	$\mathbf{EH}^t$4x4-10	9.59	0.664
IH	**S1**	L_1	$\mathbf{IH}^t$1x4-rgb-3x16	8.58	0.501
KF	**S1**	L_1	$\mathbf{KF}^t$11x9	4.03	0.510

Table 8.1 – Base effectiveness of the three configurations to be combined.

α	norm.	$w_{\mathbf{EH}}$	$w_{\mathbf{IH}}$	ρ	$\tau_{\alpha,\gamma}$	MAP
1	max	0.5	0.5	12.4	0.97	0.640
0.5		0.5	0.5	12.8	1.00	0.675
0.1		0.5	0.5	12.8	1.06	0.671
0.1	max-ρ	0.552	0.448	12.8	1.06	0.687
0.1	max-τ	0.524	0.476	12.8	1.07	0.679
0.01		0.5	0.5	12.7	1.15	0.665
0.01	max-ρ	0.580	0.420	12.8	1.15	0.690
0.01	max-τ	0.546	0.454	12.8	1.15	0.679
0.001		0.5	0.5	12.6	1.26	0.656
0.001	**max-ρ**	**0.612**	**0.388**	**12.8**	**1.26**	**0.692**
0.001	max-τ	0.550	0.450	12.8	1.26	0.672
0.0001		0.5	0.5	12.2	1.49	0.631
0.0001	max-ρ	0.668	0.332	12.8	1.45	0.688
0.0001	max-τ	0.550	0.450	12.5	1.49	0.648

Table 8.2 – Effectiveness of γ when combining distances from **EH** and **IH**.

Combination of two and three descriptors

The following experiments are based on the effectiveness achieved by combination of the following three configurations: **EH**, **IH** and **KF**. Table 8.1 summarizes the indicators and base effectiveness (MAP) achieved by each configuration (the MAP values are the same to the frame-to-frame effectiveness shown in Figure 7.3 on page 82). The intrinsic dimensionality was calculated by evaluating the distance between randomly sampled pairs of segments (x, y) with $x \in \mathcal{Q}$ and $y \in \mathcal{R}$. The searches are implemented as linear scans. Note the relationship between ρ and MAP: **EH** achieves the best MAP and it also has the highest ρ. On the other hand, **KF** has the lowest ρ but outperforms **IH**. These three configurations will be combined using the normalization by maximum distance, α-normalization, weighting by max-ρ, and weighting by max-τ.

For **EH**+**IH** (Table 8.2) the best effectiveness is achieved by weighting by max-ρ with α=0.001. For **EH**+**KF** (Table 8.3) the best effectiveness is achieved by weighting by max-ρ with α=0.0001. For **IH**+**KF** (Table 8.4) the best effectiveness is achieved by weighting by max-τ with α=0.01. For **EH**+**IH**+**KF** (Table 8.5) the best effectiveness is achieved by weighting by max-τ with α=0.1.

Because **EH** by itself achieves the best MAP and has the highest ρ, the optimal weights for **EH**+**IH** and **EH**+**KF** are selected by the weighting by max-ρ algorithm. However, in the combination of **IH**+**KF**, the weights chosen by weighting by max-ρ are biased to **IH**. In this case, the weighting by max-τ achieves the best performance. In the combination of the three descriptors,

$\boldsymbol{\alpha}$	norm.	$\boldsymbol{w_{\mathbf{EH}}}$	$\boldsymbol{w_{\mathbf{KF}}}$	$\boldsymbol{\rho}$	$\boldsymbol{\tau_{\alpha,\gamma}}$	**MAP**
1	max	0.5	0.5	10.1	0.85	0.698
0.5		0.5	0.5	9.2	1.00	0.691
0.1		0.5	0.5	8.7	1.11	0.685
0.1	max-ρ	0.794	0.206	11.3	1.08	0.706
0.1	max-τ	0.566	0.434	9.5	1.11	0.694
0.01		0.5	0.5	8.2	1.23	0.678
0.01	max-ρ	0.820	0.180	11.3	1.16	0.706
0.01	max-τ	0.532	0.468	8.6	1.24	0.684
0.001		0.5	0.5	7.4	1.42	0.665
0.001	max-ρ	0.856	0.144	11.3	1.22	0.706
0.001	max-τ	0.516	0.484	7.6	1.42	0.668
0.0001		0.5	0.5	6.5	1.69	0.645
0.0001	**max-ρ**	**0.896**	**0.104**	**11.3**	**1.28**	**0.706**
0.0001	max-τ	0.510	0.490	6.6	1.69	0.647

Table 8.3 – Effectiveness of γ when combining distances from **EH** and **KF**.

$\boldsymbol{\alpha}$	norm.	$\boldsymbol{w_{\mathbf{IH}}}$	$\boldsymbol{w_{\mathbf{KF}}}$	$\boldsymbol{\rho}$	$\boldsymbol{\tau_{\alpha,\gamma}}$	**MAP**
1	max	0.5	0.5	7.9	1.00	0.564
0.5		0.5	0.5	6.9	0.99	0.584
0.1		0.5	0.5	6.7	1.07	0.586
0.1	max-ρ	0.936	0.064	8.6	1.02	0.520
0.1	max-τ	0.538	0.462	7.0	1.07	0.583
0.01		0.5	0.5	6.6	1.16	0.587
0.01	max-ρ	0.938	0.062	8.6	1.04	0.520
0.01	**max-τ**	**0.470**	**0.530**	**6.4**	**1.16**	**0.588**
0.001		0.5	0.5	6.4	1.23	0.588
0.001	max-ρ	0.946	0.054	8.6	1.05	0.520
0.001	max-τ	0.482	0.518	6.3	1.23	0.587
0.0001		0.5	0.5	6.3	1.21	0.587
0.0001	max-ρ	0.952	0.048	8.6	1.04	0.520
0.0001	max-τ	0.516	0.484	6.4	1.21	0.588

Table 8.4 – Effectiveness of γ when combining distances from **IH** and **KF**.

α	norm.	$w_{\mathbf{EH}}$	$w_{\mathbf{IH}}$	$w_{\mathbf{KF}}$	ρ	$\tau_{\alpha,\gamma}$	**MAP**
1	max	0.333	0.333	0.333	11.2	0.89	0.670
0.5		0.333	0.333	0.333	10.5	1.00	0.694
0.1		0.333	0.333	0.333	10.1	1.11	0.690
0.1	max-ρ	0.561	0.394	0.045	12.9	1.08	0.704
0.1	**max-τ**	**0.453**	**0.179**	**0.368**	**10.2**	**1.12**	**0.711**
0.01		0.333	0.333	0.333	9.7	1.24	0.684
0.01	max-ρ	0.575	0.362	0.063	12.9	1.18	0.709
0.01	max-τ	0.427	0.19	0.383	9.4	1.25	0.702
0.001		0.333	0.333	0.333	9.0	1.44	0.676
0.001	max-ρ	0.602	0.364	0.034	12.9	1.29	0.703
0.001	max-τ	0.447	0.173	0.38	8.8	1.46	0.693
0.0001		0.333	0.333	0.333	8.2	1.70	0.658
0.0001	max-ρ	0.675	0.296	0.029	12.9	1.50	0.705
0.0001	max-τ	0.436	0.189	0.375	7.9	1.78	0.675

Table 8.5 – Effectiveness of γ when combining distances from **EH**, **IH** and **KF**.

the weighting by max-τ achieves the best result. These results also show there is not a clear rule for locating the best α since the best effectiveness is achieved with some $\alpha \leq 0.1$.

Previous tables show that both weighting algorithms can improve the trivial selection of equal weights, outperforming the α-normalization and the normalization by maximum distance. However, these tables do not show the best set of weights that may have been selected, i.e., the set of weights that maximizes MAP. To analyze this issue, we test the performance for every set of weights in the combination of **EH**, **IH** and **KF**. Because the sum of weights is 1 and no weight can be negative, all the 3-tuples $(w_{\mathrm{EH}}, w_{\mathrm{IH}}, w_{\mathrm{KF}})$ reside in a plane, specifically on an equilateral triangle of side $\sqrt{2}$. We have uniformly sampled weights in the triangle following a regular grid with step 0.1 (66 samples). For each sampled set of weights we measure the value of ρ, $\tau_{\alpha,\gamma}$ and the MAP achieved by the similarity search.

Figure 8.3 summarizes values of ρ, $\tau_{\alpha,\gamma}$ and MAP for different combinations of weights when the underlying distances of γ have been α-normalized with α=0.1 and α=0.0001. The location of the weights that result in maximum ρ and $\tau_{\alpha,\gamma}$ are marked in the graphs (the actual values are already shown in Table 8.5). The set of sampled weights that achieves the maximum MAP is also marked on each graph. The 3D graph shows the three surfaces when the vertical axis has been displaced and scaled to match the minimum value at zero and the maximum value at one, thus the relationship between increase and decrease of ρ, $\tau_{\alpha,\gamma}$, and MAP is clearer. The figure shows that the weights that produce maximum ρ and $\tau_{\alpha,\gamma}$ do not coincide with the weights that achieve maximum MAP, however they are closer to it than weights $(\frac{1}{3},\frac{1}{3},\frac{1}{3})$. Therefore, the maximization of either ρ or $\tau_{\alpha,\gamma}$ selects a set of weights that outperforms the trivial selection of equal weights, but that achieves a sub-optimal MAP.

The fact that the weighting by max-ρ selects a sub-optimal configuration implies that there is a gap in which to improve both efficiency and effectiveness, and that the highest effectiveness does not necessarily imply the lowest efficiency. The weighting by max-ρ exploits the (sometimes loose) inverse correlation between efficiency and effectiveness. We should remark this approach must be used with prudence: for example, assume there are initial weights that achieve high MAP, that were

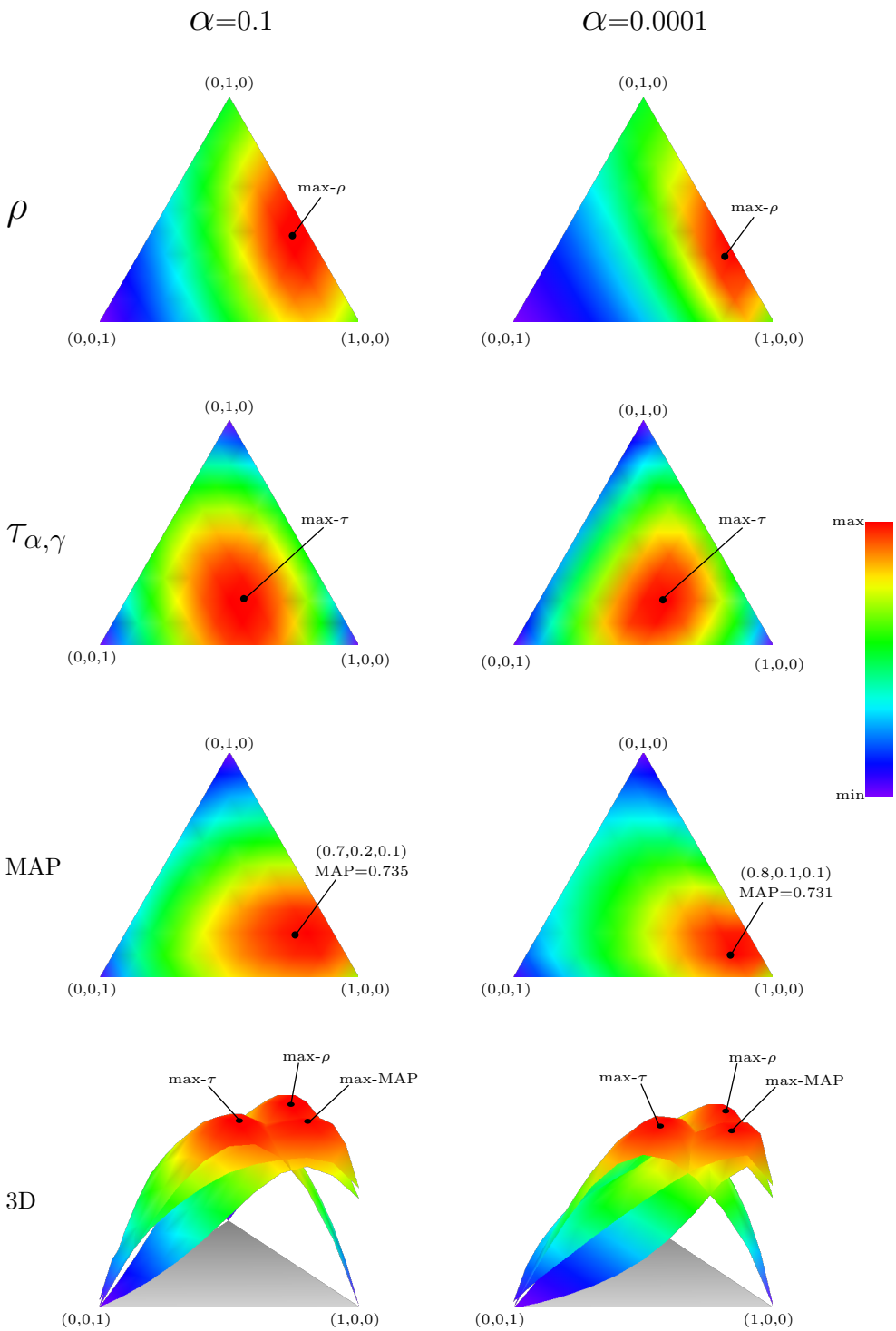


Figure 8.3 – Relationship between intrinsic dimensionality (ρ), the value that α-normalizes γ ($\tau_{\alpha,\gamma}$), and MAP, when combining distances from **EH**, **IH** and **KF**.

Name	Segm.	d	Descriptor	ρ	MAP
GH	**S1**	L_1	**GH**^{t}4x4-12	13.21	0.493
EH5	**S1**	L_1	**EH**^{t}4x4-5	7.26	0.561
OM	**S1**	L_1	**OM**^{t}9x9	6.18	0.339
CH	**S1**	L_1	**CH**^{t}2x2-rgb-4x4x4	6.92	0.439
G14	**S1**	L_1	**GH**^{t}1x4-32	8.71	0.488

Table 8.6 – Effectiveness of five more configurations to be combined.

located either by experimental evaluation or using a partial ground-truth. If we apply the weighting by max-ρ in order to improve the effectiveness, the algorithm will move the initial weights towards the weights that produce the maximum ρ, and it may happen that it is actually decreasing both effectiveness and efficiency.

The fact that the weighting by max-τ optimizes to a set with relatively similar weights shows there is not a strong correlation between **EH**, **IH** and **KF**. In order to test the performance of the weighting algorithm under correlated descriptors, we will perform a combination using more configurations.

Incremental combination

Table 8.6 shows the effectiveness for five more configurations: **GH**, **EH5**, **OM**, **CH** and **G14**. The following experiment combines all these configurations together with the configurations already shown in Table 8.1. The experiment compares the performance of the weighting algorithms for the incremental combination in the following (arbitrary) order: **EH**, **IH**, **KF**, **GH**, **EH5**, **OM**, **CH**, and finally **G14**.

Figure 8.4 shows the achieved MAP when γ combines incrementally, from one, up to eight configurations, using the normalization by maximum distance with equal weights, the α-normalization with equal weights (α=0.1), the weighting by max-ρ, and the weighting by max-τ. The results show that combining descriptors improves the effectiveness up to a saturation point (in this case, three descriptors), and adding more descriptors may even reduce the effectiveness. The α-normalization outperforms the normalization by maximum distance at every step. In turn, the weighting by max-τ outperforms the α-normalization at every step. Then, α-normalization performs a combination with a better scaling, and the weighting by max-τ can correct the trivial selection of equal weights.

In the experiment, the configuration **GH** acts as a noisy descriptor, decreasing the effectiveness of all weighting algorithms, in particular the weighting by max-ρ is highly impacted due to the high ρ of **GH**. However, as more descriptors are combined, the weighting by max-ρ corrects its behavior, outperforming the effectiveness achieved by the normalization by maximum distance when eight descriptors are combined.

8.2 Spatio-Temporal Combined Distance

Let γ be a spatial distance function between two segments, which may be a simple distance between descriptors or may be a weighted combination of descriptors. The spatio-temporal (s-t)

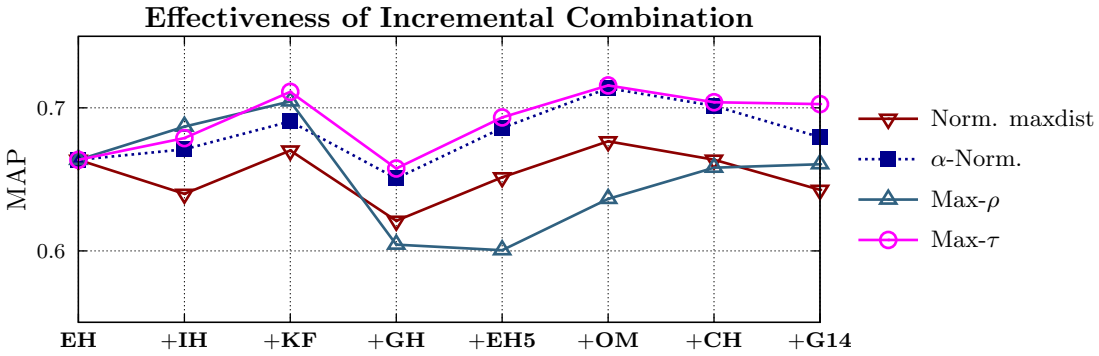


Figure 8.4 – Effectiveness of combining incrementally from one up to eight descriptors. Weights in γ use normalization by maximum distance (equal weights), α-Normalization (equal weights, α=0.1), weighting by max-ρ, and weighting by max-τ.

distance between two segments is the average distance of γ inside a temporal window.

Definition 6 (Spatio-Temporal Combined Distance) *The spatio-temporal combined distance δ between two segments is defined as:*

$$\delta(q_j, r_k) = \frac{1}{W} \sum_{w=-\lfloor W/2 \rfloor}^{\lfloor W/2 \rfloor} \gamma(q_{j+w}, r_{k+w})$$

where W is an odd number, q_j is the j^{th} segment for a video v partitioned into segment $\{q_1, ..., q_s\}$, $\forall i < 1$ q_i=q_1, and $\forall j > s$ q_j=q_s.

The scaling factor $1/W$ does not affect the nearest neighbors, thus it could be discarded. However, we included it because γ could be α-normalized, in which case a distance threshold for δ can be defined using 1 as a reference value.

If γ satisfies the metric properties, then δ also satisfies them. Note the behavior for comparing segments lying in the beginnings and endings of videos ($\forall i < 1$ q_i=q_1, $\forall j > s$ q_j=q_s). This alternative is preferred over an early-termination of the sum, because the latter may break the triangle inequality.

Some works that include a s-t distance have been reviewed in Chapter 4. In particular, the temporal distance for the ordinal measurement descriptor [Kim and Vasudev, 2005] is closely related. However, δ is more general because it can combine many descriptors and it is not associated with any extraction method.

8.2.1 Evaluation

The following experiment tests the performance of the s-t distance δ varying the size of the temporal window W. Figure 8.5 summarizes the effectiveness of varying the temporal window W={1,3,5,7,9} for the fourteen global descriptors evaluated in Chapter 7 plus the acoustic descriptor **AU**160. The experiment uses videos with preprocessing, s-t description and segmentation fixed to one second length (**S1**). The results for W=1 and global descriptors are shown in Figure 7.3 on

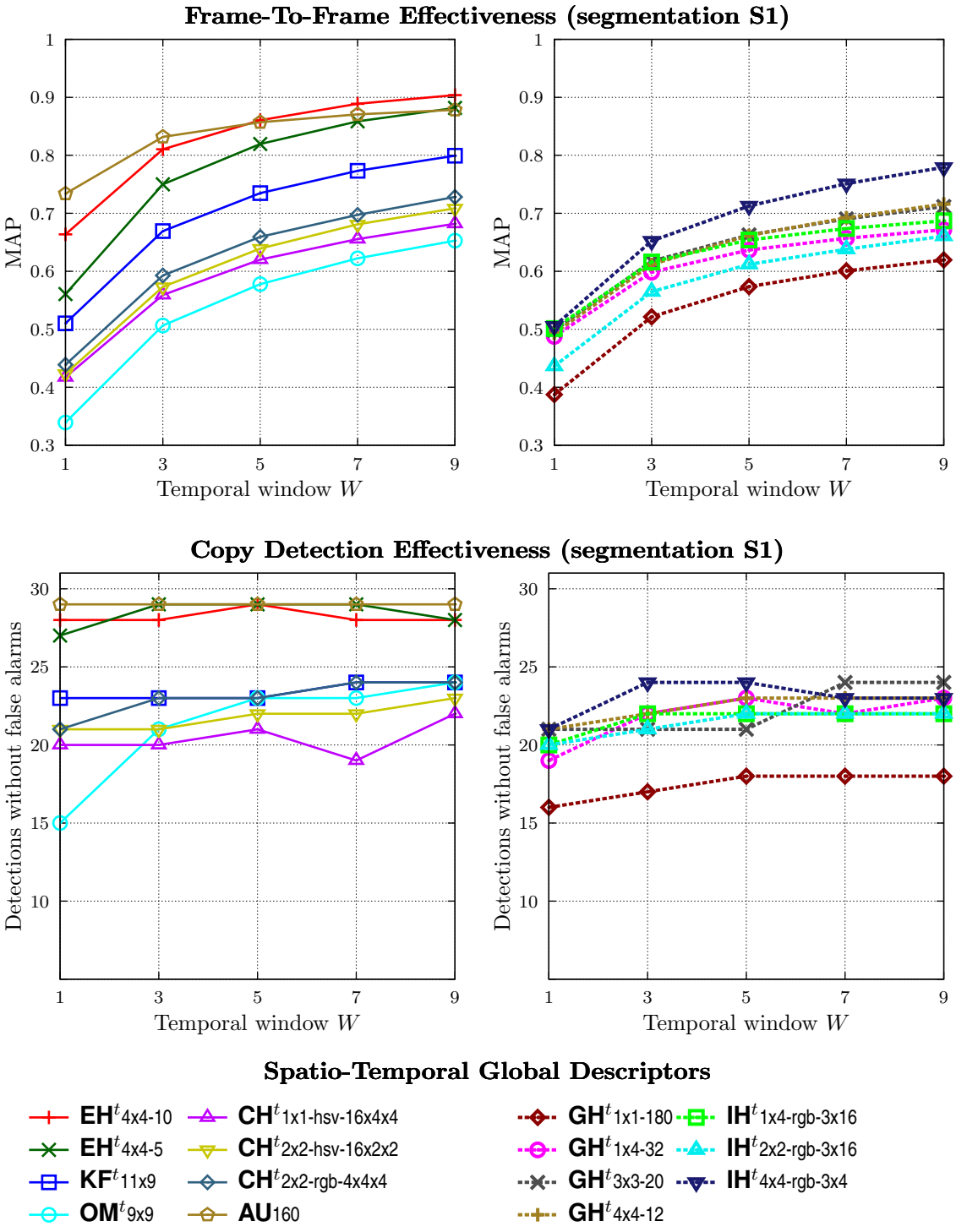


Figure 8.5 – Effectiveness of increasing temporal window W (videos with preprocessing, s-t description, segments of one second length).

page 82 for **S1**, and acoustic descriptor in Figure 7.4 on page 85 for m=160 bands, w=500 ms, s=20 ms at **S1**.

In the case of frame-to-frame effectiveness, each descriptor increases its effectiveness in almost a similar proportion. In particular, the $\mathbf{EH}^{t}_{\text{4x4-10}}$ increases its MAP up to more than 0.9 for W=9. On average for all the descriptors, the increases compared with the MAP achieved by W=1 are {29%, 41%, 48%, 52%} respectively for W in {3, 5, 7, 9}. The improvements in the MAP for every descriptor shows that a larger W can improve the frame-to-frame effectiveness. The experiment uses segments of one second length, but it should be noted that with different segment lengths the behavior is similar.

In the case of detection effectiveness, the improvements do not increase as well as the MAP. Moreover, the detections remain mainly stable with some small increases (even decreases for some descriptors). The only descriptor that greatly benefits from large W is $\mathbf{OM}^{t}_{\text{9x9}}$ no longer achieving the worst performance, but a performance comparable with the other descriptors. This result shows that large W mainly does not improve the detection performance, thus the improvement in MAP is due to better matching between frames from already detectable copies.

In summary, a large temporal window is useful for improving the copy localization (and also the detection score) for copies that are already detectable by a small window, but it is not useful for discovering more copies. Moreover, a large temporal window also implies more computational effort, e.g., an increase of W from 1 to 11 implies 10 times more evaluations of γ function.

In an exact search, where every query segment is compared to every reference segment, most of the evaluation of γ are repeated, thus an optimization, like memoization, can be used. For example, with W=3 the evaluation of $\delta(q_i, r_j)$ and $\delta(q_{i+1}, r_{j+1})$ shares the evaluation of $\gamma(q_i, r_j)$ and $\gamma(q_{i+1}, r_{j+1})$. In practice, for exact searches, an increase in W will not affect the amount of evaluations of γ, but will require more memory to store all its evaluations.

However, this optimization will not be effective for approximate searches. In approximate searches most of the evaluations of δ are avoided and only a few evaluations are actually performed. Then the evaluations of δ may not share the evaluations of γ. In this case, the cost for each evaluation of δ will increase proportionally with W, but the detection performance does not increase at the same ratio.

As we show in Chapter 11, for TRECVID 2010 we performed approximate searches using W=3 due to its increase in the effectiveness. However, for TRECVID 2011 we prefer to fix W to 1 and to improve the approximation parameters (which show better performance revenues, see Chapter 9).

Combination of acoustic and visual descriptors

Chapter 7 shows the performance of an acoustic descriptor that behaves in a similar way as global descriptors. In this experiment we perform a combination of visual and acoustic information in the similarity search using different weights and temporal window W. Table 8.7 shows the configurations **EH** and **AU** and their effectiveness.

Figure 8.6 shows the effectiveness of the s-t distance varying the size of the temporal window W. The distance γ is a weighted combination of **EH** and **AU**, using α-normalization (α=0.1) and five different sets of weights. The best frame-to-frame effectiveness is achieved by equal weights

Name	Segm.	d	Descriptor	ρ	MAP
EH	**S1**	L_1	$\mathbf{EH}^t_{4\text{x}4\text{-}10}$	9.59	0.664
AU	**S1**	L_1	$\mathbf{AU}_{160}$	7.53	0.734

Table 8.7 – Base effectiveness of the two configurations to be combined.

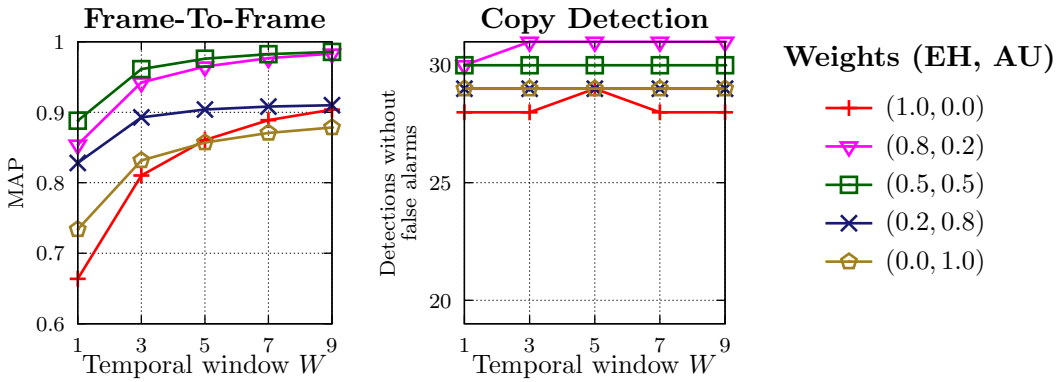


Figure 8.6 – Effectiveness of s-t distance with temporal window W, when γ is a combination of **EH** and **AU**, α-normalization (α=0.1) and five different sets of weights.

w_{EH}=w_{AU}=0.5 and W=9 with MAP 0.986. However, that configuration detects 30 copies out of 31: it misses the copy that has no audio track. In turn, weights w_{EH}=0.8, w_{AU}=0.2 achieves a slightly lower MAP but it detects all the 31 copies with $W \geq 3$.

Both weighting algorithms chose relatively similar weights: weighting by max-ρ selects weights w_{EH}=0.572, w_{AU}=0.428, and weighting by max-τ selects weights w_{EH}=0.514, w_{AU}=0.486, both sets detecting 30 of 31 copies. This behavior shows a weakness of the automatic weighted combination: the difficulty to tune the combination when some underlying distance may use a misleading descriptor. This issue may be overcome using some descriptor-specific behavior (e.g., decrease w_{AU} if a video lacks audio track), or using previous knowledge of the dataset (manually adjusting weights).

Besides, to the best of our knowledge, the result of 100% correct detections without false alarms (achieved by w_{EH}=0.8, w_{AU}=0.2, W=3, α=0.1) outperforms a CBVCD system using local descriptors that achieves the best result on the MUSCLE-VCD-2007 dataset [Poullot et al., 2010].

8.3 Other Metric and Non-Metric Distances

In previous chapters, we have exclusively used the L_1 distance for comparing descriptors. The L_1 distance satisfies the metric properties and also it is fast to evaluate, but a comparison with other distances is required.

The metric properties present a tradeoff between effectiveness and efficiency. Thus, we may improve the effectiveness of the distance function by raising the restrictions of a metric. When a distance function does not satisfy some of the metric properties it is known as a *non-metric*. Section 3.3.2 reviews some related work for the non-metric approach and some non-metric distances.

The following experiment compares the performance achieved by the fourteen descriptors and the acoustic descriptor using the following distances:

- Minkowski distance: L_1 and L_2. These distances satisfy the metric properties.
- Fractional distance: $\mathsf{L}_{0.5}$ and $\mathsf{L}_{0.1}$. These distances do not satisfy the metric properties.
- Chi-squared test statistic: χ^2. This distance does not satisfy the metric properties.
- DPF distance: $\mathsf{D}^1_{3\%}$, $\mathsf{D}^1_{6\%}$ and $\mathsf{D}^1_{20\%}$. These distances correspond to evaluate L_1 after discarding the 3%, 6% and 20% of the dimensions with higher differences, respectively. These distances do not satisfy the metric properties.

The similarity search using Fractional and Chi-squared distances is relatively slower than Minkowski distances (about two or three times slower), while the search using DPF is more than one order of magnitude slower (mainly affected by the storage and sorting of differences by dimension). We did not evaluate other more complex distances (like the Earth Mover's Distance [Rubner et al., 2000]) because their evaluation time (more than two orders of magnitude slower) makes them unfeasible to test them in the MUSCLE-VCD-2007 dataset.

Figure 8.7 compares the effectiveness achieved by the eight distances. The experiment uses videos with preprocessing, s-t description and segmentation fixed to one second length (**S1**). The results for L_1 are also shown in Figure 7.3 on page 82 for **S1**.

In the case of frame-to-frame effectiveness, the $\mathsf{L}_{0.5}$ produces a slight increase in the MAP for $\mathbf{EH}^t$, $\mathbf{GH}^t$ and $\mathbf{IH}^t$ descriptors, the χ^2 produces a slight increase for $\mathbf{CH}^t$ and $\mathbf{IH}^t$ descriptors, and the DPF distances produces a slight increase for $\mathbf{EH}^t$, $\mathbf{GH}^t$ and $\mathbf{IH}^t$ descriptors. Between the three DPFs, the $\mathsf{D}^1_{6\%}$ produces best results, while $\mathsf{D}^1_{20\%}$ only improves MAP for two $\mathbf{GH}^t$ descriptors and one $\mathbf{IH}^t$. The global maximum for global descriptors is achieved by $\mathsf{D}^1_{6\%}$ (0.692) and $\mathsf{L}_{0.5}$ (0.688) both at $\mathbf{EH}^t{}_{\text{4x4-10}}$. On the other hand, the L_2 and $\mathsf{L}_{0.1}$ are outperformed by L_1 for every descriptor (except at $\mathbf{GH}^t{}_{\text{3x3-20}}$).

In the case of detection effectiveness, the $\mathsf{L}_{0.5}$ outperforms L_1 for almost every descriptor (except at $\mathbf{CH}^t{}_{\text{1x1-hsv-16x4x4}}$), and χ^2 and $\mathsf{D}^1_{3\%}$ on average produce the same effectiveness as L_1. On the other hand, L_2, $\mathsf{L}_{0.1}$, $\mathsf{D}^1_{6\%}$ and $\mathsf{D}^1_{20\%}$ on average worsen the detection effectiveness. The global maximum for global descriptors is achieved by $\mathsf{L}_{0.5}$ at $\mathbf{EH}^t{}_{\text{4x4-10}}$ with 29 detections without false alarms.

In summary, the replacement of L_1 with a non-metric distance may improve the effectiveness. In particular, the effectiveness achieved L_1 can be outperformed by a fractional distance with p in the range $[0.5, 1)$. The intuition behind this good performance of fractional distances is that they penalize descriptors with small differences in many dimensions, and favor descriptors that concentrate high differences on few dimensions. For example, given a query vector A=$(9, 9, 9, 9)$, and two objects B=$(5, 5, 5, 5)$ and C=$(1, 1, 9, 9)$, L_2 selects B as the nearest neighbor of A, L_1 evaluates B and C with the same distance to A, and $\mathsf{L}_{0.5}$ selects C as the nearest neighbor of A. In fact, when the descriptors represent zones of a frame, the latter alternative is preferable because a postproduction transformation usually modifies a few zones while leaving the other zones unchanged.

The DPF distances can also outperform L_1. They discard dimensions with high differences before evaluating the distance, thus they can avoid the noise that a transformation may have pro-

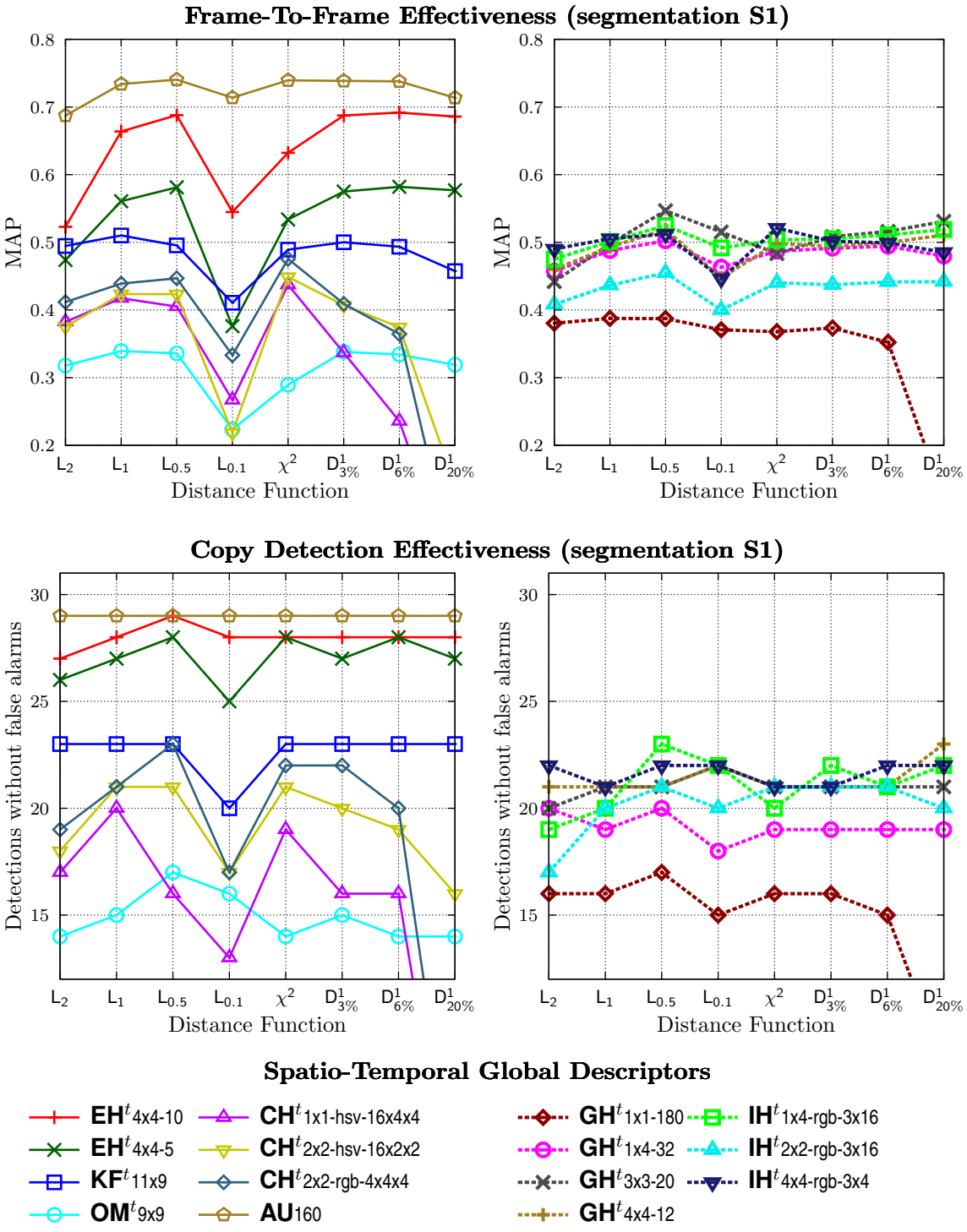


Figure 8.7 – Effectiveness achieved by different distance functions (videos with preprocessing, s-t description, segments of one second length).

duced in a descriptor. However, they tend to improve the effectiveness of copies already detectable by L_1 rather than detecting new copies.

Finally, we choose L_1 because it achieves a good balance between effectiveness and efficiency. Its effectiveness may be outperformed by a fractional distance, but at the cost of higher evaluation time and breaking the triangle inequality.

8.4 Summary

In this chapter we have presented three approaches to improve the effectiveness of the distance function.

The first approach combines the distance from many descriptors into a unique static weighted distance. This combined distance can improve the effectiveness of the search by using different sources of information. The α-normalization enables the comparison of distances between nearest neighbors from different metric spaces. The weighting by max-ρ and weighting by max-τ are two algorithms that intend to automatically select a good set of weights for the combination. The main strength of these algorithms is that they do not need training data nor effectiveness evaluations. These algorithm use straightforward statistical indicators, that can be calculated rapidly by sampling distances from the metric spaces.

The second approach uses a spatio-temporal combined distance. This distance exploits the temporal dimension of videos to improve the effectiveness. The evaluation shows that the frame-to-frame match between copies and original segments can be highly improved using a large temporal window, however the detection effectiveness does not improve as well. In particular, we have shown a combination of a global and acoustic descriptors that fully resolves the MUSCLE-VCD-2007 dataset.

The third approach raises the metric properties of the distance to use some of the well-known non-metric distances. In particular, we have shown the fractional distances can indeed improve the effectiveness, however the tradeoff between the effectiveness and efficiency still favors the L_1 distance.

The next chapter reviews the efficiency of the search assuming a good distance function has been defined.

Chapter 9

Improving Efficiency in the Similarity Search

The objective of the Similarity Search is to perform NN+range searches to retrieve the k most similar reference segments at a distance threshold ϵ for each segment $q \in \mathcal{Q}$, according to a distance function d.

In this chapter we will focus on performing an efficient search using an already defined distance d. We present two approaches to perform an efficient search: approximate searches using a static pivot table, and exact searches using a dynamic pivot table.

9.1 Approximate Search with Pivots

LAESA (see Chapter 3) selects a set of pivots from the reference objects to perform an efficient search. During the search, before evaluating the distance between the query and every object, it calculates its lower bound. If the lower bound is larger than the current k^{th} nearest neighbor candidate, the object can be safely discarded without affecting the final result.

This section presents an approximate search algorithm which uses the lower bounds as fast distance estimators. This search discards most of the objects and evaluates the actual distance only for the most promising objects (objects with the lowest lower bounds). This section also presents an application of the approximate search for local descriptors, and a Two-step search which first performs approximate searches and then performs an exact search.

9.1.1 Index Structure

Let $\mathcal{D}$ be the set of segments, $d : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ be a distance between any two segments, $\mathcal{R} \subseteq \mathcal{D}$ be the set of reference segments in the CBVCD system, and $\mathcal{P} \subseteq \mathcal{R}$ be a set of pivots. The $LB_{\mathcal{P}}$ function is defined as:

$$LB_{\mathcal{P}}(q, r) = \max_{p \in \mathcal{P}} \{|d(q, p) - d(r, p)|\} \tag{9.1}$$

Algorithm 9.1: The Sparse Spatial Selection (SSS) algorithm.

Input: $\mathcal{R}$ set of reference segments, t sparse threshold.
Output: $\mathcal{P}$ set of pivots

$\{r_0, ..., r_{|\mathcal{R}|}\} \leftarrow$ randomize objects in $\mathcal{R}$
$\mathcal{P} \leftarrow \{r_0\}$
foreach $r_i \in \{r_1, ..., r_{|\mathcal{R}|}\}$ **do**
  if $\forall$ p $\in \mathcal{P}$, $d(r_i, \text{p}) \geq t$ **then**
    $\mathcal{P} \leftarrow \mathcal{P} \cup \{r_i\}$
return $\mathcal{P}$

If d satisfies the metric properties, the object-pivot distance constraint implies that:

$$\forall\, q, r \in \mathcal{D},\ \forall\, \mathcal{P} \subseteq \mathcal{R},\ \ LB_{\mathcal{P}}(q, r) \leq d(q, r) \tag{9.2}$$

If $d(x, p)$ is precalculated $\forall x \in \mathcal{D}\, p \in \mathcal{P}$, the evaluation of $LB_{\mathcal{P}}$ costs just $|\mathcal{P}|$ operations. The index structure is LAESA, i.e., a $|\mathcal{R}| \times |\mathcal{P}|$ pivot table containing distances from each pivot to every reference segment. Additionally, if d is a spatio-temporal distance, then each segment must have a pointer to the previous segment and the next segment in the video segmentation.

The Sparse Spatial Selection (SSS) [Bustos et al., 2008] incrementally selects pivots according to the minimum distance that they should have between each other. This algorithm returns a variable number of pivots depending on the distance threshold and the traverse order of $\mathcal{R}$. Algorithm 9.1 shows the implemented SSS algorithm. It first randomizes $\mathcal{R}$ and then performs the incremental selection.

The approximate search uses the lower bound as an estimator of the distance, hence the search requires a set of pivots that produces tight values between the distance and its lower bound. Therefore, given n candidate sets $\{\mathcal{P}_1, ..., \mathcal{P}_n\}$ we select the set $\mathcal{P}_i$ that minimizes $|d(x, y) - LB_{\mathcal{P}_i}(x, y)|$, for randomly sampled x and y. This criterion is equivalent to select the set that maximizes the average lower bound $\mu_{\mathcal{P}}$, proposed by Bustos et al. [2003]. In fact, the latter is preferable in order to avoid the evaluation of $d(x, y)$ and save some computational cost. The calculation of $\mu_{\mathcal{P}}$ is performed by evaluating the lower bound between randomly sampled pair of objects in the collection.

Given two sets $\mathcal{P}_1$ and $\mathcal{P}_2$ with $|\mathcal{P}_1|<|\mathcal{P}_2|$, it may be expected that $\mu_{\mathcal{P}_1}<\mu_{\mathcal{P}_2}$. However, the $\mu_{\mathcal{P}_1}>\mu_{\mathcal{P}_2}$ scenario may appear due to either redundant pivots in $\mathcal{P}_2$ or over-fitting of $\mathcal{P}_1$ to the evaluating pairs. When selecting sets with a few pivots from a large collection of objects, it is unlikely that SSS selects redundant pivots, and in the long-term it is more likely that $\mathcal{P}_2$ will produce tighter values between d and $LB_{\mathcal{P}}$, despite $\mathcal{P}_1$ achieves a higher $\mu_{\mathcal{P}}$ on the random sample. Thus, unlike Bustos et al. [2003], we first fix a target number of pivots and then we select the set with highest $\mu_{\mathcal{P}}$. This assumption that smaller sets of pivots are outperformed by bigger sets is proved in the results shown in the experimental section.

Algorithm 9.2 shows the selection algorithm for a target number of pivots m. The parameters n and s (the n sets of pivots to evaluate and the s pairs to use in the evaluation) can be adjusted to take reasonable computational time (in our implementation the selection takes a few minutes). The algorithm calculates a histogram of distances of d, and invokes the SSS algorithm to select

Algorithm 9.2: Pivot Selection Algorithm.

Input: $\mathcal{R}$ set of reference segments, m number of pivots, n number of sets to evaluate, s number of evaluation pairs.

Output: $\mathcal{P}$ set of pivots

```
{x_1, y_1, ..., x_s, y_s} ← random sample of 2 · s objects in R
F_d ← cumulative distribution of distances of d
α ← 1
foreach i ∈ {1, ..., n} do
    while |P_i| ≠ m do
        P_i ← SSS (R, F_d^{-1}(α) )                          // Algorithm 9.1
        if |P_i| < m then
            α ← α · (1 − ε)
        else if |P_i| > m then
            discard last (m − |P_i|) pivots from P_i
    μ_{P_i} ← average value for LB_P(x_j, y_j), j = {1, ..., s}   // Equation 9.1
return set P_i with maximum μ_{P_i}
```

pivots. The sparse threshold is calculated using the accumulated distribution of distances (see Section 8.1.1). It starts with the maximum distance (α=1) and it decreases when SSS could not select the required number of pivots. Actually, it is preferable to try a few times with the SSS before decreasing α. Finally, the set $\mathcal{P}$ with maximum $\mu_{\mathcal{P}}$ is selected to create the index. This algorithm can be accelerated by selecting each candidate set in an independent thread.

9.1.2 Exact Search with pivots

For every query segment $q \in \mathcal{Q}$, the first step is to calculate $d(q,p)\,\forall p \in \mathcal{P}$. Then, the NN+range search retrieves the k closest objects inside a distance threshold ϵ to q. Algorithm 9.3 depicts the algorithm for the classic exact NN+range search using pivots. It uses Equation 9.2 to evaluate d only when the lower bound is less than both the range ϵ and the k^{th} candidate distance.

The exact search with pivots can achieve a big improvement in the search time compared to a linear scan, and it retrieves the same objects. However, when the search is performed in large collections, this algorithm may not be as fast as needed. For those cases, we propose an approximate search algorithm.

9.1.3 Approximate Search with pivots

The exact search with pivots may improve the search time compared to a linear scan, however in some large collections (as the TRECVID datasets) it is not fast enough. In fact, for our participation at TRECVID 2010, the exact search with pivots would have taken several months to complete.

Algorithm 9.4 shows an algorithm which returns an approximation of the result of the exact search and can perform much faster. It is based on the fact that the lower bounds of the nearest neighbors are usually between the smallest lower bounds. It is divided in two rounds. In the first

Algorithm 9.3: Classic NN+range search using pivots.

Input: q query segment, $\mathcal{R}$ set of reference segments, k number for NN search, ϵ threshold for range search, $\mathcal{P}$ set of pivots.
Output: List of k nearest neighbors to q

```
NNs ← new priority queue
foreach r ∈ R do
    lb ← LB_P(q, r)                                          // Equation 9.1
    if lb < ε and ( size of NNs < k or
          lb < max distance in NNs ) then                    // Equation 9.2
        dist ← d(q, r)
        if dist < ε then
            add r to NNs with distance dist
            if size of NNs > k then
                remove max distance object from NNs
return NNs
```

round it uses $LB_{\mathcal{P}}$ as an estimator for d: it evaluates $LB_{\mathcal{P}}$ for every object in $\mathcal{R}$, it discards objects with $LB_{\mathcal{P}}$ greater than threshold ϵ, and it selects the $T{\cdot}|\mathcal{R}|$ objects with the smallest values of $LB_{\mathcal{P}}$, given an approximation parameter $T \in [0, 1]$. In the second round, it evaluates d just for the selected objects, and it locates between them the k nearest neighbors that are closer than ϵ to q. This is an approximate search because there is no guarantee that the $LB_{\mathcal{P}}$ for the actual nearest neighbors will be between the $T{\cdot}|\mathcal{R}|$ smallest values of $LB_{\mathcal{P}}$.

The key difference between the exact search with pivots and the approximate search is that while the exact search uses $LB_{\mathcal{P}}$ values as lower bounds for discarding objects that are far away from the query, Algorithm 9.4 compares $LB_{\mathcal{P}}$ values between them, assuming that a low/high value for $LB_{\mathcal{P}}$ implies a low/high value for d. Note that Equation 9.2 is not used on the approximate search because $LB_{\mathcal{P}}$ is used just as a fast estimator and not as a lower bound of d.

As previously mentioned, selecting more pivots for $\mathcal{P}$ implies tighter estimations, but it also implies a higher computational cost for evaluating $LB_{\mathcal{P}}$. However, with tighter estimations a smaller T is required to select the actual nearest neighbors. The tradeoff between $|\mathcal{P}|$ and T is analyzed in the experimental section. Note that as T moves closer to 1, Algorithm 9.4 will produce the same results of Algorithm 9.3 independent of $\mathcal{P}$. In particular, when T=1 the approximate search will evaluate d for the whole reference collection, thus it will select the same nearest neighbors as the exact search (but at a higher computational cost).

In preliminary experiments we tested other estimators based on pivots. In particular, we tested the minimum upper bound, the average bound (the average between lower and upper bounds), and the median of the average bound for all the pivots. However, these estimators were not worth their higher computational cost compared with $LB_{\mathcal{P}}$.

9.1.4 Approximate Search with Pivots for Local Descriptors

The approximate search with pivots requires a single distance to compare segments. In order to use local descriptions in the search, it is possible to directly use Matches or Spatial (see

Algorithm 9.4: Approximate NN+range search using pivots.

Input: q query segment, $\mathcal{R}$ set of reference segments, k number for NN search, ϵ threshold for range search, $\mathcal{P}$ set of pivots, T approximation parameter.
Output: List of approximate k nearest neighbors to q

```
MinLbs ← new priority queue
foreach r ∈ R do
    lb ← LB_P(q, r)                                   // Equation 9.1
    if lb < ε then
        add r to MinLbs with distance lb
        if size of MinLbs > T · |R| then
            remove max distance object from MinLbs
NNs ← new priority queue
foreach r ∈ MinLbs do
    dist ← d(q, r)
    if dist < ε then
        add r to NNs with distance dist
        if size of NNs > k then
            remove max distance object from NNs
return NNs
```

Chapter 7) as d in Algorithm 9.4. However, those distances are non-metrics, hence $LB_{\mathcal{P}}$ will not correctly estimate them, and the approximate search with pivots will achieve poor performance. Nonetheless, those distances rely on a metric distance d^L to compare local vectors (e.g., L_2 for SIFT vectors), thereby we can adapt the approximate search to estimate d^L instead of the non-metric distance.

As in previous section, $\mathcal{Q}$ and $\mathcal{R}$ are to be the sets of query and reference segments, where each segment is now represented by one or more local vectors. Let $\mathcal{Q}^L$ be the set of all the local vectors for every query segment, and $\mathcal{R}^L$ be the set of all the local vectors for reference segments. Essentially, for every vector in $\mathcal{Q}^L$ the search retrieves its m closest vectors in $\mathcal{R}^L$ according to d^L.

First, a set of vectors $\mathcal{P}^L \subseteq \mathcal{R}^L$ are selected as pivots using Algorithm 9.2. Then, given two segments q and r we define the ApMatches distance as:

$$\mathsf{ApMatches}_{\mathcal{P},m,T}(q,r) = 1 - |ApPairs_{\mathcal{P},m,T}(q,r)|/|\mathbf{SF}(q)|$$

$$ApPairs_{\mathcal{P},m,T}(q,r) = \left\{(x,y) \in \mathbf{SF}(q) \times \mathbf{SF}(r),\ y \in \mathrm{ApNN}_{\mathcal{P},m,T}(x)\right\} \quad ,$$

where $\mathrm{ApNN}_{\mathcal{P},m,T}(x)$ is the set of the m nearest neighbors of x, which are retrieved from $\mathcal{R}^L$ using approximation parameter T and $LB_{\mathcal{P}^L}$ as the estimator of d^L.

Finally, the approximate search is just a linear scan, where for each query segment in $\mathcal{Q}$ retrieves its k closest reference segments in $\mathcal{R}$ according to the distance ApMatches. The scan may evaluate $\mathsf{ApMatches}(q,r)$ $\forall q \in \mathcal{Q}$, $\forall r \in \mathcal{R}$ at any order, but in practice the scan first calculates the sets ApNN for every local vector in q, and afterwards resolves all the $\mathsf{ApMatches}(q,\cdot)$ consecutively.

Amato et al. [2011] present a distance between images which is based on k-NN searches in the local-vectors space. They use that approach for image classification, reporting comparable results

to both image-to-image matches and visual codebooks. The main difference with our approach is that ApMatches is based on approximate searches instead of exact searches, thus it can scale to larger collections. Unlike Amato et al. [2011], for efficiency considerations, we allow ApNN to contain vectors from the same segment and avoid performing spatial consistency tests.

The performance of ApMatches depends on parameters m, T, $|\mathcal{P}|$ which are used to retrieve the ApNN sets. In the experimental section we evaluate the effectiveness of ApMatches with different parameters compared to Matches and Spatial.

9.1.5 Two-step Search

As tested in Chapter 8, a distance which combines many descriptors can be used in order to increases the effectiveness of the search. However, a combined distance usually implies higher intrinsic dimensionality (as shown in the experimental section). The increase of the intrinsic dimensionality, in turn, affects the quality of the estimations, which decreases the effectiveness of the approximate search. This sort of paradox motivated us to define the following Two-step search.

As noted in the previous section, the effectiveness of the approximate search with pivots can be improved either by increasing $|\mathcal{P}|$ or by increasing T, both at the cost of increasing the search time. In order to avoid those alternatives for a combined distance, we propose to locate candidates by performing approximate searches using the underlying distances, and then perform an exact search between the candidates using the combined distance.

More formally, let $\{d_1, ..., d_m\}$ be the underlying distances of γ, let $\mathcal{V}$ be the set of reference videos, and let $\mathcal{C}$ be the set of query videos. Given a query video $c \in \mathcal{C}$, for every query segment $q \in c$ the first step performs m approximate k-NN searches –one for each d_i– retrieving the sets of reference segments k-NN(q, d_i). All the retrieved reference segments are then gathered:

$$N(c) = \bigcup_{\substack{q \in c \\ 1 \leq i \leq m}} k\text{-NN}(q, d_i)$$

Given the set $N(c) \subseteq \mathcal{R}$, the set $\mathcal{V}(c) \subseteq \mathcal{V}$ is defined as the D videos with more different segments in $N(c)$:

$$\mathcal{V}(c) = \{X \subseteq \mathcal{V}, |X| \leq D, \forall v \in X, u \in \mathcal{V} - X, |v \cap N(c)| \geq |u \cap N(c)|\}$$

The second step performs an exact k-NN search for every $q \in c$ using γ. The nearest neighbors are retrieved from the search space $\mathcal{R}(c) \subseteq \mathcal{R}$, which is defined as the union of all the segments from videos in $\mathcal{V}(c)$, i.e., $r \in \mathcal{R}(c)$ iff $\exists v \in \mathcal{V}(c)$ which $r \in v$. With this reduction of the search space (considering segments from the most promising videos), it is expected that $|\mathcal{R}(c)| \ll |\mathcal{R}|$ and thus the exact search with γ can be resolved in a reduced time.

Moreover, this definition can be extended to define a general Two-step search for a query video c, where the first step performs one or more similarity searches in the reduced search space $\mathcal{R}$ using distances d_i, and the second step performs an exact search in the reduced search space $\mathcal{R}(c)$ using a distance γ. The γ function may combine different descriptors or perform some complex computations, and if it satisfies the metric properties the search efficiency can be improved by using some MAM.

The Two-step search was used for our participation at TRECVID 2011 to efficiently search with a distance which combined audio and visual descriptors. The first step retrieved k_1 nearest neighbors according to distance d_a (which compares acoustic descriptors) and k_2 nearest neighbors according to distance d_v (which compares visual descriptors), and the second step performs a k-NN search according to distance d_{av} (which combines acoustic and visual descriptors) in a reduced set of candidate videos.

9.1.6 Evaluation

The evaluation uses the MUSCLE-VCD-2007 dataset, with reference videos from DB-MPEG1 collection and query videos from ST1 and ST2 collections. As stated in Section 5.6, $\mathcal{R}$ is the set of reference segments, $\mathcal{Q}$ is the set of query segments. Each reference and query video was preprocessed and partitioned into segments of one second length (**S1**), thus $|\mathcal{R}|$=211,479 and $|\mathcal{Q}|$=13,942 segments.

	Configuration			Histogram of distances H_d						
Name	Segm.	d	Descriptor		max	μ	σ	ρ	Time	MAP
KF	**S1**	L_1	$\mathbf{KF}^t$11x9		25199	7692	2708	4.0	139	0.510
EH	**S1**	L_1	$\mathbf{EH}^t$4x4-10		8101	3279	749	9.6	270	0.664
EIK	**S1**	γ	L_1-$\mathbf{EH}^t$4x4-10 L_1-$\mathbf{IH}^t$1x4-rgb-3x16 L_1-$\mathbf{KF}^t$11x9		3.371	1.50	0.33	10.1	1319	0.690
EK3	**S1**	δ	L_1-$\mathbf{EH}^t$4x4-10 L_1-$\mathbf{KF}^t$11x9		5.945	2.16	0.55	7.9	2254	0.795

Table 9.1 – Configurations used in following experiments. The combination in γ uses α-normalization (α=0.1) with equal weights, and the temporal window in δ is W=3.

The following experiments use four different configurations: **OM**, **KF**, **EH**, **IH**, **EIK**, and **EK3**. Table 9.1 depicts the histogram of distances (maximum distance, mean, standard deviation and intrinsic dimensionality) for $d(x, y)$ with randomly sampled $x \in \mathcal{Q}$ and $y \in \mathcal{R}$, time spent (in seconds) by a non-parallel process which performs $|\mathcal{Q}|$ linear scans to retrieve the 1-NN, and the achieved MAP according to the ground-truth. The distance γ for **EIK** is the combination of three L_1 distances with α-normalization, α=0.1, and equal weights. The spatio-temporal distance δ for **EK3** uses a window W=3 combining two L_1 distances with α-normalization, α=0.1, and equal weights.

Performance of the approximate search with pivots

This experiment analyzes the performance of the approximate search with pivots compared to the linear scan. Given a query object q whose nearest neighbor retrieved by the linear scan is r^q, the approximate search with pivots will return the same nearest neighbor depending on the value of $LB_{\mathcal{P}}(q, r^q)$. In fact, if $LB_{\mathcal{P}}(q, r^q)$ is between the $T{\cdot}|\mathcal{R}|$ smallest values of $LB_{\mathcal{P}}(q, r)$ for all $r \in \mathcal{R}$, then r^q will be promoted to the second round, where the actual distance $d(q, r^q)$ will be evaluated and r^q will be returned as the first nearest neighbor.

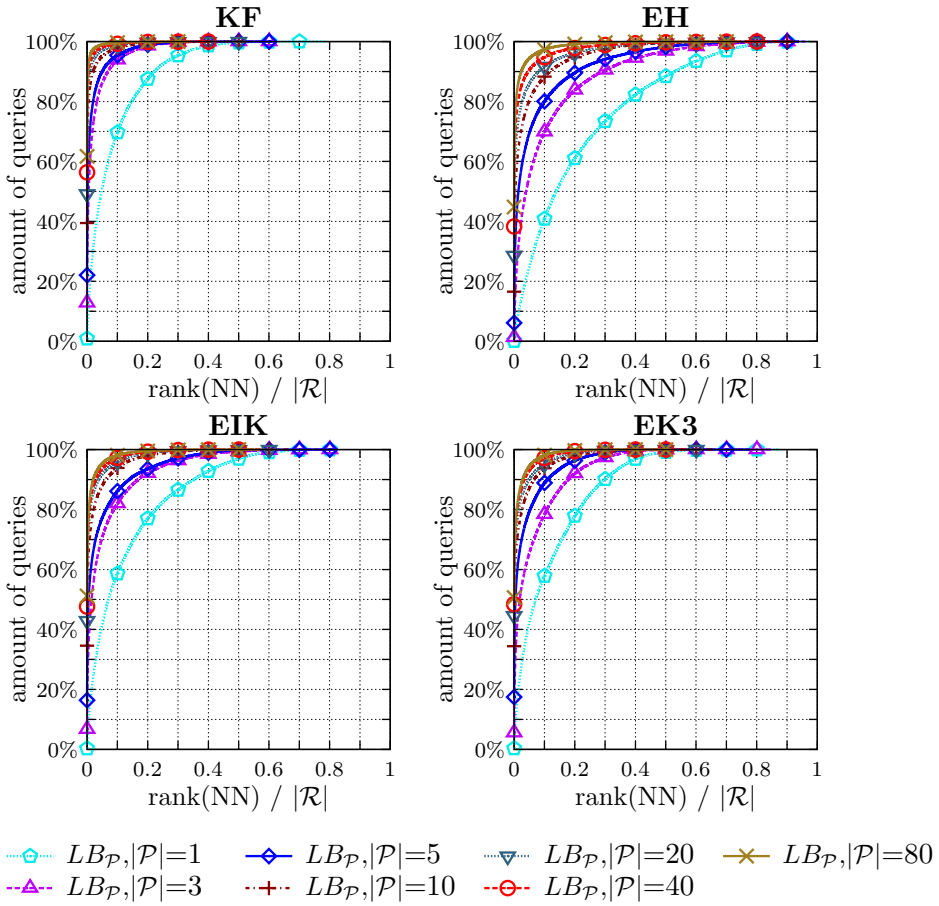


Figure 9.1 – Amount of queries whose actual nearest neighbor is between the smallest values of $LB_{\mathcal{P}}$ for $|\mathcal{P}| \in \{1, 3, 5, 10, 20, 40, 80\}$.

Figure 9.1 compares the approximation performance of $LB_{\mathcal{P}}$ under the four configurations using different sets of pivots with sizes $|\mathcal{P}| \in \{1, 3, 5, 10, 20, 40, 80\}$. The figure depicts the amount of queries in which $LB_{\mathcal{P}}$ ranks the correct NN between the $T \cdot |\mathcal{R}|$ smallest values. Because the pivot selection algorithm has a random component, the plotted value is obtained by averaging the result of many sets of pivots with the same size. Note that the ideal result is located in the upper left corner, where $LB_{\mathcal{P}}$ would rank the correct NN in the first place for 100% of queries.

The figure proves the assumptions that the lower bounds for the nearest neighbors are usually between the smallest lower bounds and that the quality of the approximation depends on the number of pivots, i.e., larger sets of pivots obtain (on average) better approximations than smaller sets. However, the sets of pivots show diminishing returns, e.g., the improvement from 5 to 10 pivots is higher than the improvement from 10 to 20, which are higher than the improvement from 20 to 40, and so on. The quality of the approximation also depends on the configuration: $LB_{\mathcal{P}}$ produces better approximations for **KF**, which is consistent with its lower ρ. On the other hand, $LB_{\mathcal{P}}$ produces worse approximations for **EH** and **EIK**, also consistently with their higher ρ. In general, the approximation parameter $T \geq 1\%$ achieves reasonable high accuracy.

Figure 9.2 depicts the time spent by the approximate search in proportion to the time spent by the linear scan. The time spent by the approximate search depends on three factors: the cost of computing $LB_{\mathcal{P}}$ (which depends linearly on $|\mathcal{P}|$), the cost of computing d (C_d, which depends on size of descriptors), and the amount of distance evaluations T. The search is divided in two rounds,

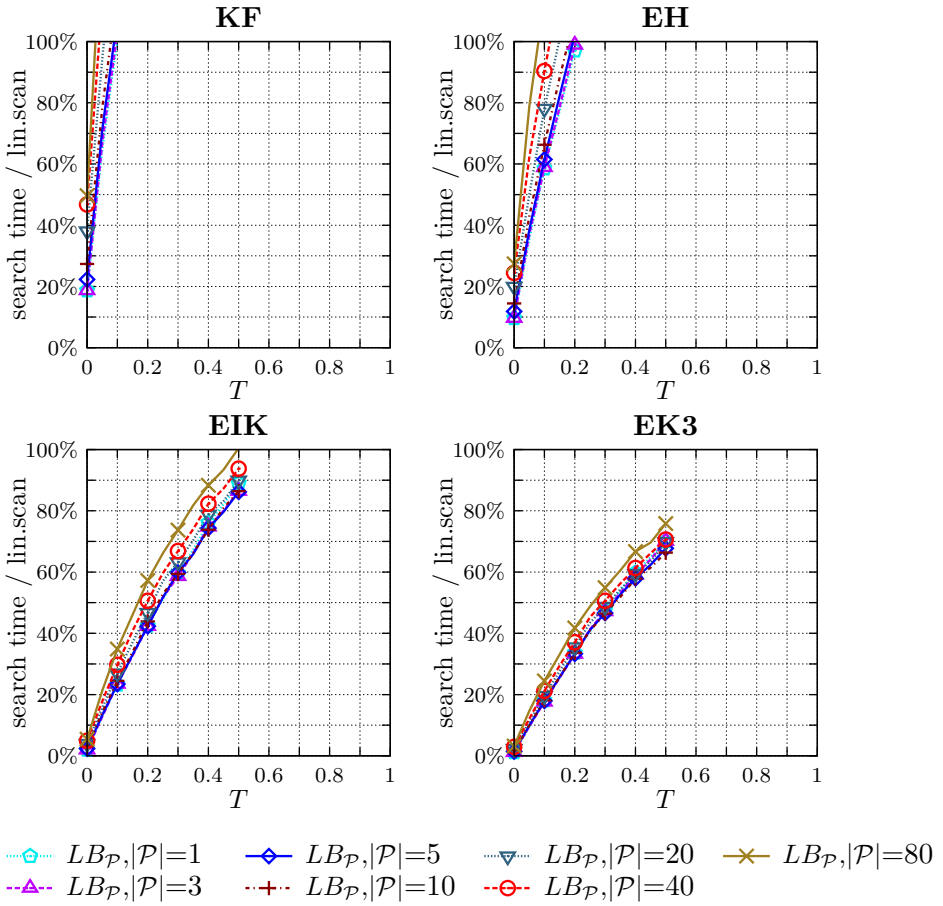


Figure 9.2 – Computational time of approximate searches with pivots compared to linear scan.

the first round computes $LB_{\mathcal{P}}$ for every object in $\mathcal{R}$ and then selects the smallest T objects, while the second round computes d for T objects. Therefore, the expected cost of the approximate search is $\mathcal{O}(|\mathcal{R}|\cdot|\mathcal{P}|+|\mathcal{R}|\cdot\log T+T\cdot C_d)$, compared to the expected cost of linear scan which is $\mathcal{O}(|\mathcal{R}|\cdot C_d)$.

The figure shows the search times grow almost linearly with T. The rate of growth is given by the cost of selecting T objects plus computing T evaluations of d. In the case of time-inexpensive distances the cost of selecting the T minimum distances dominates the times. In fact, fixing $T = 0.1$ implies creating and updating a heap with size 21,147 ($T \cdot |\mathcal{R}|$ objects) which for **KF** turns out to be more expensive than directly computing L_1 distances between 99-d vectors. In the case of time-expensive distances, the cost of maintaining a heap worth the saved distance computations, hence producing much better improvement in efficiency compared to linear scan. In fact, in **EK3** an approximate search using 10 pivots and $T = 5\%$ is able to retrieve the correct nearest neighbor in 90% of queries while reducing the search time to 10% of the time spent by a linear scan.

In summary, the approximate search with pivots enables to perform k-NN searches using combined distances in large datasets achieving high effectiveness. This search was a key algorithm for our participation at TRECVID: in TRECVID 2010 we used a configuration of T=0.001 and $|\mathcal{P}|$=9, and for 2011 we adjusted the parameters to T=0.01 and $|\mathcal{P}|$=5. This tuning highly improved the effectiveness of the search with only a minor change in the search time.

Configuration			Histogram of distances H_d				
Segm.	d	Descriptor		max	μ	σ	ρ
S1	L_1	**SF6**		3169	2016	281	25.8
S5	L_1	**SF6**		3204	2015	281	25.7
S2	L_1	**SF4**		3070	2015	283	25.4
S1	L_1	**SF2**		3127	2003	293	23.4
S5	L_1	**SF1**		3122	1999	298	22.5

Table 9.2 – Metric spaces defined by different local descriptors and segmentations.

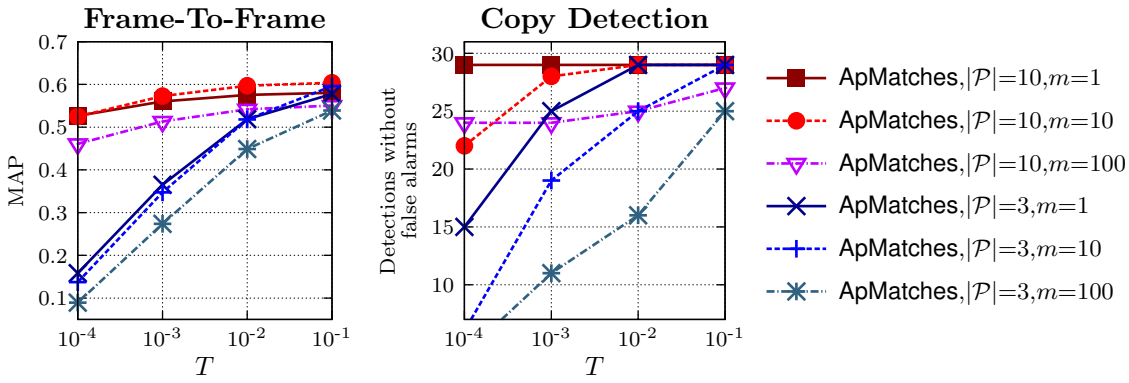


Figure 9.3 – Effectiveness of approximate search with pivots for local description **SF4** and segmentation **S3** varying $|\mathcal{P}|$, m and T.

Approximate search with pivots for local descriptors

The following experiments analyze the performance of ApMatches function, which is our approach for approximate search with pivots for local descriptors. The performance of ApMatches depends on the search parameter m (amount of nearest neighbors to retrieve for each local vector), and approximation parameter T.

Table 9.2 shows five samples of metric spaces defined by different segmentations and local descriptors. The table shows these metric spaces are fairly similar, with a small increase in the intrinsic dimensionality when the frames are reduced. The values for ρ are higher than those at Table 9.1. We chose to compare SIFT vectors with L_1 because it consistently achieves higher effectiveness than L_2 (see Section 8.3).

The following experiment measures the impact of parameters $|\mathcal{P}|$, m, and T on detection performance. Figure 9.3 shows the performance of ApMatches using parameters $|\mathcal{P}|$={3, 10}, m={1, 10, 100}, and T={0.0001, 0.001, 0.01, 0.1}. The segmentation is fixed to **S3** and local description to **SF4**. The figure shows that m=1 achieves the best overall effectiveness. Although m=10 achieves a slightly better MAP for $|\mathcal{P}|$=10, m=1 detects more copies without false alarms. A $\mathcal{P}$ with more pivots improves the effectiveness, however the difference in effectiveness between $|\mathcal{P}|$=3 and $|\mathcal{P}|$=10 is reduced as T increases.

The next experiment measures the performance of approximate search for local descriptions **SF2**, **SF4** and **SF6** fixing parameters m=1 and $|\mathcal{P}|$=3. The performance of ApMatches is compared

with Matches. Figure 9.4 shows the frame-to-frame effectiveness and detection effectiveness achieved by each configuration together with the total time spent in the $|\mathcal{Q}|$ searches. The effectiveness of Matches is also shown in Figure 7.5 on page 87. The influence of T in the effectiveness is shown, where T=0.01 achieves a convenient balance between effectiveness and efficiency. In fact, ApMatches with T=0.01 outperforms Matches detecting 30 out of 31 copies instead of 28, and furthermore it is one order of magnitude faster.

The experiment shows that ApMatches can outperform Matches. The fact that an approximate search outperforms the exact search maybe counterintuitive. The reason for this behavior is that the matched pairs in ApMatches are stricter than the matched pairs in Matches. For instance, given two segments q and r, Matches may match local vector x from q with local vector y from r when y is the nearest neighbor between the vectors in r and $d(x, y)$ satisfies the distance ratio s. It considers every segment independently of the others, thus the value of Matches(q, r) does not affect the value of Matches(q, s), for another segment s. On the other hand, ApMatches may match vectors x and y when y is the nearest neighbor considering the whole collection of vectors $\mathcal{R}^L$ (m=1). Therefore, the value of ApMatches(q, r) can indeed affect the value of ApMatches(q, s) because the vectors in q paired with a vector in r cannot be also paired with another vector in s.

The experiment also shows a limit for the approximate search on SIFT vectors. Given a configuration that achieves good effectiveness but requires too much time, the search time can be reduced by decreasing the approximation parameter. However, if T decreases below 0.001 the search time is slightly reduced but the effectiveness may be ruined. An alternative is to represent each frame by less vectors (e.g., to use **SF**4 instead of **SF**2), which also decreases the effectiveness but it gives more gain in efficiency. However, a reduction beyond **SF**6 also highly impacts the effectiveness. In fact, for very small frames a global descriptor is more meaningful than local descriptors.

In summary, the approximate search with pivots is able to both reduce the search times and achieve high detection effectiveness. However, its performance has a limit mainly determined by: 1) the amount of local vectors to search in (which in turn depends on the dataset size), and 2) the intrinsic dimensionality of the metric space created by the descriptor (which for SIFT vectors and L_1 is consistently high). In a medium-sized dataset, like MUSCLE-VCD-2007, it can successfully reduce search times by about one order of magnitude while still achieving high effectiveness, but it is an open issue to successfully use it on larger collections (like the TRECVID datasets). Moreover, it should be noted that in the case of approximate searches using SIFT descriptors, the multidimensional indexes can achieve much better effectiveness-versus-efficiency tradeoff than metric indexes. In fact, the main benefit of using metric indexes is to improve the efficiency of complex distances, hence a time-inexpensive distance between 128-d vectors does not provide the best scenario to evaluate them. More details on this issue are given in Section 9.3 and Section 12.2.

9.2 Efficiency for Streams of Exact Searches

In our CBVCD system, the similarity searches are resolved for every segment in a query video, thus it can be expected that two consecutive segments from the same video are similar (especially for fine-grained segmentations). In that case the distance between i^{th} and $(i{+}1)^{\text{th}}$ segments should be small, and also the nearest neighbor of i^{th} segment may be close to the nearest neighbor of $(i{+}1)^{\text{th}}$ segment.

Based on this idea, the efficiency of the search can be improved by using the i^{th} query segment as a dynamic pivot to resolve the $(i{+}1)^{\text{th}}$ search. This requires a dynamic pivot table

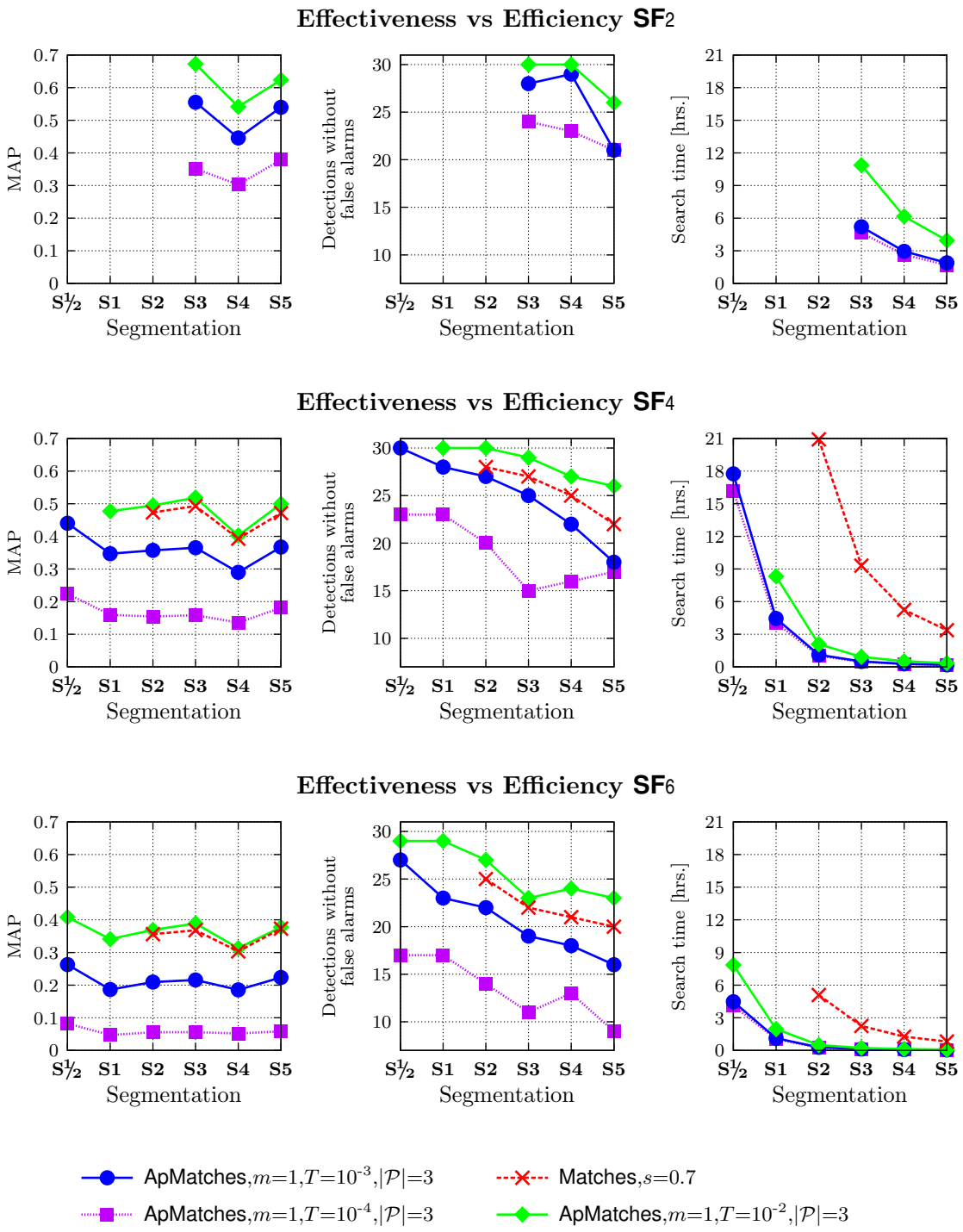


Figure 9.4 – Performance of approximate search for local descriptors and exact search.

that stores the distances between the previous query segment and every reference segment, hence if query segments are nearby, the stored distances may produce tight lower bounds that discard most of the distances in the $(i+1)^{\text{th}}$ search. However, in that case the $(i+1)^{\text{th}}$ query segment will be a poor pivot for the $(i+2)^{\text{th}}$ segment, because the dynamic pivot table would have stored only a few distances. Then, to minimize that "min-max" effect we will use a sliding window with the last p query segments, instead of just the last one.

The D-file is a dynamic structure that can be used for this approach (see Section 3.3.1). However, as we show in the experimental section, the D-file suffers from high internal complexity. The main problem arises when the distance function is not time-expensive. In that case, the internal complexity associated with the hash function and collision resolution dominates the search times rendering it unviable to use in many scenarios. In order to solve this problem, we introduce the Snake Table that preserves the idea and advantages of D-file and D-cache, but exhibits lower internal complexity.

9.2.1 Snake Table

The life cycle of the Snake Table is as follows: First, when a new session starts, an empty Snake Table is created and associated with it. When a query object q_1 is received, a k-NN search is performed. The distances between q_1 and the objects in the collection are added to the Snake Table, and the result is returned. When a new query object q_i is received, a k-NN is performed using the previous query objects $q_{i-p}, ..., q_{i-1}$ as pivots to accelerate the search. Finally, when the session ends, the Snake Table is discarded. Therefore, like D-file and unlike most of MAMs, the Snake Table is a session-oriented and short-lived MAM, see Figure 9.5.

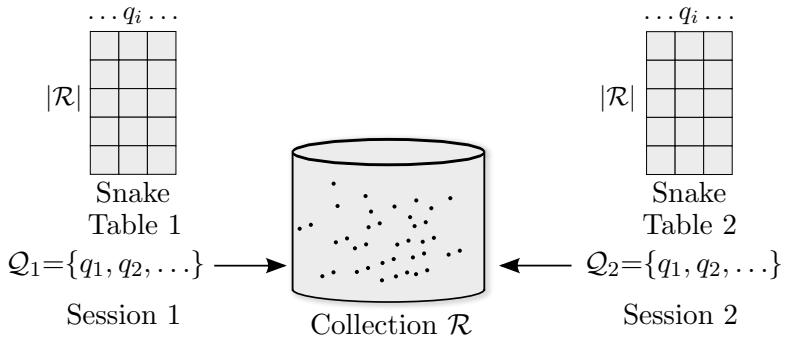


Figure 9.5 – Snake Tables created for stream of queries $\mathcal{Q}_1$ and $\mathcal{Q}_2$.

The Snake Table is implemented with a fixed-size $|\mathcal{R}|\times p$ matrix used as a dynamic pivot table. As in LAESA, the j^{th} row in the dynamic pivot table represents the object o_j in $\mathcal{R}$ and contains the distances between o_j and up to p previously processed query objects. Each cell in the j^{th} row of the table contains a pair $(q, d(q, o_j))$ for some query object q (not necessarily in order). When processing a new query object q_i, the lower bound $LB_{\mathcal{P}}(q_i, o_j)$ is computed, with $\mathcal{P}$ dynamically determined by the query objects and distances in the j^{th} row. As in the exact search with pivots, the object o_j is discarded when $LB_{\mathcal{P}}(q_i, o_j)$ is greater than the distance between q_i and the current k^{th} nearest neighbor candidate (obtained between o_1 and o_{j-1}). If o_j is not discarded, the actual distance $d(q_i, o_j)$ is computed, added to some cell in the j^{th} row, and the NN candidates are updated if necessary.

We present three different replacement strategies to assign a distance $d(q_i, o_j)$ to one of the p cells in the j^{th} row:

1. Sparse/FIFO: Each query q_i picks a column in round-robin mode, i.e., the distance $d(q_i, o_j)$ is stored in the (i mod p) column of the j^{th} row, eventually replacing the stored distance $d(q_{i-p}, o_j)$. If the distance was not evaluated because it was discarded by $LB_{\mathcal{P}}(q_i, o_j)$ then the corresponding cell is not used. This behavior can be implemented following two options: 1) the cell is updated with an ∞ distance; or 2) the cell is left unmodified, but before any read operation the query stored in the cell is matched with the last query for that column (the experimental section uses the latter). This strategy produces sparse rows containing at most p distances between $d(q_{i-p}, o_j)$ and $d(q_i, o_j)$. As a consequence, if p=1 and most of the distances for q_{i-1} were discarded, then q_i will achieve poor efficiency. In order to diminish this "min-max effect" a larger p or a different strategy should be chosen.

2. Compact/Unsorted: The distance $d(q_i, o_j)$ is compared to every distance in the j^{th} row and the cell with the minimum distance is replaced, independently of its position in the row. With this strategy, every row stores the highest p distances between $d(q_1, o_j)$ and $d(q_i, o_j)$ that have not been discarded.

3. Compact/FIFO: The distance $d(q_i, o_j)$ is stored in a cell chosen in an independent round-robin for every row. With this strategy, the j^{th} row stores the last p computed distances for o_j, discarding the old ones. $LB_{\mathcal{P}}$ starts its evaluation from the last stored distance and goes backwards, therefore favoring the most recently stored distances.

For strategy Sparse/FIFO, distances $d(q_i, q_j)$ with $j \in \{i-p, ..., i-1\}$ are calculated and stored in memory at the beginning of every search. For strategies Compact/Unsorted and Compact/FIFO, distances $d(q_i, q_j)$ with $j \in \{1, ..., i-1\}$ are calculated on-demand by $LB_{\mathcal{P}}$. Note that Sparse/FIFO uses a global index to mark the current column where computed distances are stored, Compact/FIFO uses an independent index for each row, and Compact/Unsorted does not need any index because it determines the cell to replace when it calculates each lower bound.

D-file uses a combination of Sparse/FIFO and Compact/Unsorted. It always replaces an old distance (older than q_{i-p}), but if there is no old distance in the collision interval, it replaces the worst distance (defined as the distance closer to the median or some predefined percentile).

The performance achieved by these three replacement strategies are compared in the experimental section. However, it should be stressed that despite the replacement strategy, the overall performance of the Snake Table mainly depends on the distribution of the query objects.

9.2.2 Snake Distribution

The Snake Table is intended to be used when the query objects in a stream fit a "snake distribution". Intuitively, we define that a set of objects fits a snake distribution when the distance between two consecutive objects in $\mathcal{Q}$ is small compared to the median distance between any two objects (see Figure 9.6). In order to measure and compare the degree of fitness, we define an indicator using the histogram of distances of d for $\mathcal{Q}$ and $\mathcal{R}$.

Definition 7 (Difference Δ) *Let F_1 and F_2 be two cumulative distributions, the difference Δ between F_1 and F_2 is defined as:*

$$\Delta(F_1, F_2) = \int_0^\infty F_1(t) - F_2(t)\, dt$$

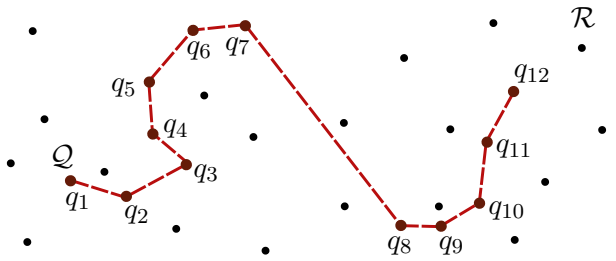


Figure 9.6 – Stream of queries $\mathcal{Q}$=$\{q_1, ..., q_{12}\}$ with a snake distribution. Most distances $d(q_i, q_{i+1})$ are smaller than $d(x, y)$ for randomly selected pairs x,y in $\mathcal{R}$.

The Difference Δ is meaningful only when both F_1 and F_2 originate from the same metric space. Note that $\Delta(F_1, F_2)$ is greater than zero when the distances accumulated by F_1 are smaller than distances accumulated by F_2.

Definition 8 (Snake Distribution) *Let $\mathcal{M} = (\mathcal{D}, d)$ be a metric space, let $\mathcal{R} \subset \mathcal{D}$ be the collection of objects, and let $\mathcal{Q} \subset \mathcal{D}$ be a set of m query objects $\mathcal{Q} = \{q_1, ..., q_m\}$. Let F be the cumulative distribution of $d(x, y)$ with random pairs $x, y \in \mathcal{Q} \cup \mathcal{R}$, p be a number between 1 and m-1, and $F^p_{\mathcal{Q}}$ be the cumulative distribution of $d(q_i, q_{i-p})$ $\forall$ $i \in \{p+1, ..., m\}$. $\mathcal{Q}$ fits a snake distribution of order p if $\Delta(F^p_{\mathcal{Q}}, F) > s$, for some threshold value $s \in \mathbb{R}^+$.*

Note that when both $\mathcal{Q}$ and $\mathcal{R}$ are random samples of $\mathcal{D}$ without any special ordering (i.e., the i^{th} sample does not depend on previous samples), then $\Delta(F^p_{\mathcal{Q}}, F) \approx 0$. When a distribution fits a Snake Distribution of order 1 to p then a Snake Table can be created with a sliding window containing up to p query objects.

9.2.3 Evaluation

The following experiments compare the improvements in the efficiency achieved for six index structures using either static pivots or dynamic pivots. The number of pivots for each index varies between 1 and 20. The evaluated indexes are:

- **D-file**: It uses a D-cache with a fixed size hash table of $n = 10 \cdot |\mathcal{R}| \cdot p$ cells, collision interval 1, and hash function $h(q_i, o_j)$=$(rnd_i * rnd_j)$ mod n, where rnd_i and rnd_j are unique random IDs assigned to each object.
- LAESA-based: Following its definition, LAESA does not require any information of the query objects, but for a fair comparison, we allow LAESA to use $\mathcal{Q}$ in the selection process. **LaesaQ** chooses p static pivots from $\mathcal{Q}$, and **LaesaR** chooses p static pivots from $\mathcal{R}$. Both selections are performed using the selection algorithm depicted in Algorithm 9.2.
- Snake Table: We test the three strategies depicted in Section 9.2.1. **SnakeSF** uses Sparse/FIFO strategy (sparse row with the last p queries), **SnakeCU** uses Compact/Unsorted strategy (compact row containing p prior query in no particular order), and **SnakeCF** uses Compact/FIFO strategy (compact row with the last p evaluated distances for each object).

We evaluate these indexes under four configurations: **KF**, **EH**, **EIK**, and **EK3**, see Table 9.1. These configurations are split into two groups: Group 1 (**KF** and **EH**) contains the

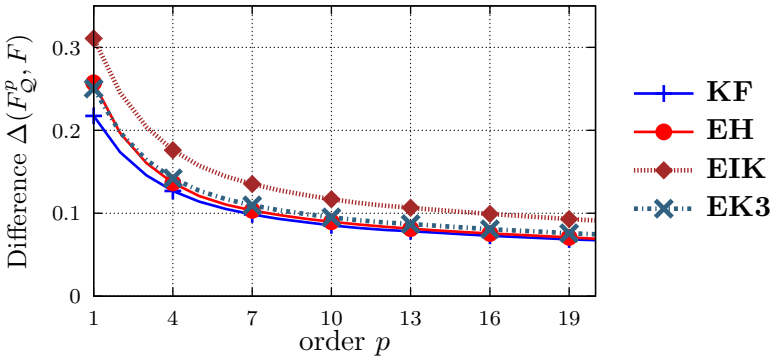


Figure 9.7 – Snake distribution of order p for the four configurations.

configurations where the linear scan takes less amount of time, and Group 2 (**EIK** and **EK3**) contains the configurations in which the linear scan is slower by one order of magnitude.

Snake Distribution

Figure 9.7 depicts the snake distribution of order p for the four configurations. The value of difference $\Delta(F_Q^p, F)$ for $p \in \{1, ..., 20\}$ is shown. The four configurations present a difference Δ higher than zero, hence the streams of queries fit a snake distribution, i.e., distances between q_i and q_{i-p} are smaller than distances between random pairs. The four configurations present higher fitness for p close to 1, and as p increases the snake distributions tend to disappear. As shown in the following experiments, the different configurations present satisfactory results with p roughly between 1 and 5 pivots.

Group 1

Figure 9.8 shows the efficiency achieved by the six indexes under the two fastest configurations. It depicts the amount of distance computations and search time achieved by each index, as a proportion of the linear scan, when the number of pivots varies between 1 and 20. These values only consider the online phase, i.e., they do not include the distance computations required by LAESA to select pivots and build the pivot table.

The disparity between saved distances and saved time reveals the internal complexity of each index. Because these configurations use fast distances, an index must have low internal complexity in order to outperform the linear scan, otherwise it will be faster to directly compute each distance instead of trying to discard them.

In the case of static pivots, **LaesaQ** shows an almost identical performance as **LaesaR**. This result implies that knowing a priori the set of queries does not produce a noticeable improvement in LAESA performance. The instability in LAESA for consecutive p is due to the random nature of pivot selection. In order to reduce this effect for each number of pivots we averaged the result achieved by four different sets of pivots.

Configuration **KF** has low intrinsic dimensionality, therefore it is expected that pivots will discard most of the distance computations. In this case, LAESA achieves the best search times even

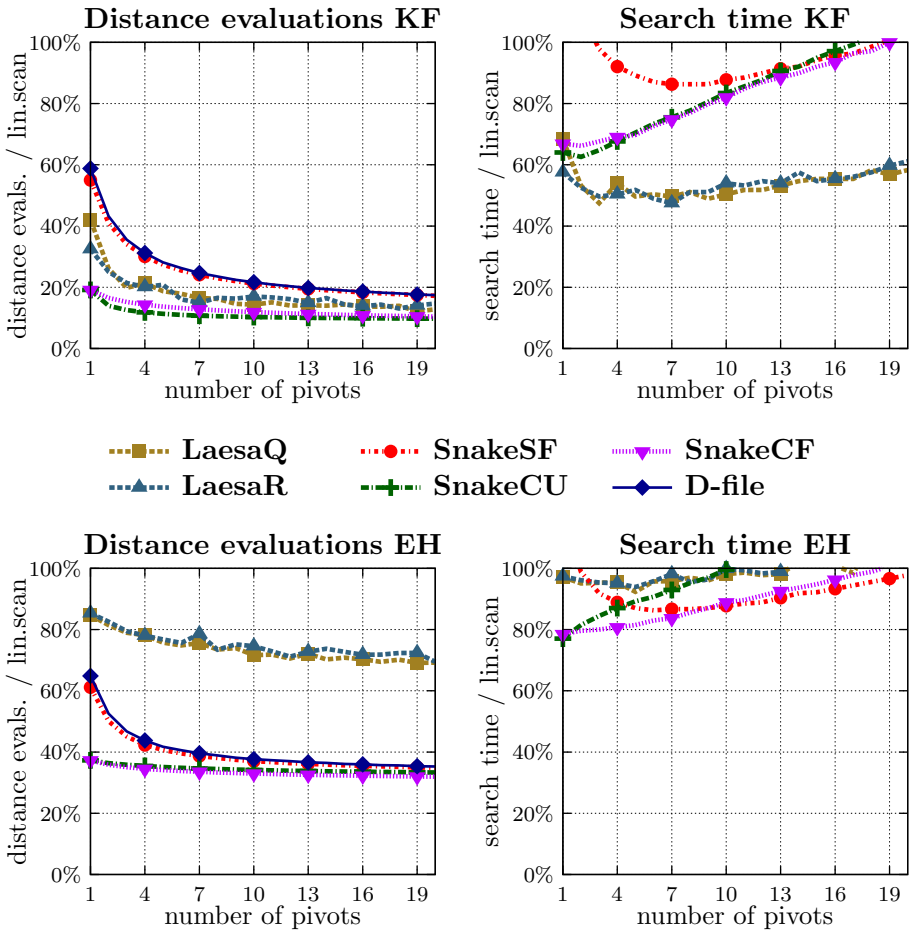


Figure 9.8 – Search time and distance evaluations for **KF** and **EH** (Group 1).

though the Snake Table discards slightly more distances. In fact, **LaesaQ** and **LaesaR** discarding 80% of distance computations can reduce by about 50% the search time, whereas **SnakeCU** and **SnakeCF** also discarding 80% of distance computations reduces by less than 40% the search time. This difference reveals that a static pivot table has lower internal complexity than a dynamic pivot table, which is an expected result due to the overhead produced by updating the pivot table.

Configuration **EH** has higher intrinsic dimensionality, therefore it is expected that pivots will discard less distance computations than in **KF**. In fact, both **LaesaR** and **LaesaQ** discard just about 25% of distance computations and search time is reduced by less than 10%. On the other hand, Snake Table can profit from the snake distribution in order to discard more distance computations: both **SnakeCU** and **SnakeCF** discard more than 60% distance computations, reducing the search time by up to 20%. This result shows the snake distribution can reduce search times even in scenarios with high intrinsic dimensionality and fast distances.

Comparing the Snake Table replacement strategies, compact rows from **SnakeCU** and **SnakeCF** show superior performance than sparse rows from **SnakeSF**. The "min-max" effect described in Section 9.2.1 becomes apparent: **SnakeSF** cannot achieve high performance when using one and two pivots because every discarded distance implies an empty cell in the pivot table, which affects the performance for subsequent pivots. This undesired effect of Sparse/FIFO strategy decreases as the number of pivots increases. On the other hand, Compact/Unsorted and Compact/FIFO strategies do not suffer the "min-max" effect because the compact rows prevent harming the performance with empty cells.

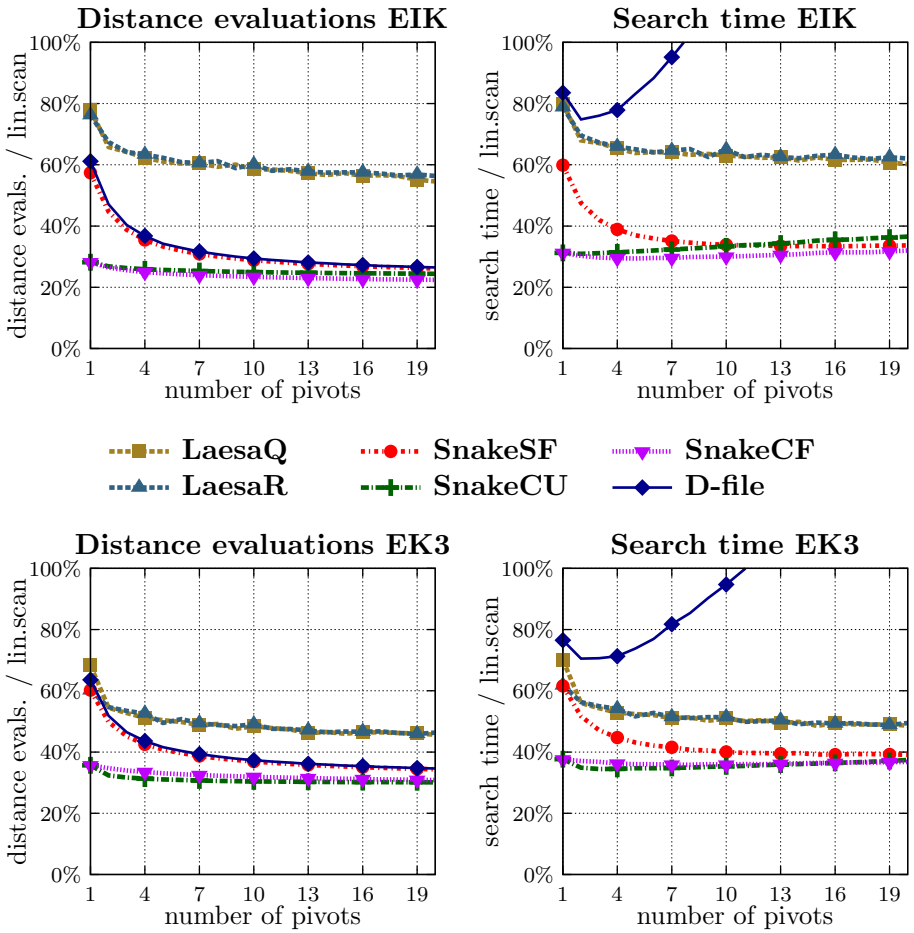


Figure 9.9 – Search time and distance evaluations for **EIK** and **EK3** (Group 2).

The **D-file** presents very high internal complexity: it discards as many distance computations as **SnakeSF**, but the search times are always higher than linear scan. In fact, the search using **D-file** takes more than two times the linear scan for **EH**, and more than three times the linear scan for **KF**. This high internal complexity is due to the cost of computing a hash function instead of directly accessing some table cell (as **SnakeSF** does). Moreover, **D-file** discards slightly less distance computations than **SnakeSF** due to the existence of collisions in the hash table which may overwrite already computed distances.

In summary, LAESA can achieve high performance in scenarios with fast distance and low intrinsic dimensionality. However, in scenarios with higher intrinsic dimensionality, the Snake Table achieves the best performance due to its better pivot selection (profiting from the snake distribution).

Group 2

Figure 9.9 shows the performance achieved by the six indexes under configurations with time-expensive distances. It depicts the amount of distance computations and time spent by the search, as a proportion of the linear scan. As in previous experiment, the values for LAESA do not consider the cost of building the index structures.

These scenarios produce a higher correlation between discarded distances and search time savings. In fact, LAESA search time is less affected by the number of pivots, while Snake Table starts

to increase the search time from about five pivots. Besides, **D-file** using a few pivots outperforms the linear scan, however as the number of pivots increases the search times quickly exceeds the linear scan due to the high cost of computing the hash function for each pivot.

The existence and the exploitation of a snake distribution in the query objects becomes a remarkable approach to improve the efficiency of exact searches. In the case of **EK3**, LAESA reduces the search time by 50%, while the Snake Table can reduce the search times by more than 60%. In the case of **EIK** the difference is even greater, LAESA reduces the search time by 40%, while the Snake Table can reduce the search times by up to 70%.

In summary, in scenarios with a time-expensive distance and a query set that fits a snake distribution, the Snake Table achieves much higher performance than LAESA. The best performances for Stake Table are achieved between three and five pivots and using Compact/Unsorted or Compact/FIFO replacement strategies.

Discussion

The high performance of the Snake Table compared with LAESA and D-cache is due to its properties of dynamic selection of pivots and low internal complexity. The Snake Table is able to exploit snake distributions in order to reduce the search time for both fast and time-expensive distances even in spaces with high intrinsic dimensionality. In particular, the Snake Table is a better alternative than D-file in the tested scenarios.

The Snake Table presents an approach to index spaces when consecutive queries are similar to each other. This behavior usually arises in content-based video retrieval (when the queries are consecutive keyframes), interactive multimedia retrieval systems (when the user selects a new query object from the answers of a previous query), and similarity searches using local descriptors. In a more general domain, given an unsorted set of queries, the test of snake distribution presented in this work may be useful to determine an optimal ordering of queries which will achieve a high performance in the Snake Table.

One usage of the Snake Table is to create an index for each stream of queries. When a user starts a session, an empty Snake Table is associated with it. As the user performs queries with snake distribution, the index improves its performance because it will select pivots close to following queries. However, the Snake table is not memory efficient as it requires space proportional to the size of the dataset and to the number of sessions connected. This approach is more suitable for medium-sized databases with long k-NN streams. Moreover, because it does not need to use a central shared index structure, it is also suitable for highly dynamic datasets.

On one hand, pivots in a sliding window with snake distribution satisfy one desirable property: they should be close to either the query or the collection objects. On the other hand, those pivots do not satisfy another desirable property: they should be far away from each other. Hence, using a Snake Table with many pivots will only increase the internal complexity without increasing the efficiency because pivots will be mostly redundant. In order to overcome this issue, the Snake Table and LAESA can seamlessly be combined by with a unique pivot table containing both static and dynamic pivots. Moreover, the SSS algorithm can also be combined with the Snake Table by fixing one pivot when it is far away from all the previous ones. This combined approach enables the profiting from both dynamic pivots close to queries and non-redundant static pivots.

LAESA can benefit from multi-core architectures by sharing the pivot table and resolving

Name	Segm.	d	Descriptor	dim.	ρ	MAP
AU	**S1**	L_1	**AU**160	160	7.5	0.734
EH	**S1**	L_1	**EH**^{t}4x4-10	160	9.6	0.664
IH	**S1**	L_1	**IH**^{t}1x4-rgb-3x16	192	8.6	0.501
KF	**S1**	L_1	**KF**^{t}11x9	99	4.0	0.510
OM	**S1**	L_1	**OM**^{t}9x9	81	6.2	0.339
SF	**S5**	L_1	**SF**6	128	24.6	—

Table 9.3 – Configurations used in the comparison between multidimensional and metric indexing.

each query in different threads. In the case of Snake Table, in order to efficiently resolve parallel queries we recommend partitioning the queries into independent subsets, and resolving each subset by a Snake Table in an independent thread.

9.3 Comparison with Multidimensional Indexes

In this section we evaluate the performance of the presented techniques compared to state-of-the-art search techniques. The main strength of Approximate Search with Pivots and Snake Table is they can work with general distances and any kind of descriptors, as long as the distance satisfies the metric properties. On the other hand, most state-of-the-art techniques are designed to work with multidimensional vectors and Euclidean or Manhattan distance. Hence, because that scenario is a particular case of metric spaces, we will use configurations based on multidimensional vectors in order to make the comparison. However, accordingly to Section 3.4, the metric approach achieves lower performance in those scenarios than multidimensional indexes.

Muja and Lowe [2009] presents the FLANN library (Fast Library for Approximate Nearest Neighbors), which efficiently implements different indexing techniques from multidimensional vectors. In particular, we evaluate two families of algorithms: kd-tree and k-means tree (described in Chapter 3).

In the following experiments we evaluate multidimensional and metric indexes under different descriptors. Table 9.3 summarizes the descriptors: four global descriptors **EH**^{t}4x4-10, **OM**^{t}9x9, **KF**^{t}11x9, and **IH**^{t}1x4-rgb-3x16, one acoustic descriptor **AU**160, all of them were computed from preprocessed videos, segmentation **S1**, considering query videos from ST1 and ST2 collection; and local descriptor **SF**6 computed from preprocessed videos, segmentation **S5**, and query videos from ST2 collection. The table shows the number of vectors for the query set $\mathcal{Q}$, the number of vectors for the reference dataset $\mathcal{R}$, the vector dimensionality, the intrinsic dimensionality, and the MAP achieved by the linear scan according to the ground-truth (already shown in Figure 7.3 on page 82).

In this experiment the linear scans are resolved by FLANN, which provides an efficient implementation of distance computations. This produces a much faster scan than the used by previous experiments. That issue is the reason the results from the following experiments are not consistent with results from previous experiments in this chapter.

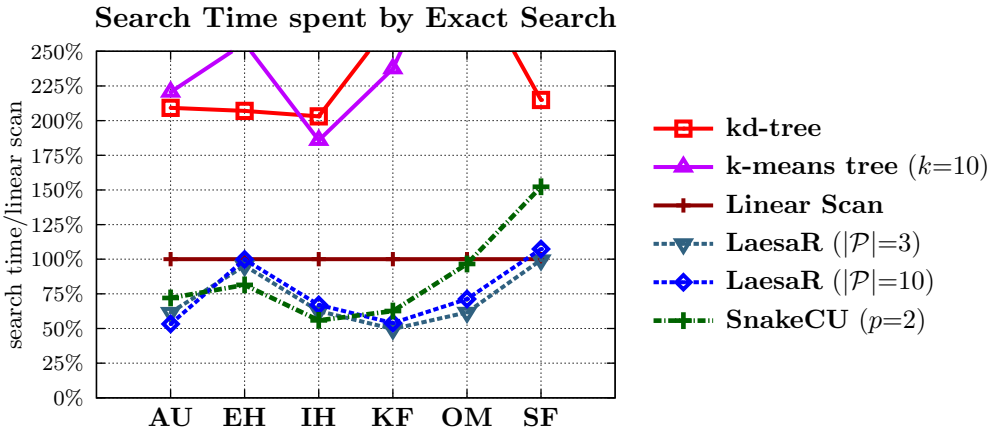


Figure 9.10 – Search time spent by the exact search implemented by different multidimensional and metric indexes.

9.3.1 Exact search

This experiment evaluates the performance achieved by different indexes for exact searches, i.e. when the search is restricted to retrieve identical nearest neighbors as the linear scan. Therefore, in order to compute effectiveness the ground-truth is set to the nearest neighbor retrieved by the linear scan.

Figure 9.10 summarizes the performance for the multidimensional indexes: kd-tree, and k-means tree (with branching k=10 per node); and for the metric indexes: LAESA (using 1, 3, and 10 pivots selected by SSS), and the Snake Table (using 2 dynamic pivots replacement strategy 2). The figure compares the time spent by the exact search relative to the time spent by the linear scan under the six tested configurations.

The experiment shows the exact search is not favorable for multidimensional indexing. In fact, in every tested scenario multidimensional indexes are slower than the linear scan, with the exception of k-means tree at **IH** configuration (4% faster than the linear scan). This result evinces the impact of curse dimensionality in multidimensional indexing: in order to resolve an exact search, every tree node that potentially may contain an object closer than the NN must be visited. Under high dimensional spaces most of space regions have some intersection with a query region, forcing the search to visit almost the whole tree. We must note that an early termination of the search indeed retrieves the correct NN most the times (see next experiment), but forcing the search to guarantee the highest effectiveness inevitably ruins the efficiency.

On the other hand, metric indexes are able to reduce the search time by about 20% to 60% without affecting the effectiveness. As said in previous experiments, the number of pivots that must be selected by LAESA in order to achieve the best efficiency depends on two factors: intrinsic dimensionality, and cost of the distance (which in turn depends on vector dimensionality). The former influences the quality of the lower bounds: a high intrinsic dimensionality forces to select many pivots in order to save some distance computations. The latter sets an upper bound for the internal cost: a cheap distance forces the use of only a few pivots to achieve some gain compared to the linear scan. The tested scenarios use a cheap distance, therefore the best balance is achieved by few pivots. In fact, the figure shows that one and three pivots always outperforms ten pivots. In scenarios with higher intrinsic dimensionality (**EH** and **IH**) the Snake Table is able to outperform LAESA because it profits from the snake distribution, reducing the impact of

intrinsic dimensionality in the quality of lower bounds. The Snake Table achieves low performance at **SF** (6% slower than the linear scan) because SIFT descriptors do not produce a profitable snake distribution.

In summary, the exact search shows that metric indexing can outperform the linear scan without affecting effectiveness. On the other hand, multidimensional indexing necessarily has to reduce the effectiveness in order to achieve high efficiency. The next experiment analyzes this issue. An interesting issue that needs more investigation is that multidimensional indexing and metric indexing do not show the same behavior for different configurations. For example, metric indexes achieve a relatively similar performance for **IH** and **KF**, but multidimensional indexes achieve much better performance at **IH** than **IH**. A similar issue occurs between **AU** and **EH**: almost same performance for metric indexes, but **AU** is more difficult for multidimensional indexes than **EH** .

9.3.2 Approximate search

In this experiment we compare the performance of multidimensional and metric indexes for approximate searches. Approximate search algorithms rely on one or more parameters to control the effectiveness-vs-efficiency tradeoff. The evaluation method fixes approximation parameters, performs the search, and measures the search time and the amount of identical nearest neighbors compared to the linear scan. Following this method for different approximation parameters we produce a curve that reveals the tradeoff between search time and effectiveness cost.

Figure 9.11 shows the effectiveness-versus-efficiency tradeoff for different indexes. The figure compares the time spent by the approximate search relative to the time spent by the linear scan, and the amount of correct nearest neighbors retrieved, under the six tested configurations. Three types of indexes are evaluated: kd-tree, using a single tree and ten trees; k-means tree, using a branching 10 and 20 per node; and LAESA, using 3, 5, 10 and 20 pivots selected by SSS. Additionally, the search time achieved by the best exact search is included for each configuration (see Figure 9.10). The exact search sets an upper bound for the search time that an approximate search can spend.

The experiment shows multidimensional indexes achieve a much better effectiveness-versus-efficiency tradeoff than metric indexes. In fact, multidimensional indexes achieve a 90/10 tradeoff (i.e., to retrieve more than 90% correct NNs requiring less than 10% the time spent by the linear scan) at every tested configuration. Moreover, in the case of **SF**, the approximate search using ten kd-trees is able to achieve a surprising 97/3 tradeoff (i.e., to retrieve 97% of correct NNs while reducing the search time to less than 3% compared to the linear scan).

On the other hand, approximate search with pivots using LAESA achieves a less satisfactory effectiveness-versus-efficiency tradeoff. In fact, the metric approximate search is able to achieve a 70/30 tradeoff (i.e., to achieve a near 70% precision requiring about 30% of the time spent by the linear scan) for **OM** and **EH** configurations, and improving up to a 80/20 tradeoff for **AU**, **IH**, **KF**, and **SF** configurations. Those tradeoffs are commonly achieved by setting an approximate parameter T between 1% and 5%, i.e., by estimating and discarding more than 95% of objects and evaluating less than 5% of distances. Increasing the effectiveness of the search may produce a large increase in search times, even achieving search times close to the exact search. Trying to improve the efficiency by reducing T beyond that interval may produce a large decrease of effectiveness, even ruining the search results.

The experiment proves the high performance that multidimensional indexes can achieve in the particular case of configurations based on single multidimensional vectors. Therefore, the metric

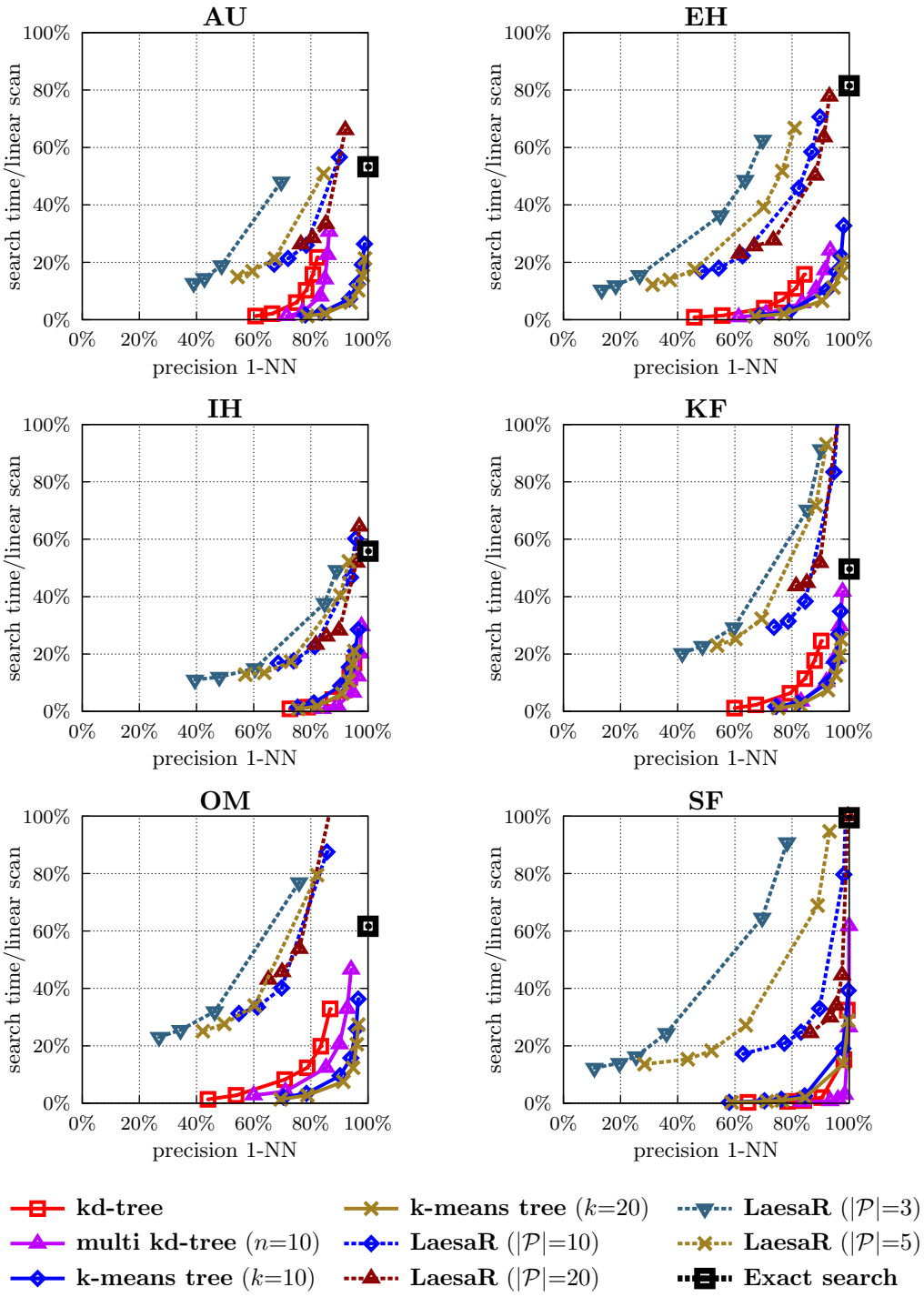


Figure 9.11 – Effectiveness-versus-efficiency tradeoff for multidimensional and metric indexes. Exact search is the best time achieved by a metric index in Figure 9.10.

Algorithm 9.5: k-NN search by distance aggregation of partial nearest neighbors.

Input: $\{\text{NN}_{d_1}, ..., \text{NN}_{d_n}\}$ partial nearest neighbors according to distances d_1 to d_n, $\{w_1, ..., w_n\}$ weights for underlying distances, k amount of NN to retrieve.
Output: the k nearest neighbors according to DistAgg.

```
NN ← new priority queue
foreach r ∈ R do
    foreach d_i ∈ {d_1, ..., d_n} do
        if ∃(r, ·) ∈ NN_{d_i} then
            retrieve (r, val_{d_i}) from NN_{d_i}
        else
            val_{d_i} ← defaultVal                          // missing object
    dist ← Σ_i w_i val_{d_i}                                // compute DistAgg
    add (r, dist) to NN
    if size of NN > k then
        remove max distance object from NN
return NN
```

approach is intended to be used in scenarios where the highest effectiveness is desired and/or it is impossible to use a multidimensional index. Two samples of those scenarios are: 1) the design of a complex distance that may combine many descriptors in order to achieve higher effectiveness, which makes the use of multidimensional indexes unviable; and 2) the size of the search space permits to resolve exact searches in reasonable search time, as is the case of global descriptors and the MUSCLE-VCD-2007 dataset.

9.3.3 Fusion of descriptors

In Chapter 8 we defined the distance $\gamma(q, r) = \sum_{i=1}^{n} w_i \, d_i(q, r)$, and we experimentally proved that γ outperforms the effectiveness of any single d_i. An efficient search based on γ requires a metric index, however underlying distances d_i are based on vectors. In the previous section we showed that multidimensional indexes present a much better tradeoff for approximate searches, thus it is desirable to use some technique to use them when vectors are present.

In this section we compare some techniques to efficiently resolve similarity searches based on a linear combination of distances. They take the result of independent k-NN searches for each of the n underlying distances (referred to as "partial" nearest neighbors), combine them in some way, and produce the final list of fused nearest neighbors. This design enables to replace the similarity search based on γ by fusing n searches, which may efficiently be resolved by multidimensional indexes. This approach requires that d_i be some distance supported by multidimensional indexes, like Minkowski distances.

The first fusion approach is the distance aggregation, which normalizes and sums the distances for the partial nearest neighbors. Algorithm 9.5 shows an implementation of the distance aggregation. It computes the nearest neighbors using the function:

$$\text{DistAgg}(q, r) = \sum_{i=1}^{n} w_i \cdot \overline{d_i}(q, r)$$

$$\overline{d_i}(q,r) = \begin{cases} d_i(q,r) & \text{iff } r \in k\text{-NN list according to } d_i \\ \mathit{defaultVal} & \text{otherwise} \end{cases}$$

The weights w_i must normalize the distances from d_i. An algorithm to automatically compute those weights is the α-normalization, described in Chapter 8.

If the n partial lists contain all the objects in $\mathcal{R}$ (i.e., they are the result of k-NN searches with k=$|\mathcal{R}|$) then DistAgg retrieves the same nearest neighbors as γ. However, computing and storing searches with k=$|\mathcal{R}|$ is usually unviable, thus partial NN lists use a small k and the fusion algorithm will miss some $d_i(q,r)$. In those cases, the algorithm replaces the missing values by a constant *defaultVal*, which should be large enough to represent an object that is farther than the k^{th} nearest neighbor.

The second fusion approach consists in directly merging the n partial NN lists and sorting them by distance. The distances must first be normalized by w_i in order to make them comparable. In case of duplicated nearest neighbors, only the occurrence with minimum distance remains and the rest is discarded. In fact, this merging algorithm computes nearest neighbors according to distance:

$$\mathsf{DistMin}(q,r) = \min_{i=1..n} \left\{ w_i \cdot \overline{d_i}(q,r) \right\}$$

The third fusion approach corresponds to the well-known rank aggregation. In this case, a voting algorithm is performed between the partial NN, where each one sums one vote weighted by its position in the list. In this experiment, we fix weights as the inverse of the rank, i.e., the first NN sums 1, the second NN sums 0.5, and so on:

$$\mathsf{RankAgg}(q,r) = \sum_{i=1}^{n} \sum_{j=1}^{k} \mathrm{vote}_{i,j}(r)$$

$$\mathrm{vote}_{i,j}(r) = \begin{cases} \frac{1}{j} & \text{iff } r \text{ is the } j\text{-th NN according to } d_i \\ 0 & \text{otherwise} \end{cases}$$

The rank aggregation does not consider the distance values from the partial NN lists, hence there is no need for any normalization. The most similar objects are the ones that achieve more votes, therefore RankAgg is a similarity function.

Figure 9.12 compares the effectiveness achieved by DistAgg, DistMin, and RankAgg when combining consecutively from one up to eight descriptors. As a baseline, it shows the effectiveness of the exact search according to γ distance (these results were also shown in Figure 8.4 on page 99). The weights w_i from γ were computed by the α-normalization with α=0.1. Two scenarios are evaluated: when the partial NN lists are computed by exact searches, and when the partial NN lists are computed by approximate searches. In the case of exact searches, the distance aggregation performs best when combining two or three descriptors, and then it is highly affected by the fourth descriptor, which behaves as a noisy (spammer) distance. Remarkably, the merge by minimum distance is not harmed by **GH**, however its performance decreases steadily from the fifth descriptor. The rank aggregation achieves a low effectiveness when combining few descriptors, but its performance highly improves from the fifth descriptor, achieving the highest MAP overall (0.768) at the sixth descriptor.

The comparison shows that DistAgg outperforms γ distance. This result shows a strength of the late fusion: cutting the partial NN lists to the top k achieves better performance than using

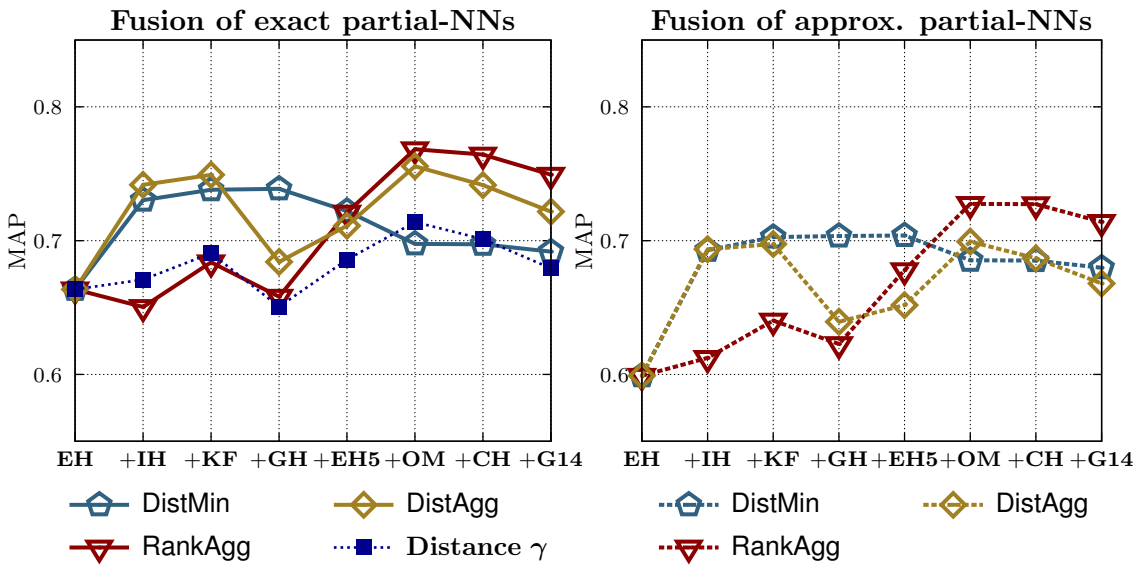


Figure 9.12 – Effectiveness of incremental combination from one up to eight descriptors.

the whole $|\mathcal{R}|$ set. A high value for k may admit too many noisy objects in the partial lists, while a too small k may drop many correct objets. In fact, the results shown in the figure were achieved with k=10, which presented the best balance.

The key difference between DistAgg and DistMin is that the former privileges objects with many repetitions in the partial NN lists, while the latter privileges the closest NN according to any d_i. Selecting objects by the minimum distance is a criterium that may give at first some robustness to noisy objects, however in the long term it fails at selecting the best objects.

The rank aggregation achieves a low performance with few descriptors. Therefore, the distance values contain relevant information for the fusion that should not be discarded when combining only a few lists. Indeed, DistAgg and DistMin outperform RankAgg when combining between two to four descriptors, where DistAgg achieves a MAP even below that using a single descriptor. However, in the long term, rank aggregation is able to achieve higher performance than distance fusions, because when distance values proceed from many search spaces, they tend to be noisy and lose their relevance.

Finally, the fusion of approximate searches was evaluated by resolving the k-NN searches using a k-means tree with branching factor 20 (which achieved high performance in Figure 9.11). The same algorithms DistAgg, DistMin, and RankAgg are evaluated but replacing the exact search by the result of approximate searches. The figure shows that the MAP decreases by less than a ten percent compared to the fusion between exact searches. On the other hand, the search times are reduced to about one fourth the time spent by an exact search indexed by a metric index. The behavior of the algorithms is fairly similar to the exact searches, achieving RankAgg the highest MAP (0.727) at six descriptors.

In summary, this experiment shows that the combination of partial results can improve both the effectiveness of the search and its efficiency. Computing partial results by each modality enables to build high performance indexes. The use of partial lists discards many irrelevant objects, increasing the effectiveness of the search. The fusion algorithm may profit from distance values in order to increase the effectiveness, in which case the α-normalization achieves appropriate results. In the particular case of combination of multidimensional vectors, the fusion may benefit from the

effectiveness-versus-efficiency tradeoff of approximate searches. In a more general case, the metric indexes can be used to increase the performance of the partial exact searches.

9.4 Summary

In this chapter we have presented two main approaches to improve the efficiency of the search.

The first approach is an approximate search using a static pivot table. The approximation relies on a fast distance estimator to discard most of the irrelevant objects without evaluating their actual distance. The approximate search shows a convenient tradeoff between effectiveness and efficiency, which enables to drastically reduce the search time while maintaining high effectiveness. We also presented a novel approach for copy detection based on local descriptor using this approximate search. Additionally, we presented the Two-step search which is an efficient approach to search in large collections using approximate searches and a complex distance.

The second approach is an exact search using a dynamic pivot table. We defined the snake distribution and we presented the Snake Table, which dynamically selects the previous query objects as pivots. Its efficiency was analyzed under different configurations and compared with LAESA and D-cache. The Snake Table reduces the search times by exploiting streams of queries with snake distributions, improving the search time under both fast and complex distances.

The two presented approaches can be combined using the Two-step search: the first step performs approximate searches with static pivots, and the second step can use the Snake Table to efficiently resolve an exact search in a reduced search space.

An interesting and open issue is the designing of an algorithm to perform approximate searches with the Snake Table. Exploiting the snake distributions to perform distance estimations could improve the effectiveness of the approximations, even in high intrinsic dimensionality spaces. However, when most of the distances are estimated and discarded, the Snake Table will not store enough distances to create good estimations.

Additionally, we included a comparison of the proposed techniques with state-of-the-art implementations of widely used multidimensional indexes. The comparison shows the metric indexes can outperform multidimensional indexes in the exact search, but the latter shows a much better effectiveness-versus-efficiency tradeoff in approximate searches. In the particular case of linear combination of distances, the many approximate searches can be combined by a late fusion algorithm in order to achieve both high effectiveness and high efficiency.

The next chapter reviews the copy localization task, which uses the output of the similarity search to locate a copy excerpt.

Chapter 10

Copy Localization

The copy localization process analyzes the lists of nearest neighbors retrieved by the similarity search, in order to look for chains of segments belonging to the same reference video. By using these chains, this process reports detections $(\bar{c}, \bar{v}, s)$, where $\bar{c}$ is an excerpt from the query video $c \in \mathcal{C}$, $\bar{v}$ is an excerpt from the reference video $v \in \mathcal{V}$, and $s \in \mathbb{R}^+$ is a confidence score (a high value implies more certainty that the reported detection is actually a copy).

in this chapter, following the procedure used in TRECVID, we model a copy detection using six values: the name of the query video, the start time and end time in the query video, the name of the reference video, the offset, and the confidence score. The offset is the constant value that needs to be added to the boundaries in the query video to get the boundaries in the reference video. The usage of the offset instead of start/end times assumes that the copy and original excerpts are the same length. This assumption is based on the fact that neither TRECVID nor MUSCLE-VCD-2007 include a transformation that accelerates or slows down a copy.

The main component for the copy localization is the voting algorithm, which produces many copy candidates between preprocessed videos. Afterwards, a gathering process combines the candidates between preprocessed videos to create the detections between query and reference videos.

10.1 Voting algorithm

Given a set of preprocessed query videos $\mathcal{C}'$ and preprocessed reference videos $\mathcal{V}'$, the input for the voting algorithm is a query video $c' \in \mathcal{C}'$ (with segments $\{s_1, ..., s_r\}$), and the list of nearest neighbors for each segment. In the algorithm, a nearest neighbor is a reference segment and a distance value. Therefore, the list of nearest neighbors for the segment s_i is $N_{s_i}=\{(r^1, dist^1), ..., (r^k, dist^k)\}$, where $\exists v' \in \mathcal{V}'\ \exists r \in v'$, $dist^j=d(s_i, r)$ for some distance function d, $dist^j \in [0, \epsilon]$, and $dist^j \leq dist^{j+1}$.

The segmentation used for the query video and for the reference video is not necessarily identical. Therefore, between a query segment q and a reference segment r, we define the offsets

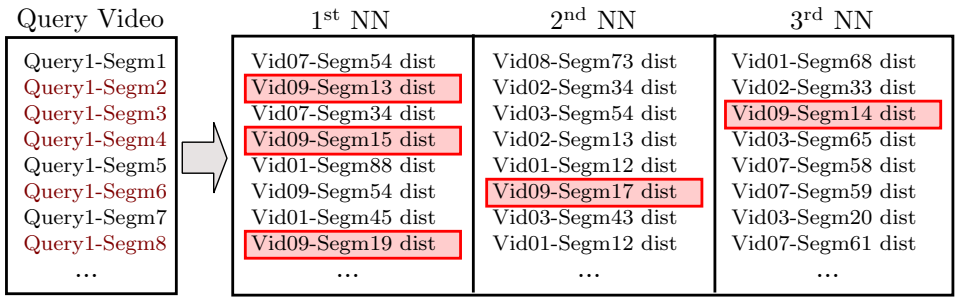


Figure 10.1 – Example showing the result of the voting algorithm. Search using three nearest neighbors, video candidate "Vid09", and offset candidate "-11". The voter segments are highlighted. The copy is reported between excerpts [Segm2, Segm8] from Query1, and [Segm13, Segm19] from Vid09.

$offset_{low}$ and $offset_{high}$ as:

$$\begin{aligned} offset_{low}(q,r) &= \min\{offset_1(q,r), offset_2(q,r)\} \\ offset_{high}(q,r) &= \max\{offset_1(q,r), offset_2(q,r)\} \\ offset_1(q,r) &= \texttt{start\_time}(q) - \texttt{start\_time}(r) \\ offset_2(q,r) &= \texttt{end\_time}(q) - \texttt{end\_time}(r) \end{aligned}$$

Given a reference video v', we define the candidate offsets between c' and v' as the range between $offset_{min}$ and $offset_{max}$ defined as:

$$\begin{aligned} offset_{min}(c',v') &= \min\{offset_{low}(q,r)\} \\ offset_{max}(c',v') &= \min\{offset_{high}(q,r)\} \\ & \quad \forall q \in c',\ \forall r \in v',\ (r, d(q,r)) \in N_q \end{aligned}$$

For each pair of videos c' and v', the range of valid offsets $[offset_{min}, offset_{max}]$ is divided into small intervals of fixed length (for instance, l=0.25 seconds) and the voting algorithm is invoked (see Algorithm 10.1). Given a reference video and the offset, the algorithm returns the query video bounds (start/end) and the copy detection score (see Figure 10.1).

The detection score s' is the sum of votes that received that copy candidate from the segments in the lists of nearest neighbors. The value of each vote and the voter segment is calculated by the function *CalculateVote*, described in Algorithm 10.2. The constant *MatchVote* is the base value for a supporting vote (for instance, *MatchVote*=1), which is weighted according to the relevance of the rank and distance of the voter segment. The constant *MissCost* is the default value when there is not a reference segment supporting the detection. This value should be zero or negative to favor detections without discontinuities (for instance, *MissCost*=−0.1).

The relevance of the rank is a weight that decreases as the rank increases. In particular, we use a geometric progression with constant ratio a, with $0 < a < 1$ (for instance, a=0.75):

$$rank\_relevance(j) = a^j \tag{10.1}$$

Analogously, the relevance of the distance is a weight that decreases as the distance increases. In particular, our implementation is based on the cumulative distribution F_d:

$$dist\_relevance(z) = 1 - F_d(z) \tag{10.2}$$

Algorithm 10.1: Voting algorithm for copy localization.

Input: $\{(s_1, N_1), ..., (s_r, N_r)\}$ set of query segment with its nearest neighbors, v' reference video candidate, ioffset offset interval candidate.

Output: start, end and score for copy detection on video v' with offset ioffset

```
(start, end, score) ← (null, null, 0)
(cstart, cend, cscore) ← (null, null, 0)
foreach (s_i, N_i) ∈ {(s_1, N_1), ..., (s_r, N_r)} do
    (voter, vote) ← CalculateVote(s_i, N_i, v', ioffset)
    cscore ← cscore + vote
    if cscore < 0 then
        (cstart, cend, cscore) ← (null, null, 0)
    else if vote > 0 then
        if cstart is null then
            cstart ← voter
        cend ← voter
        if cscore > score then
            (start, end, score) ← (cstart, cend, cscore)
return (start, end, score)
```

10.2 Evaluation of voting algorithm

All the previous evaluations in this thesis (specifically in Chapters 7, 8, and 9) follow this procedure: a configuration is fixed (i.e., input videos, segmentation, description, and distance function), the similarity search retrieves the first nearest neighbor (i.e., k-NN searches with k=1 and range search ϵ=∞), and the voting algorithm generates the detections and their confidence scores (using parameters l=0.25, *MatchVote*=1, *MissCost*=−0.1). The evaluation method uses the ground-truth to determine the amount of actual copies detected by correct detections with score higher than any incorrect detection. Because the searches used k=1, there is no influence of weighting functions *rank_relevance* (Equation 10.1) and *dist_relevance* (Equation 10.2) in the final detection result.

The following experiments evaluate the impact on detection effectiveness of parameter k, *rank_relevance*, and *dist_relevance*. The experiments use preprocessed videos, **S1** segmentation, s-t global description and acoustic description, and L_1 distance, thus the detection effectiveness for k=1 is already shown in Figure 7.3 on page 82 in the **S1** column.

Table 10.1 shows the variations in the number of detected copies without false alarms with k between 1 and 10 and both *rank_relevance* and *dist_relevance* are disabled (i.e., they both return the constant 1). The detection effectiveness may increase or decrease depending on k and the descriptor. The major improvement occurs with k=3, when the effectiveness improves for eight descriptors and it decreases for one of the fifteen tested descriptors. On the other hand, with k=10 the effectiveness improves for four descriptors and it decreases for six configurations.

Table 10.2 shows the variations in the detection effectiveness with k between 1 and 10, and *rank_relevance* is used with parameter a=0.75. The table highlights the impact of *rank_relevance*: differences in boldface mark an improvement with respect to the previous experiment, and differences in italics mark a decrease. Globally, eight descriptors are benefited by the use of *rank_relevance*

Algorithm 10.2: *CalculateVote* function for copy localization.

Input: s query segment, $\{(r^1, dist^1), ..., (r^k, dist^k)\}$ list of the k nearest neighbors to s, v' reference video candidate, ioffset offset interval candidate.
Output: voter best matching segment, vote score of match.

```
(voter, vote) ← (null, MissCost)
foreach (r^j, dist^j) ∈ {(r^1, dist^1), ..., (r^k, dist^k)} do
    if r^j ∈ v' and offset(s,r^j) ∈ ioffset then
        v ← MatchVote
                × rank_relevance(j)
                × dist_relevance(dist^j)
        if v > vote then
            (voter, vote) ← (r^j, v)
return (voter, vote)
```

while two show some decrease in effectiveness. The major improvement occurs again with k=3, when the effectiveness improves for eight descriptors without showing decreases.

Table 10.3 shows the variations in the detection effectiveness with k between 1 and 10 including both *rank_relevance* and *dist_relevance*. The table highlights the impact of *dist_relevance*: differences in boldface mark an improvement with respect to the previous experiment, and differences in italics mark a decrease. Globally, the table shows a little impact of *dist_relevance*. The major improvement affects $\mathbf{OM}^t_{9x9}$ which can detect one more copy from $k \geq 6$.

The experiments show that performing a search with $k > 1$ can increase the effectiveness but it also increases the number of false alarms. In this case, the inclusion of *rank_relevance* reduces the increase of false alarms, producing a satisfactory balance with k=3. On the other hand, the effect of *dist_relevance* is minimal, i.e., it shows neither an improvement or decrease for most of the descriptors. We also tested the inclusion of *dist_relevance* and disabling *rank_relevance*, which also shows almost no impact on detection effectiveness for *dist_relevance*.

10.3 Combination of candidates

The voting algorithm returns a list of candidate detections between preprocessed query videos $\mathcal{C}'$ and preprocessed reference videos $\mathcal{V}'$. The final step consists in processing the list of candidates $(\bar{c}', \bar{v}', s')$ in order to produce the final list of copy detections $(\bar{c}, \bar{v}, s)$.

Let $P(c) \subset \mathcal{C}'$ be the videos created by the preprocessing task for the query video $c \in \mathcal{C}$. If the preprocessing considers some reversions (like the reversion of PIP described in Chapter 6), then $|P(c)| \geq 1$, but if the preprocessing step only considers a quality normalization then $|P(c)| = 1$. All the candidates are gathered, and the candidates referring to the same query and reference videos are joined, i.e., $\bar{c} = \bigcup \bar{c}'$, $\bar{v} = \bigcup \bar{v}'$, and $s = \sum s'$. Optionally, the final scores may be scaled to the interval [0,1]. The list of detections $(\bar{c}, \bar{v}, s)$ corresponds to the final answer of the system.

Descriptors	Dets. no f.a. k=1	Difference to k=1								
		2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
EH^{t}4x4-10	28	0	+1	0	0	0	0	0	0	0
EH^{t}4x4-5	27	+1	+1	+1	+1	+1	+1	+1	+1	+1
KF^{t}11x9	23	+1	0	0	0	0	0	0	0	0
OM^{t}9x9	15	+1	+1	0	+2	+2	+1	+1	+1	0
CH^{t}1x1-hsv-16x4x4	20	−1	−1	−1	−1	−1	−1	−1	−1	−1
CH^{t}2x2-hsv-16x2x2	21	0	0	+1	0	+1	+1	+1	0	+1
CH^{t}2x2-rgb-4x4x4	21	+1	+1	+1	+1	0	+1	0	0	−1
GH^{t}1x1-180	16	+1	+1	+1	0	0	−1	0	0	0
GH^{t}1x4-32	19	0	0	0	0	−1	−1	−1	−1	−1
GH^{t}3x3-20	21	−1	0	0	0	−1	0	−1	−1	−1
GH^{t}4x4-12	21	0	+1	+1	+1	0	0	0	+1	+1
IH^{t}1x4-rgb-3x16	20	−1	0	+1	0	0	0	0	0	0
IH^{t}2x2-rgb-3x16	20	0	+1	+1	+1	+1	−1	−1	−1	−1
IH^{t}4x4-rgb-3x4	21	0	+1	+1	+1	+1	+1	+1	+1	+1
AU160	29	0	0	−1	−1	−1	0	0	0	−1

Table 10.1 – Detections without false alarms when increasing k and both $rank\_relevance$ and $dist\_relevance$ are disabled.

Descriptors	Dets. no f.a. k=1	Difference to k=1								
		2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
EH^{t}4x4-10	28	0	*0*	0	0	0	0	0	0	0
EH^{t}4x4-5	27	+1	+1	+1	+1	+1	+1	+1	+1	+1
KF^{t}11x9	23	+1	0	0	0	0	0	0	0	0
OM^{t}9x9	15	+1	+1	**+1**	+2	+2	**+2**	**+2**	**+2**	**+2**
CH^{t}1x1-hsv-16x4x4	20	**0**	**0**	−1	−1	−1	−1	−1	−1	−1
CH^{t}2x2-hsv-16x2x2	21	0	0	*0*	0	*0*	*0*	*0*	0	*0*
CH^{t}2x2-rgb-4x4x4	21	+1	+1	+1	+1	**+1**	+1	**+1**	**+1**	**+1**
GH^{t}1x1-180	16	+1	+1	+1	**+1**	**+1**	**+1**	**+1**	**+1**	**+1**
GH^{t}1x4-32	19	0	0	0	0	**0**	**0**	**0**	**0**	**0**
GH^{t}3x3-20	21	−1	0	0	0	−1	0	−1	−1	−1
GH^{t}4x4-12	21	0	+1	+1	+1	**+1**	**+1**	**+1**	+1	+1
IH^{t}1x4-rgb-3x16	20	**0**	**+2**	**+2**	**+2**	**+2**	**+2**	**+2**	**+2**	**+2**
IH^{t}2x2-rgb-3x16	20	0	+1	+1	+1	+1	**+1**	**+1**	**+1**	**+1**
IH^{t}4x4-rgb-3x4	21	0	+1	+1	+1	+1	+1	+1	+1	+1
AU160	29	0	0	**0**	**0**	**0**	0	0	0	**0**

Table 10.2 – Detections without false alarms when increasing k, $rank\_relevance$ is active and $dist\_relevance$ is disabled. Differences in boldface mark an improvement with respect to Table 10.1, and differences in italics mark a decrease.

Descriptors	Dets. no f.a. k=1	Difference to k=1								
		2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
EH^{t}4x4-10	28	0	0	0	**+1**	0	0	0	0	0
EH^{t}4x4-5	27	+1	+1	+1	+1	+1	+1	+1	+1	+1
KF^{t}11x9	23	+1	0	0	0	0	0	0	0	0
OM^{t}9x9	15	+1	+1	+1	+2	**+3**	**+3**	**+3**	**+3**	**+3**
CH^{t}1x1-hsv-16x4x4	20	0	0	−1	−1	−1	−1	−1	−1	−1
CH^{t}2x2-hsv-16x2x2	21	0	0	0	0	0	0	0	0	0
CH^{t}2x2-rgb-4x4x4	21	+1	+1	+1	+1	*0*	+1	+1	+1	+1
GH^{t}1x1-180	16	+1	+1	+1	+1	+1	+1	+1	+1	+1
GH^{t}1x4-32	19	0	0	0	0	0	0	0	0	0
GH^{t}3x3-20	21	−1	0	0	0	−1	0	−1	−1	−1
GH^{t}4x4-12	21	0	+1	+1	+1	+1	+1	+1	+1	+1
IH^{t}1x4-rgb-3x16	20	0	+2	+2	+2	+2	+2	+2	+2	+2
IH^{t}2x2-rgb-3x16	20	0	+1	+1	+1	+1	+1	+1	+1	+1
IH^{t}4x4-rgb-3x4	21	0	+1	+1	+1	+1	+1	+1	+1	+1
AU160	29	0	0	0	0	0	0	0	0	0

Table 10.3 – Detections without false alarms when increasing k, both *rank_relevance* and *dist_relevance* are active. Differences in boldface mark an improvement with respect to Table 10.2, and differences in italics mark a decrease.

10.4 Summary

In this chapter we have presented our approach to locate copies from the lists of nearest neighbors retrieved by the similarity search. The voting algorithm locates chains of nearest neighbors belonging to the same reference video. The combination process gathers all the candidate between preprocessed videos to produce the final detection list.

The evaluation showed that the voting algorithm can improve its effectiveness when it considers more nearest neighbors ($k > 1$), however a weighting of the voter by its rank position is needed in order to prevent an increase in the false alarms. We also evaluated the weighting of the voter according to its distance to the query object, but it showed almost no impact on system effectiveness.

The next chapter reviews our participation at TRECVID 2010 and 2011, where we compared the performance of our system using most of the developed techniques, with other state-of-the-art systems.

Chapter 11

Evaluation at TRECVID

During the development of this thesis, we participated at TRECVID's CBVCD evaluation. In that evaluation, the performance of our system was compared with other state-of-the-art CBVCD systems.

In this chapter we review our participation at TRECVID 2010 and TRECVID 2011, and we compare our results with the other participants. Additionally, Appendix B and Appendix C present detailed data with the evaluation of all the participating teams.

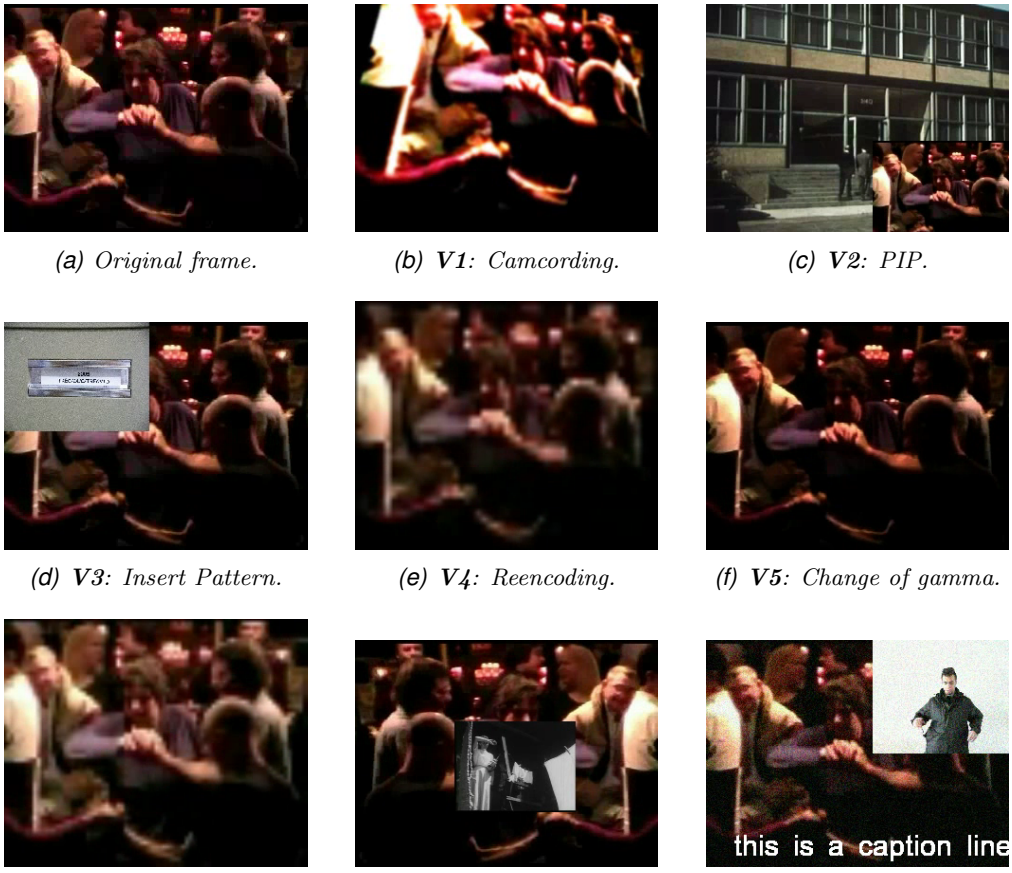
11.1 Introduction

The Text Retrieval Conference (TREC), organized yearly by the National Institute of Standards and Technology (NIST), focuses on the research in information retrieval. As a part of TREC since 2003, the TREC Video Retrieval Evaluation (TRECVID) promotes research in video information retrieval by providing large test collections, uniform scoring procedures, and a forum for organizations interested in comparing their results [Smeaton et al., 2006]. Between 2008 and 2011, TRECVID included an evaluation for CBVCD systems, called Content-Based Copy Detection (CCD). Many teams from different universities and private companies around the world participated in the CCD evaluation. This evaluation served as a benchmark to test and compare different CBVCD techniques. During 2008 and 2009 the evaluation used datasets from the BBC collection, while for 2010 and 2011 it used a more general dataset with videos from internet, called IACC.1.

11.2 TRECVID datasets

TRECVID 2010 and 2011 used the datasets IACC.1.A and IACC.1.tv10.training as the reference collection. It contains more than eleven thousand internet short video clips with duration between 11 seconds and 4.1 minutes. The visual and audio quality of reference videos is variable: the videos proceed from TV rips, handheld and cellphone cameras, movie excerpts, slideshows, etc. The videos have different resolutions (between 320×240 and 1104×240) and different frame rates (from less than 1 fps up to more than 75 fps).

The query collections for TRECVID 2010 and 2011 were created following the same process.



(a) Original frame. *(b) **V1**: Camcording.* *(c) **V2**: PIP.*

*(d) **V3**: Insert Pattern.* *(e) **V4**: Reencoding.* *(f) **V5**: Change of gamma.*

*(g) **V6**: gamma+ frame drop+reencoding.* *(h) **V8**: mirroring+PIP+ contrast.* *(i) **V10**: PIP+noise+ caption.*

Figure 11.1 – Example of visual transformations in TRECVID datasets. (a) Frame from a reference video. (b-i) Same frame for the eight visual transformations.

Code	Description
V1	Simulated camcording.
V2	Picture-in-picture (PIP) original video in foreground.
V3	Insertion of pattern.
V4	Strong reencoding.
V5	Change of gamma.
V6	Combination of three quality transformations: blur, change of gamma, frame dropping, contrast, reencoding, ratio, and white noise.
V8	Combination of three postproduction transformations: crop, shift, contrast, caption, mirroring, insertion of pattern, and PIP original video in background.
V10	Random combination of three previous transformations.

(a) The eight visual transformations evaluated at TRECVID.

Code	Description
A1	No transformation.
A2	Mp3 compression.
A3	Mp3 compression and multiband companding.
A4	Bandwidth limit and single-band companding.
A5	Mix with speech.
A6	Mix with speech and multiband compress.
A7	Bandpass filter mix with speech and compress.

(b) The seven acoustic transformations evaluated at TRECVID.

Table 11.1 – The evaluated transformations at TRECVID 2010 and 2011.

First, 201 base videos are created, each one with a length between 3 seconds and 3 minutes: 67 videos were an excerpt from a reference video, 67 were an excerpt from a video not in the reference collection, and 67 contain an excerpt from a reference video embedded into a longer excerpt from a video not in the reference collection. The audio query collection was created by applying seven acoustic transformations (with codes **A1** to **A7**) to each of the 201 base videos. The visual query collection was created by applying eight visual transformations (with codes **V1** to **V10**) to each of the 201 base videos. Transformations **V7** (combination of five quality transformations) and **V9** (combination of five postproduction transformations) were only evaluated in 2008 and then they were discarded for the subsequent years. The final a+v query collection was created by combining the audio queries and the visual queries for the same base video, producing 56 query videos from each base video. Table 11.1 details the eight visual transformations and the seven acoustic transformations. Figure 11.1 shows an example of the eight visual transformations applied to a base video. Table 11.2 summarizes the sizes of the collections.

11.3 Evaluation process

Two configuration profiles were tested: Balanced and No False Alarms (NoFA). Teams can submit up to four Runs containing the detection results produced by their CBVCD system according to one of these profiles. A Run is a text file with a list detections, each detection reports the boundaries of the copy excerpt for the query video, the offset of the boundaries for some reference video, and a confidence score. Additionally, the Run contains a decision threshold submitted by the team, and the total time in seconds required to process each query video (including the decoding and feature extraction, read/write of intermediate results, and generation of the final output), i.e., the time required for the online phase.

Collection	Videos	Hours	Frames	GB
Reference	11,524	425.4	39,463,431	100.3
Query				
Audio	1,407	27.7	—	0.8
Visual	1,608	32.0	3,402,620	25.3
Audio+Visual	11,256	222.8	23,818,340	—

(a) TRECVID 2010

Collection	Videos	Hours	Frames	GB
Reference	11,485	423.8	39,280,446	99.9
Query				
Audio	1,407	28.3	—	0.8
Visual	1,608	32.3	3,359,710	24.3
Audio+Visual	11,256	226.3	23,517,970	—

(b) TRECVID 2011

Table 11.2 – Summary of TRECVID 2010 and 2011 collections.

The evaluation of each Run relies on three measures (two for effectiveness and one for efficiency). NIST calculates these measures separately for each transformation, it separates the detections in the Run into independent files by transformation, and for each of the 56 transformations the following three measures are calculated:

- **Normalized Detection Cost** (NDCR): Measures the effectiveness of the detection. It considers the probability of missing a detection and the probability to falsely indicate that there is a copy for a query video:

  $$\text{NDCR} = P_{\text{MISS}} + \beta \cdot R_{\text{FA}}$$

  where P_{MISS} is the conditional probability of a missed copy, R_{FA} is the conditional probability of a false alarm rate, and β is a weight parameter. NDCR sets β=200 for Balanced profile, and β=200,000 for NoFA profile. NDCR ranges between 0 and $+\infty$, where being closer to zero means better detection effectiveness. A trivial NDCR of 1.0 can be obtained by submitting an empty Run, thus a good result should not be greater than this value.

- **F1**: Measures the accuracy in localization after a copy has been correctly detected. It is the harmonic mean of the precision and recall calculated over the lengths of the detected and copied excerpts:

  $$\text{F1} = \frac{2}{\frac{1}{\text{presicion}} + \frac{1}{\text{recall}}}$$

  F1 ranges between 0 and 1, closer to 1.0 means better accuracy at detecting the copy boundaries.

- **Mean query processing time** (MPT): Measures the efficiency at processing the query videos. It is calculated by averaging the reported time for queries belonging to each transformation.

NDCR and F1 are calculated using two decision thresholds for the confidence score: the submitted threshold by the team (it gives the actual performance of the system); and the optimal

threshold, which is the threshold that produces the minimal NDCR for each transformation (it gives the optimal performance of the system). Thus, the evaluation of a Run produces five measures (Actual NDCR, Actual F1, Optimal NDCR, Optimal F1, and MPT) for each of the 56 transformations. The optimal threshold is calculated for each transformation while the submitted threshold is fixed for every transformation. Unfortunately, the evaluation process did not consider an optimal threshold fixed for all transformations. In order to globally compare system performances, we averaged the 56 measures for each transformation into one average value, producing five measures for each Run: Average Actual NDCR, Average Actual F1, Average Optimal NDCR, Average Optimal F1, and Average MPT.

TRECVID 2010 and 2011 allowed CBVCD systems to rely on both acoustic with visual information, although some systems may use only one source and discard the other. For comparison purposes we classified every Run into audio-only (A), visual-only (V), and audio+video (AV). We state that a Run is type V when its results for NDCR and F1 are identical for the seven acoustic transformations in a same visual transformation (thus, its results are not influenced by changes in audio). Analogously, we state that a Run is type A when NDCR and F1 are identical for the eight visual transformations in a same acoustic transformation. Finally, a Run is type AV if it is neither type A nor type V.

11.4 Participation at TRECVID 2010

This section reviews our first participation in the CCD evaluation. In this participation we tested a system based exclusively on visual global descriptors, thus we discarded the audio information generating exclusively Runs type V.

11.4.1 Submissions

The preprocessing task created new query videos for PIP, camcording and vertical flip transformation. As a result of this task, the number of query videos used by next tasks increased from 1,607 to 5,378 (containing 11,526,076 frames).

The video segmentation task used a variable-length partitioning: first it used a base segmentation $\mathbf{S}^1\!/_3$, then the descriptor **KF** with parameters 20×15 was extracted for representative frames, and finally two consecutive segments were joined if the L_1 distance for their descriptor was smaller that threshold 24. The 5,378 preprocessed query videos produced 990,246 query segments, and the 11,524 reference videos produced 3,967,815 reference segments.

The feature extraction task described each segment by three global descriptors: $\mathbf{EH}^t{}_{\text{4x4-10}}$, $\mathbf{GH}^t{}_{\text{3x3-20}}$, and $\mathbf{IH}^t{}_{\text{2x2-rgb-3x16}}$. Table 11.3 shows the sizes of the extracted descriptors.

Descriptor	Length	Query size	Reference size
$\mathbf{EH}^t{}_{\text{4x4-10}}$	160 bytes	151 MB	605 MB
$\mathbf{GH}^t{}_{\text{3x3-20}}$	180 bytes	170 MB	681 MB
$\mathbf{IH}^t{}_{\text{2x2-rgb-3x16}}$	192 bytes	181 MB	727 MB

Table 11.3 – Descriptors used at TRECVID 2010.

In order to create our submissions we defined two configurations: **ehdNgry** and **ehdNclr**, summarized in Table 11.4. These configurations used a spatio-temporal distance δ with temporal window W=3, α-normalization (α=0.0001), and weighting by max-ρ.

Name	Segmentation	Distance	Descriptor
ehdNgry	$\mathbf{S}^{\mathbf{1}}/_{\mathbf{3}}$+comb.	δ	L_1-$\mathbf{EH}^t$4x4-10 L_1-$\mathbf{GH}^t$3x3-20
ehdNclr	$\mathbf{S}^{\mathbf{1}}/_{\mathbf{3}}$+comb.	δ	L_1-$\mathbf{EH}^t$4x4-10 L_1-$\mathbf{IH}^t$2x2-rgb-3x16

Table 11.4 – Configurations used at TRECVID 2010.

For each configuration a k-NN+range searches were performed with k=6 and ϵ=6. The exact search with pivots would have taken nearly eleven months to complete. In order to reduce the search time we use the approximate search with pivots using parameters $|\mathcal{P}|$=9 and T=0.001·$|\mathcal{R}|$. We fixed that parameters by first deciding the amount of time that similarity search should take (we decided the search should not take more than 24 hours total for all queries), and then we tested different values for T and $|\mathcal{P}|$ to fulfill that performance. In both configurations the evaluation of δ needed more than 1,000 operations, but the approximate search estimated it as using $LB_{\mathcal{P}}$ (which requires only 9 operations) and evaluated δ only 0.1% times (3,967 evaluations for each query segment).

We submitted two Runs for the Balanced profile: `balanced.ehdNgryhst` (configuration **ehdNgry**) and `balanced.ehdNclrhst` (configuration **ehdNclr**); and two Runs for the No False Alarms profile: `nofa.ehdNgryhst` (configuration **ehdNgry**) and `nofa.ehdNghT10` (same as previous but with a stricter decision threshold).

We performed all the processes on a single desktop computer with Intel Q9400 CPU (2.66 GHz × 4 cores) and 4 GB RAM on a GNU/Linux 2.6.18. The whole system was implemented in C using OpenCV and FFmpeg libraries.

11.4.2 Results

Twenty-two teams participated in the CCD evaluation at TRECVID 2010. Each team submitted four Runs, which resulted in 37 submissions for the NoFA profile (14 Runs type V), and 41 submissions for the Balanced profile (15 Runs type V). The list of participants, the results for each participant, and the results by transformation are detailed in Appendix B.

Average Results

Table 11.5 shows the average results for all the transformations for the submitted Runs to NoFA profile and Balanced profile. Figure 11.2 depicts a comparison with the other Runs for Average Optimal NDCR, Average Optimal F1 and Average MPT.

In the NoFA profile, both submissions `nofa.ehdNghT10` and `nofa.ehdNgryhst` achieved the same Optimal NDCR and Optimal F1 because they only differed in the decision threshold. Both optimal NDCR and F1 are better than the median, and considering just Runs type V they achieved the best results. In the case of Actual NDCR, the decision thresholds were too permissive (both NDCR are higher than 1.0 because the threshold accepts too many false alarms, but still both

Indicator	Value	Overall Rank	Rank in Runs V
`nofa.ehdNghT10`			
Average Optimal NDCR	0.611	10 of 37	1 of 14
Average Optimal F1	0.828	14 of 37	1 of 14
Average Actual NDCR	40.75	9 of 37	2 of 14
Average Actual F1	0.846	15 of 37	1 of 14
Average MPT	128 s.	23 of 37	11 of 14
`nofa.ehdNgryhst`			
Average Optimal NDCR	0.611	10 of 37	1 of 14
Average Optimal F1	0.828	14 of 37	1 of 14
Average Actual NDCR	147.7	13 of 37	4 of 14
Average Actual F1	0.811	16 of 37	2 of 14
Average MPT	128 s.	23 of 37	11 of 14
`balanced.ehdNgryhst`			
Average Optimal NDCR	0.597	14 of 41	1 of 15
Average Optimal F1	0.820	16 of 41	3 of 15
Average Actual NDCR	9.057	28 of 41	7 of 15
Average Actual F1	0.723	19 of 41	11 of 15
Average MPT	128 s.	26 of 41	11 of 15
`balanced.ehdNclrhst`			
Average Optimal NDCR	0.658	16 of 41	3 of 15
Average Optimal F1	0.820	15 of 41	2 of 15
Average Actual NDCR	8.902	27 of 41	6 of 15
Average Actual F1	0.724	20 of 41	12 of 15
Average MPT	132 s.	27 of 41	12 of 15

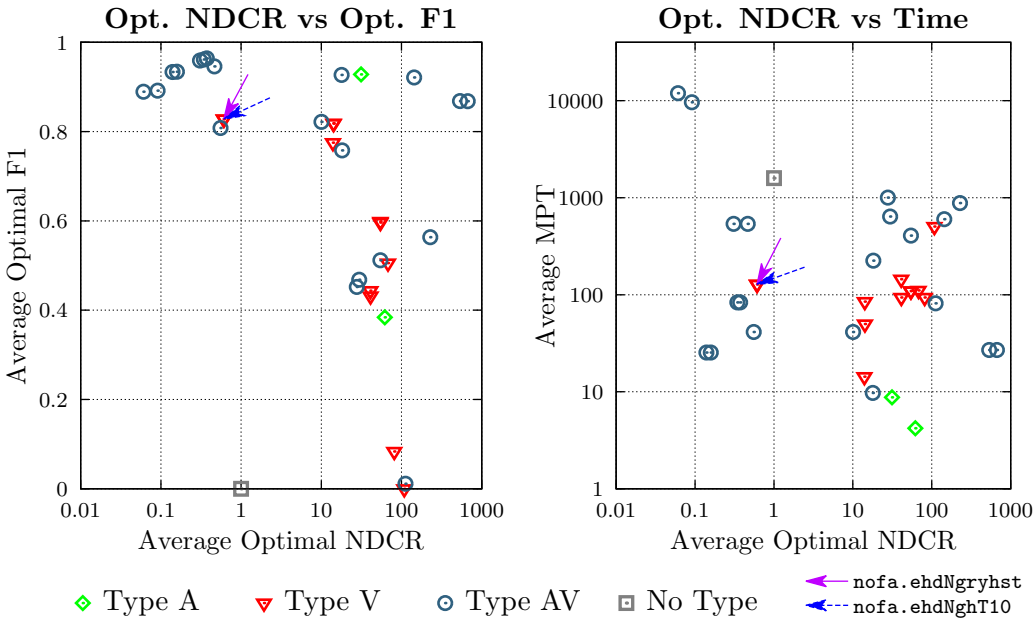
Table 11.5 – Evaluation for submitted Runs to TRECVID 2010 (average values for the 56 transformations).

NDCR are better than the median). Fixing an appropriate threshold was a difficult task for all the teams, in fact only five submissions achieved an Average Actual NDCR lower than 1.0.

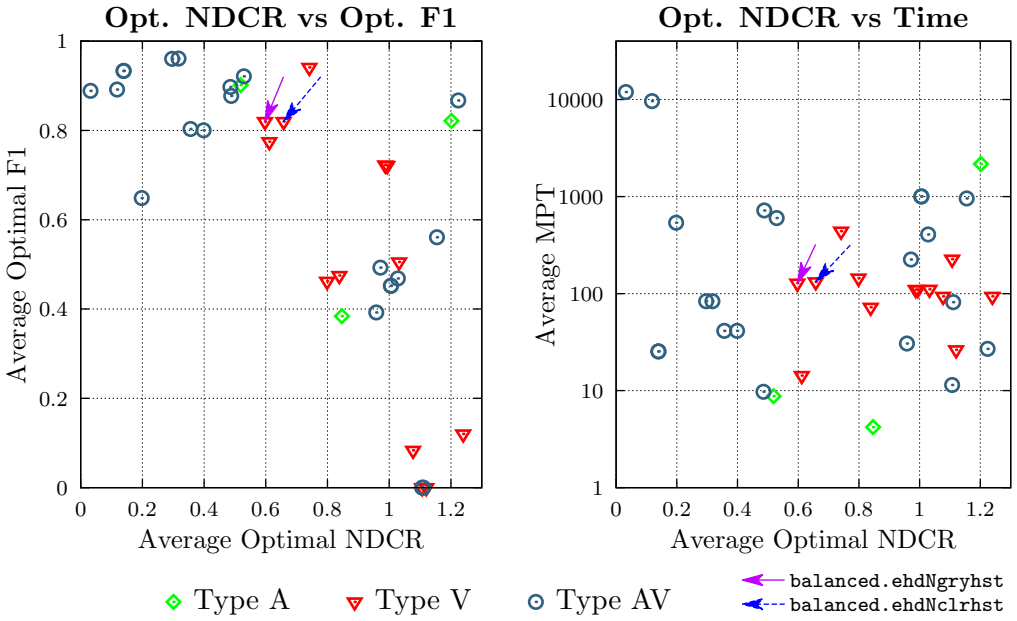
In the Balanced profile, `balanced.ehdNgryhst` and `balanced.ehdNclrhst` achieved an Average Optimal NDCR better than the median, and `balanced.ehdNgryhst` achieved the best optimal NDCR between Runs type V. As in NoFA profile, the high Actual NDCR is due to a too permissive decision threshold.

Comparing Optimal NDCR, `nofa.ehdNgryhst` was the 10^{th} between 37 submissions, while `balanced.ehdNgryhst` was the 14^{th} between 41 submissions. These results show that the system achieves a better performance for the NoFA profile than for the Balanced profile.

The four submissions achieved a Mean Processing Time higher than the median, thus the system was relatively slow. This is mainly due to the preprocessing task, which was inefficiently implemented and took almost half of the total processing time. In particular, queries with PIP and camcording required about three and four times, respectively, more time to be processed. Furthermore, the restriction of taking no more than 24 hours in the search could have been reduced in order to increase the efficiency of the system.

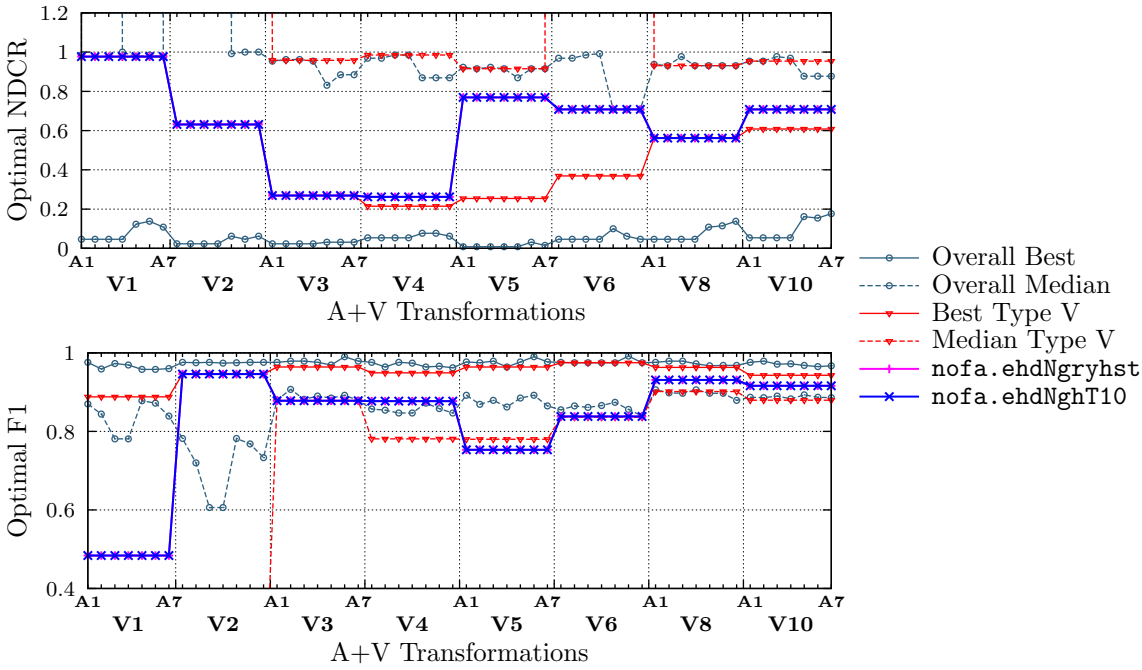


(a) Runs for NoFA profile. The data for this figure is shown in Table B.2 on page 190.

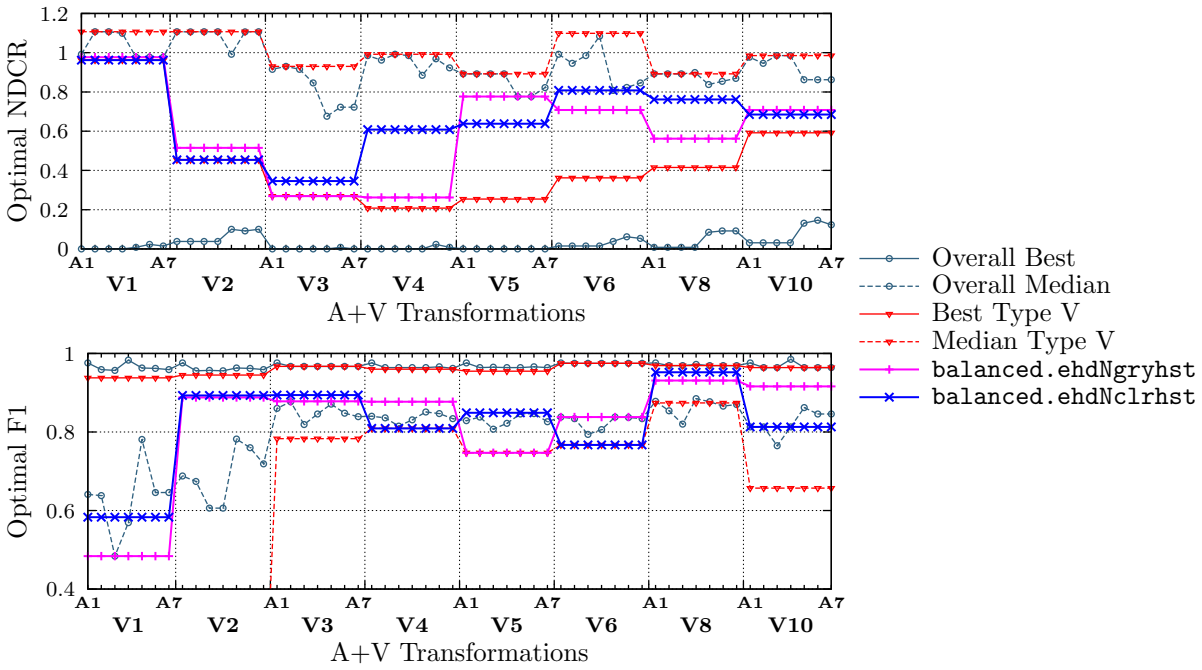


(b) Runs for Balanced profile. The data for this figure is shown in Table B.4 on page 192.

Figure 11.2 – Average Optimal NDCR, Average Optimal F1 and Average MPT for Runs at TRECVID 2010. The ideal indicators are NDCR=0, F1=1, and MPT=0.



(a) NoFA profile. Data shown in Table B.3 on page 191.



(b) Balanced profile. Data shown in Table B.5 on page 193.

Figure 11.3 – Optimal NDCR and Optimal F1 by transformation at TRECVID 2010.

Results by Transformation

In order to analyze the performance by transformation, we compare the results of Optimal NDCR and Optimal F1 for all submissions at NoFA profile and Balanced profile. Figure 11.3 shows the results for each transformation for our submissions compared with: the best result, the median, the best result for Runs type V, and the median for Runs type V.

In the NoFA profile, `nofa.ehdNghT10` and `nofa.ehdNgryhst` achieved their best effectiveness at **V4** and **V3** (NDCR 0.262 and 0.269, respectively). Their effectiveness were always better than the overall median, and between Runs type V they achieved the best NDCR for **V1**, **V2**, **V3** and **V8**. Globally, the most difficult transformation was **V2**, where only three submissions type V achieved a NDCR less than 1.0, and the second most difficult was **V1**, where only five submissions type V achieved NDCR less than 1.0. This shows the relevance of the preprocessing, where the inclusion of detection and reversion enabled a remarkable performance. In copy localization (F1), the best localization was achieved for **V2** and the worst localization for **V1**.

In the Balanced profile, `balanced.ehdNgryhst` and `balanced.ehdNclrhst` achieved better effectiveness than the overall median and the median for Runs Type V for every transformation. `balanced.ehdNclrhst` achieved the best effectiveness for Runs type V at **V1** and **V2** and `balanced.ehdNgryhst` achieved the best effectiveness for Runs type V at **V3**. The localization was accurate for **V8** but it was not accurate for **V1**.

In summary, the results for our submitted Runs were positioned above the overall median for Optimal NDCR at every transformation. Considering just Runs type V, they achieved the best detection performance for **V1**, **V2** and **V3**. The localization was not very accurate, since some transformations were difficult to locate (in particular **V1** and **V5**).

11.5 Participation at TRECVID 2011

This section reviews our second participation in the CCD evaluation. We tested a combination of visual and acoustic information at the similarity search level, i.e., the system compared segments using both acoustic descriptors and global descriptors in the distance function. This is a novel approach because most of the CBVCD systems fuse candidates from independent subsystems, each subsystem using either visual or audio information.

11.5.1 Submissions

The preprocessing task creates new query videos for PIP, camcording and vertical flip transformation. As a result of this task, the number of visual query videos increased from 1,608 to 5,147, thus the number of a+v queries also increased from 11,256 to 36,029.

The video segmentation used a segmentation $\mathbf{S}\frac{\mathbf{1}}{\mathbf{3}}$. Unlike our previous participation, we chose a fixed-length segmentation in order to simplify the fusion of acoustic with visual descriptors. The 11,485 reference videos produced $|\mathcal{R}_v|$=4,522,262 visual segments, and $|\mathcal{R}_a|$=4,441,717 audio segments (some videos have different lengths for audio and visual tracks). The 5,147 visual queries produced $|\mathcal{Q}_v|$=1,120,455 visual segments, and the 1,407 audio queries produced $|\mathcal{Q}_a|$=306,304 audio segments.

The feature extraction task described each visual segment in $\mathcal{Q}_v \cup \mathcal{R}_v$ by three global descriptors (**EH**^{t}4x4-10, **GH**^{t}4x4-12, and **IH**^{t}4x4-rgb-3x4), and each audio segment in $\mathcal{Q}_a \cup \mathcal{R}_a$ by one acoustic descriptor (**AU**160). Table 11.6 shows the sizes of the extracted descriptors. Unlike previous participation, we prefer to use zoning of 4×4 grid for every descriptor because changing zoning highly impacts the effectiveness of the description (see Chapter 7).

Descriptor	Length	Query size	Reference size
EH^{t}4x4-10	160 bytes	212 MB	691 MB
GH^{t}4x4-12	192 bytes	206 MB	829 MB
IH^{t}4x4-rgb-3x4	192 bytes	206 MB	829 MB
AU160	640 bytes	187 MB	2,7 GB

Table 11.6 – Descriptors used at TRECVID 2011.

Let $\mathcal{Q}_{av}$ be the set of a+v-segments for query videos, and $\mathcal{R}_{av}$ be the set of a+v-segments for reference videos. We created $\mathcal{Q}_{av}$ by combining sets $\mathcal{Q}_a$ and $\mathcal{Q}_v$ and their descriptors following the script `tv11.make.av.queries.sh`, which produced $|\mathcal{Q}_{av}|$=7,840,587 a+v-segments. We created $\mathcal{R}_{av}$ by combining sets $\mathcal{R}_a$ and $\mathcal{R}_v$ and their descriptors, producing $|\mathcal{R}_{av}|$=4,387,633 a+v-segments. The combination process requires that visual-segments and audio-segments have the same length to create an a+v-segment, and it guarantees that every created a+v-segment has all the visual and acoustic descriptors, i.e., an a+v-segment is discarded if it lacks of any acoustic or visual descriptor.

The submissions were based on four configurations: **EhdGry**, **EhdRgb**, **Aud** and **EhdRgbAud**, summarized in Table 11.7. The γ distance for configurations **EhdGry**, **EhdRgb**, and **EhdRgbAud** used α-normalization (α=0.001) and weighting by max-τ. Unlike previous participation, we chose a temporal window W=1 because the improvement of effectiveness due to a large W was not worth the increase in the search time (see Chapter 8).

In **EhdRgbAud** we had to manually decrease w_{Aud} because we already knew from TRECVID 2010 that audio tracks from the query videos were not as reliable as the visual tracks to detect copies. The guidelines for the CCD evaluation ensures a copy exists in both visual and audio tracks at the same time, however there are some valid copies whose audio track is ruined by the acoustic transformation. Moreover, after analyzing the ground-truth, we realized that an acoustic transformation replaces (maybe unintentionally) the audio track from a valid copy with an audio track from an unrelated video. This creates some a+v queries where the visual track matches the original while the audio track does not match (i.e., the copy exists only in the visual track). In particular, this behavior is observable in query videos `8960.mpg`, `6008.mpg`, `7468.mpg`, `7996.mpg`, `9261.mpg`, `9472.mpg`, `9517.mpg`, and maybe others. This issue was reported to NIST but they ruled that those queries were valid.

The submitted Runs `nofa.EhdGry` and `balanced.EhdGry` were similar to the previous participation. The similarity search used configuration **EhdGry** to perform a k-NN approximate search in $\mathcal{R}_v$ for every query segment in $\mathcal{Q}_v$. The approximate search fixed k=10 and approximation parameters T=1% and $|\mathcal{P}|$=5 pivots. Hence, each estimation cost five operations, and for the $0.01 \cdot |\mathcal{R}_v|$=45,222 objects with lowest estimations the actual γ was calculated. Compared with previous participation, we increased parameter T and decreased parameter $|\mathcal{P}|$, because T is more relevant than $|\mathcal{P}|$ for the effectiveness of the approximation (see Chapter 9). The copy localization algorithm located chains of nearest neighbors belonging to the same reference video and offset. The chain with the highest score for each query video was reported in `nofa.EhdGry`, and the two chains with highest scores were reported in `balanced.EhdGry`.

Name	Segmentation	Distance	Descriptor
EhdGry	$\mathbf{S}^{1}/_{3}$	γ	L_1-$\mathbf{EH}^{t}$4x4-10 L_1-$\mathbf{GH}^{t}$4x4-12
EhdRgb	$\mathbf{S}^{1}/_{3}$	γ	L_1-$\mathbf{EH}^{t}$4x4-10 L_1-$\mathbf{IH}^{t}$4x4-rgb-3x4
Aud	$\mathbf{S}^{1}/_{3}$	L_1	**AU**160
EhdRgbAud	$\mathbf{S}^{1}/_{3}$	γ	L_1-$\mathbf{EH}^{t}$4x4-10 L_1-$\mathbf{IH}^{t}$4x4-rgb-3x4 L_1-Aud

Table 11.7 – Configurations used at TRECVID 2011.

The submitted Runs `nofa.EhdRgbAud` and `balanced.EhdRgbAud` performed a similarity search using configuration **EhdRgbAud**. However, we did not directly apply the approximate search with pivots (as in our previous participation) due to two major drawbacks:

- The similarity search must perform $|\mathcal{Q}_{av}|$ approximate searches in $\mathcal{R}_{av}$, which compared against **EhdGry** ($|\mathcal{Q}_{v}|$ searches in $\mathcal{R}_{v}$) is an increase of almost seven times. Therefore, the approximate parameters T and $\mathcal{P}$ should be adjusted to reduce the search time to one seventh, at the cost of decreasing the effectiveness.
- The distance function in **EhdRgbAud** has a higher intrinsic dimensionality than **EhdGry**, thus it is more difficult to approximate because it combines more independent distances (see a similar behavior in Figure 9.1 on page 113). Therefore, T and $\mathcal{P}$ should be adjusted to increase the effectiveness at the cost of increasing the search time.

In order to overcome these issues we chose to use the Two-step search (see Section 9.1.5). Given a query video c, the first step performs approximate searches using both configurations **EhdRgb** and **Aud** in order to collect candidate reference videos $\mathcal{V}(c)$. For **EhdRgb**, the approximate search parameters were k=10, T=1%, and $|\mathcal{P}|$=5. For **Aud**, the approximate search parameters were k=30, T=2%, and $|\mathcal{P}|$=5. The set $\mathcal{V}(c)$ is created with the D most voted reference videos that received at least 2 votes. We defined parameter D=40 based exclusively on the resulting search time. The second step performs exact searches using **EhdRgbAud** between every query segment in q in c and every reference segment r in $\mathcal{R}_{av}(c) \subseteq \mathcal{R}_{av}$, where $\mathcal{R}_{av}(c)$ is the set of a+v segments for candidate videos in $\mathcal{V}(c)$.

Finally, the copy localization located copies using the (exact) k-NN lists according to **EhdRgbAud**. The chain with the highest score for each query video was reported in `nofa.EhdRgbAud`, and the two chains with highest scores were reported in `balanced.EhdRgbAud`.

We performed all the processes on a single desktop computer with Intel Core i7-2600K CPU (3.4 GHz × 4 cores) with 8 GB RAM on a GNU/Linux 2.6.38. Like our previous participation, the whole system was implemented in C using OpenCV and FFmpeg libraries.

Additionally, we performed a joint submission with Telefonica Research team for the Balanced profile, under the name `Telefonica-research.balanced.joint`. This submission tested the combination at the decision level of Telefonica's local descriptor, Telefonica's acoustic descriptor, and the **EhdRgb** configuration.

11.5.2 Results

Twenty-two teams participated in the CCD evaluation at TRECVID 2011. Each team submitted four Runs, which resulted in 32 submissions for the NoFA profile (14 Runs type V), and 41 submissions for the Balanced profile (15 Runs type V). The list of participants, the results for each participant, and the results by transformation are detailed in Appendix C.

Average Results

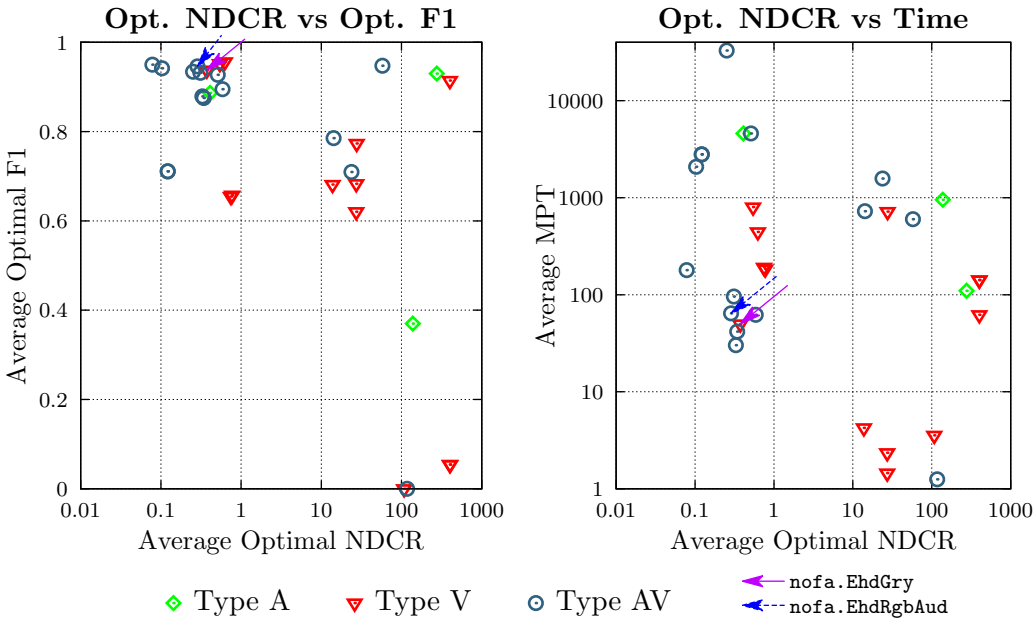
Indicator	Value	Overall Rank	Rank in Runs V
`nofa.EhdGry`			
Average Optimal NDCR	0.374	10 of 32	1 of 14
Average Optimal F1	0.938	7 of 32	3 of 14
Average Actual NDCR	0.419	4 of 32	1 of 14
Average Actual F1	0.956	3 of 32	2 of 14
Average MPT	49.9 s.	8 of 32	5 of 14
`nofa.EhdRgbAud`			
Average Optimal NDCR	0.286	6 of 32	—
Average Optimal F1	0.946	5 of 32	—
Average Actual NDCR	0.336	2 of 32	—
Average Actual F1	0.962	1 of 32	—
Average MPT	64.4 s.	12 of 32	—
`balanced.EhdGry`			
Average Optimal NDCR	0.412	18 of 41	3 of 15
Average Optimal F1	0.938	11 of 41	4 of 15
Average Actual NDCR	3.716	30 of 41	9 of 15
Average Actual F1	0.913	20 of 41	6 of 15
Average MPT	49.9 s.	7 of 41	3 of 15
`balanced.EhdRgbAud`			
Average Optimal NDCR	0.300	12 of 41	—
Average Optimal F1	0.955	3 of 41	—
Average Actual NDCR	8.462	37 of 41	—
Average Actual F1	0.935	11 of 41	—
Average MPT	64.4 s.	10 of 41	—

Table 11.8 – Evaluation for submitted Runs to TRECVID 2011 (average values for the 56 transformations).

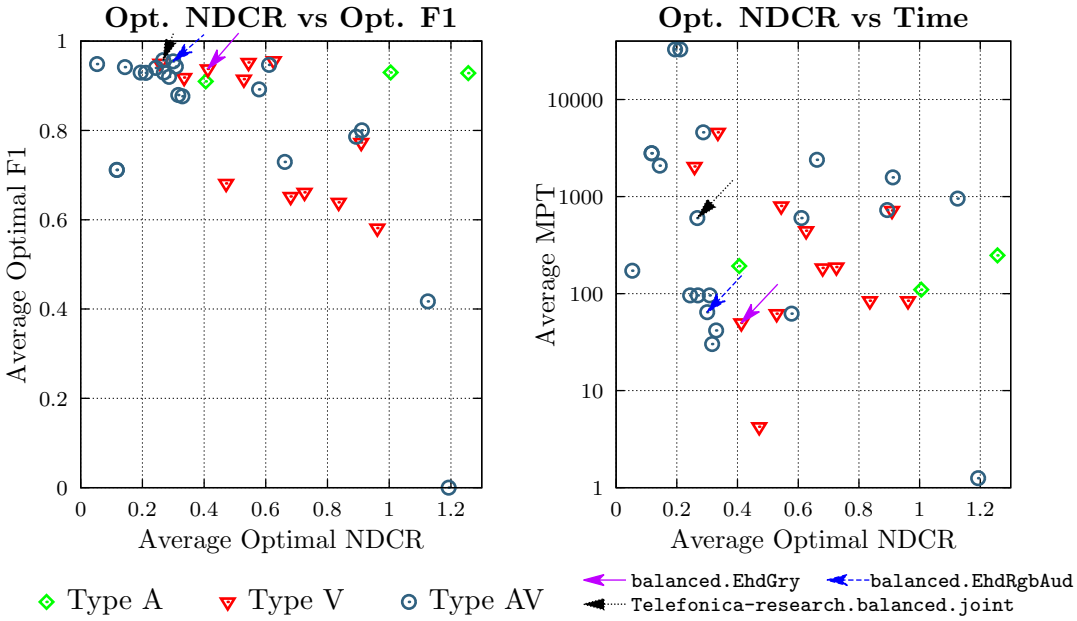
Table 11.8 shows the average results for all the transformations for the submitted Runs to NoFA profile and Balanced profile. Figure 11.4 depicts a comparison with the other Runs for Average Optimal NDCR, Average Optimal F1 and Average MPT.

In the NoFA profile, both submissions `nofa.EhdGry` and `nofa.EhdRgbAud` achieved an Average Optimal NDCR and F1 better than the median. Particularly, considering just Runs type V `nofa.EhdGry` achieved the best detection effectiveness. In the case of Actual values, the decision threshold was appropriate, because NDCR kept below 1, and `nofa.EhdRgbAud` achieved the second best Actual NDCR and the best Actual F1 overall.

In the Balanced profile, `balanced.EhdGry` and `balanced.EhdRgbAud` achieved an Average Optimal NDCR and F1 better than the median. In this case of Actual values, the decision thresholds were too permissive, because Actual NDCR was higher than 1 for both submissions.



(a) Runs for NoFA profile. The data in this figure is shown in Table C.2 on page 199.



(b) Runs for Balanced profile. The data in this figure is shown in Table C.4 on page 201.

Figure 11.4 – Average Optimal NDCR, Average Optimal F1 and Average MPT for Runs at TRECVID 2011. The ideal indicators are NDCR=0, F1=1, and MPT=0.

The improvement in detection effectiveness between `nofa.EhdGry` and `nofa.EhdRgbAud`, and between `balanced.EhdGry` and `balanced.EhdRgbAud`, proves that acoustic descriptors can be successfully combined with visual descriptors at the distance function.

All the submissions achieved a very high Optimal F1, i.e., the system correctly delimits the boundaries of the copies. This is mainly due to the Two-step search: the inaccuracies produced by the approximate searches in the first step are reduced by the exact searches in second step.

The submissions achieved better results for the NoFA profile than for the Balanced profile, which is consistent with our participation in 2011. Non-copies are usually easier to discard for global descriptors than for local descriptors, thus global descriptors may detect more correct copies before the first false alarm. However, the NDCR decreases for Balanced profile because copies with complex transformations may be undetectable for global descriptors affecting the detection rate.

In the case of Mean Processing Time, the four submissions are between the fastest Runs with good detection performance. This good balance between effectiveness and efficiency is due to a better implementation of the Preprocessing task, approximate search parameters properly adjusted, and a desktop computer with a more powerful CPU. Note also that the Two-step search can efficiently resolve audio+video searches: the MPT for audio+video submissions increased nearly 30% compared with video-only submissions, instead of the seven times as with the naive approach.

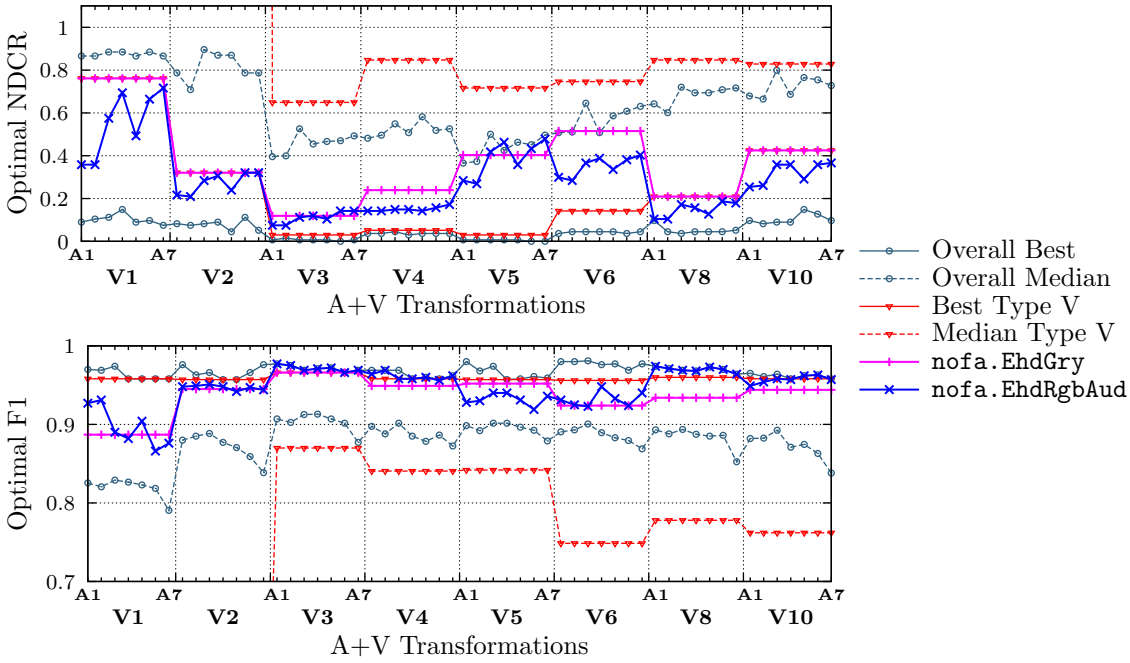
Results by Transformation

Figure 11.5 shows the results for each transformation for our submissions compared with: the best result, the median, the best result for Runs type V, and the median for Runs type V.

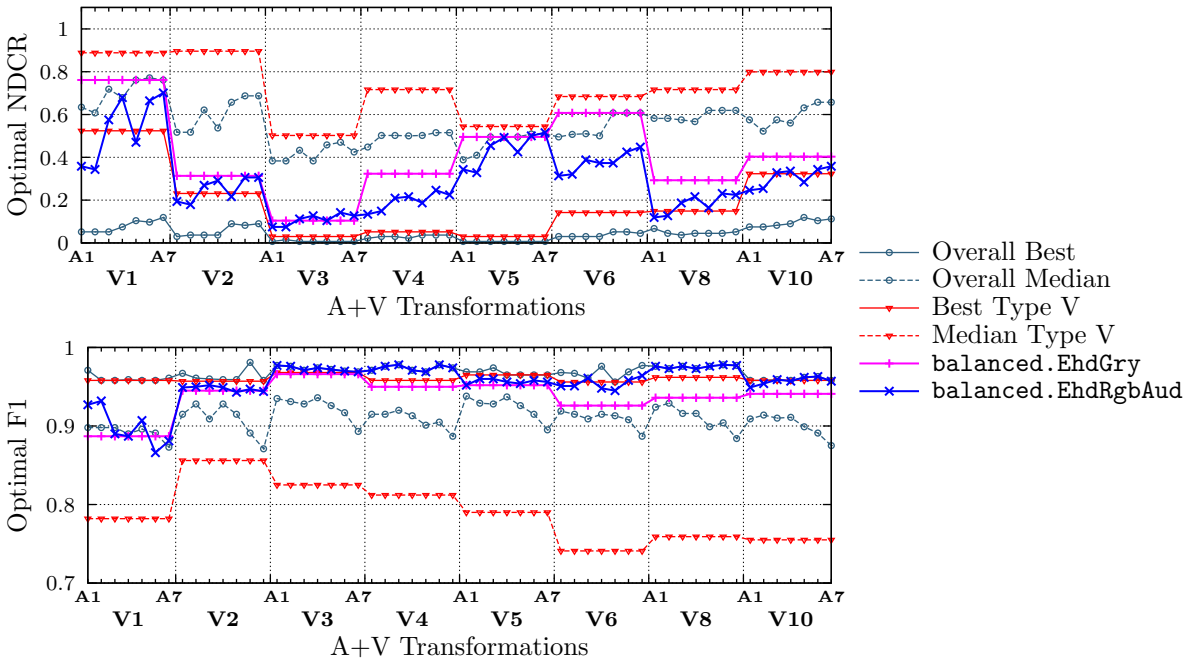
In the case of NoFA profile, `nofa.EhdGry` achieved the best detection effectiveness between Runs type V for **V1**, **V2**, **V8**, **V10**. The most difficult transformations to detect for this Run were **V1** and **V6**. In copy localization (F1), the best localization was achieved for **V3** and the worst localization for **V1**. `nofa.EhdRgbAud` improves the detection effectiveness for almost every transformation. The acoustic descriptor achieves high performance for transformations **A1**, **A2**, and **A5**. This Run achieves its best NDCR for **V3** and **V8**. In copy localization, the Run achieves the best overall localization for transformations **V3**, **V4** and **V8**.

In the case of Balanced profile, `balanced.EhdGry` achieves a better NDCR than the median for submissions Type V for every transformation, but for **V1**, **V5** and **V6** its result is worse than the overall median. It achieves a good localization (higher than median) for every transformation except for **V1**. `balanced.EhdRgbAud` achieves a better NDCR than the median for every transformation. The inclusion of the acoustic descriptor improves the effectiveness, except for **V3**, where it is outperformed by `balanced.EhdGry`. As in the NoFA profile, it achieves the best overall localization for transformations **V3**, **V4** and **V8**.

In summary, **V1** and **V2** are the most difficult visual transformations, while **A3** is the most difficult acoustic transformation for the NoFA profile, as well as **A6** for the Balanced profile (see the Overall Median line for Optimal NDCR at Figure 11.5).



(a) NoFA profile. Data shown in Table C.3 on page 200.



(b) Balanced profile. Data shown in Table C.5 on page 202.

Figure 11.5 – Optimal NDCR and Optimal F1 by transformation at TRECVID 2011.

11.6 Precision/Recall analysis

As previously described, TRECVID evaluation requests that each team adjust its system either for NoFA or Balanced profile. The evaluation divides each submission according to video transformation and it evaluates each transformation using either the optimal threshold (Optimal NDCR) or the submitted threshold (Actual NDCR). The Optimal NDCR hinders the comparison of global performances because the threshold is calculated by transformation. Actual NDCR uses the same threshold for every transformation but most of the teams failed at submitting a threshold with satisfactory performance. In previous sections, we relied on Average Optimal NDCR to evaluate the global performance, however an indicator using global thresholds would be more appropriate.

In this section we evaluate the global performance using Precision and Recall measures (see definition in Section 5.6.2). In particular, we use two values: R_{P1} (maximum Recall with Precision 1) and $R_{P.5}$ (maximum Recall with Precision greater or equal than 0.5).

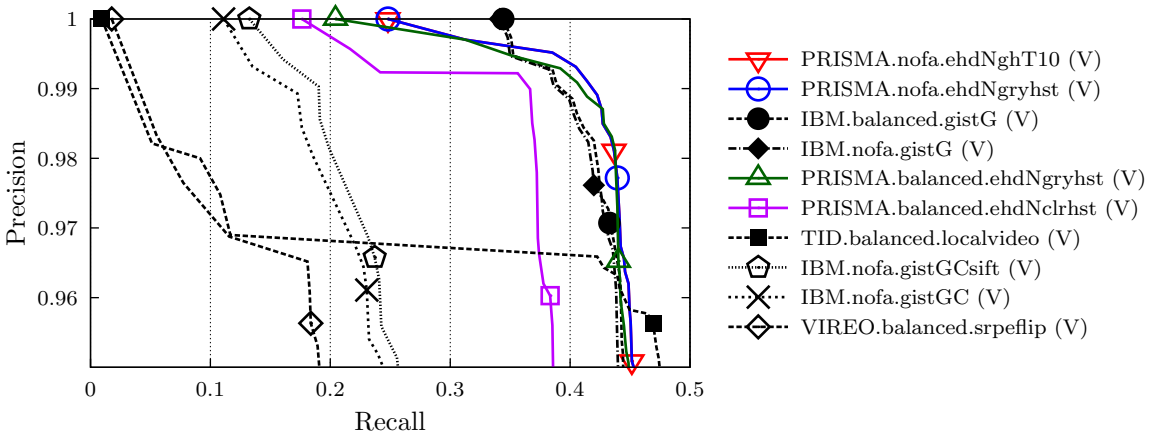
11.6.1 TRECVID 2010

Figure 11.6 depicts the precision/recall curves for selected Runs type V. Additionally, Appendix B lists R_{P1} and $R_{P.5}$ values for all submissions at TRECVID 2010. In particular, our analysis considers four Runs type V: IBM's `balanced.gistG` [Natsev et al., 2010], which uses a global descriptor based on Gabor filter responses in different scales and orientations; TID's `balanced.localvideo` [Younessian et al., 2010], which is based on DART local descriptors (an alternative to SIFT descriptors) and computing spatial coherence with RANSAC; VIREO's `balanced.srpeflip` [Ngo et al., 2010], which follows the codebook approach with Hamming Embedding; and our `balanced.ehdNgryhst` [Barrios and Bustos, 2010], already described in Section 11.4.1.

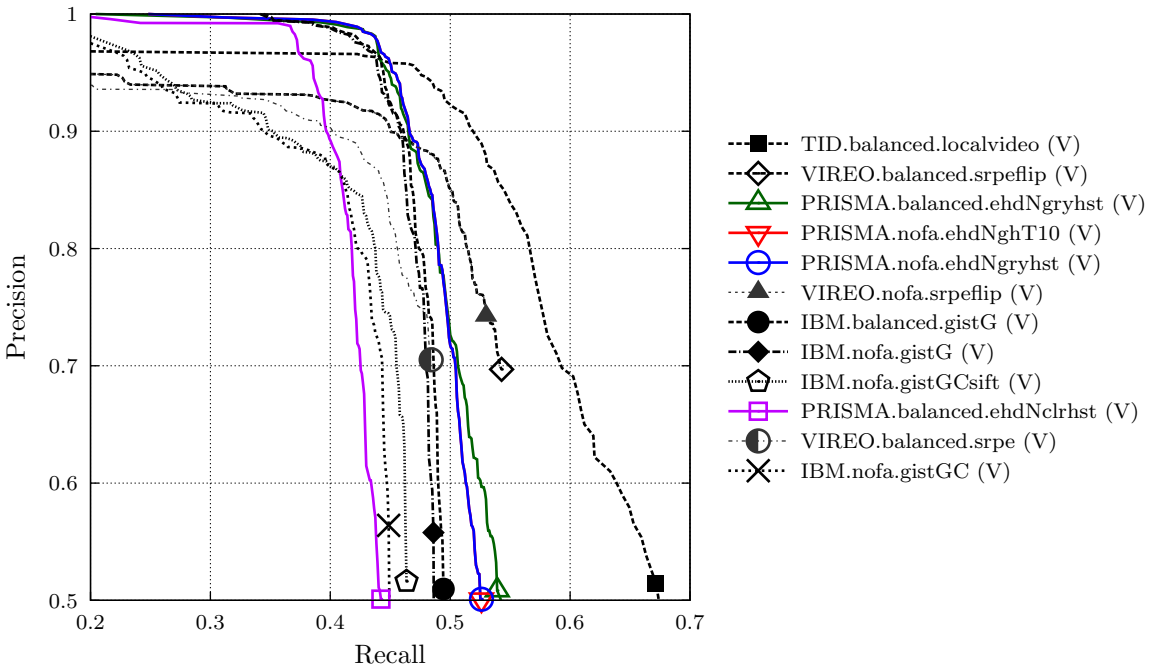
The data suggests that Runs based on global descriptors achieve the best performance at detecting copies without reporting false alarms. In fact, `balanced.gistG` detects 34% of the copies and `nofa.ehdNgryhst` detects 25% of copies before reporting the first false alarm, compared to less than 2% for `balanced.srpeflip` and `balanced.localvideo`. On the other hand, Runs based on local descriptors achieve better performance in a balanced scenario: `balanced.localvideo` detects 67% of copies and `balanced.srpeflip` detects 54.3% of copies when producing a similar amount of correct and incorrect detections, compared to 53.9% for `balanced.ehdNgryhst` and 49% for `balanced.gistG`.

An interpretation for this behavior is that non-copy query videos are usually easier to discard for global descriptors than for local descriptors. In fact, TID team reported that local descriptors may trigger a false detection between two unrelated videos if they share captions or logos. This behavior is unlikely for global descriptors because captions and logos only change small zones. Hence, the amount of correct detections up to the first false alarm can be higher for global descriptors than for local descriptors. However, as precision decreases local descriptors detect transformations that may be undetectable for global descriptors.

Figure 11.7 compares the performance for the four Runs. It summarizes the detection performance for each one of the 1072 visual queries: 67 base queries with a full copy (red pixels) plus 67 base queries with a copy excerpt (green pixel), and each base video with eight visual transformations (from **V1** to **V10**). A dark red/green represents a correct detection at R_{P1}, a light red/green represents a correct detection at $R_{P.5}$, yellow represents an undetected copy neither at R_{P1} nor $R_{P.5}$.

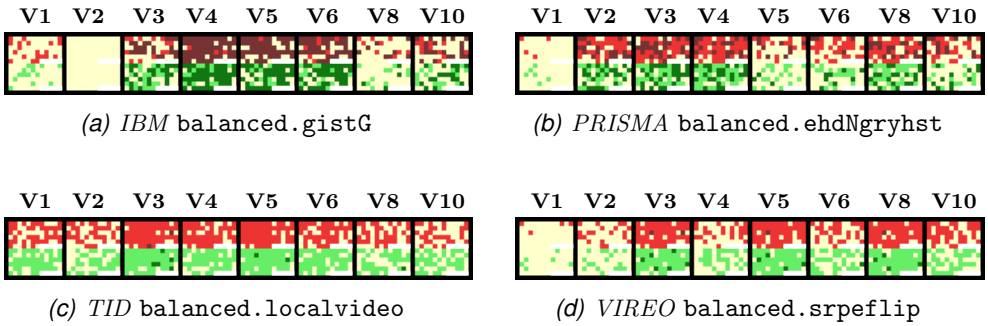


(a) Precision/Recall curves for submissions type V with high R_{P1}. Data shown in Table B.6 on page 194. Additional markers have been added to each curve (at an arbitrary position) to make them more visible.



(b) Precision/Recall curves for submissions type V with high $R_{P.5}$. The marker shows the $R_{P.5}$ on each precision/recall curve. Data shown in Table B.8 on page 196.

Figure 11.6 – Precision/Recall curves for selected submissions type V to TRECVID 2010.



(a) IBM `balanced.gistG` *(b) PRISMA* `balanced.ehdNgryhst`

(c) TID `balanced.localvideo` *(d) VIREO* `balanced.srpeflip`

Figure 11.7 – Copy detections at R_{P1} and $R_{P.5}$ at TRECVID 2010. Each pixel represents a visual query video. Queries containing a full copy are shown in red, queries containing a copy excerpt are shown in green. A dark red/green represents a correct detection at R_{P1}, a light red/green represents a correct detection at $R_{P.5}$, yellow represents an undetected copy neither at R_{P1} nor $R_{P.5}$.

The figure shows that submission `balanced.gistG` achieves high detection performance at transformations **V4**, **V5** and **V6**, thus GIST descriptor is highly robust to quality transformations. In fact, most of the queries are detected at R_{P1}. However, the descriptor shows weak performance at postproduction transformations.

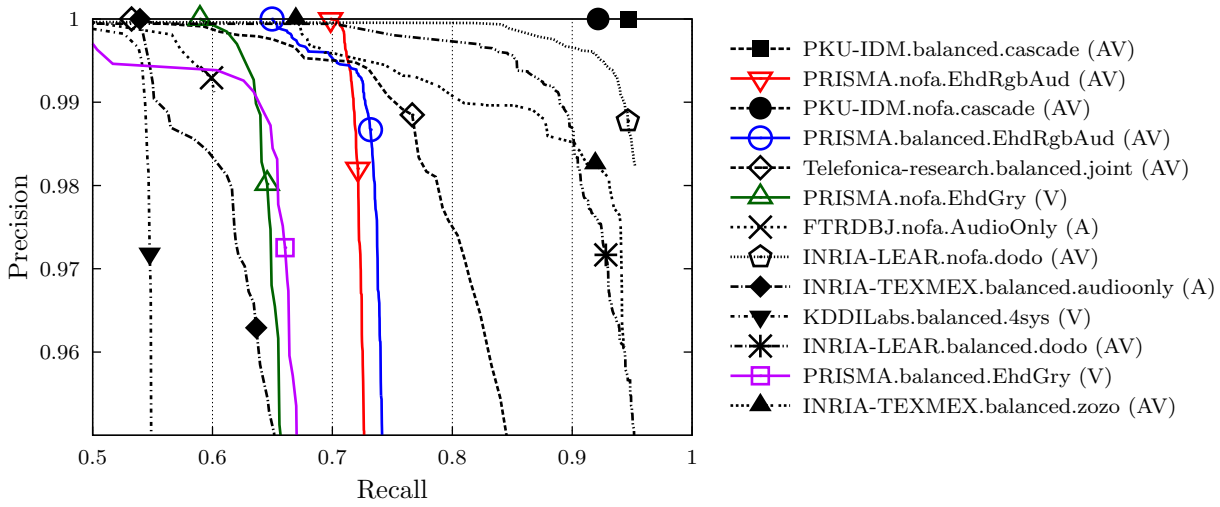
Detections at `balanced.ehdNgryhst` are sparsely distributed between **V2** and **V10**. In fact, between the 134 base videos it successfully detects at least one copy from 126 base videos (94%). In comparison, `balanced.gistG` was able to detect at least one copy from 122 base videos (91%), `balanced.srpeflip` was able to detect at least one copy from 121 base videos (90%), and `balanced.localvideo` was able to detect at least one copy from 126 base videos (94%). This behavior shows the combination of global descriptors can achieve high performance, even comparable to local descriptors. `balanced.ehdNgryhst` may have achieved higher performance with more suitable approximate search parameters.

Regarding the reported processing times of the four Runs, `balanced.gistG` is the fastest (it uses an approximate search for GIST descriptors using FLANN), `balanced.ehdNgryhst` is the second fastest, while both Runs based on local descriptors are slower.

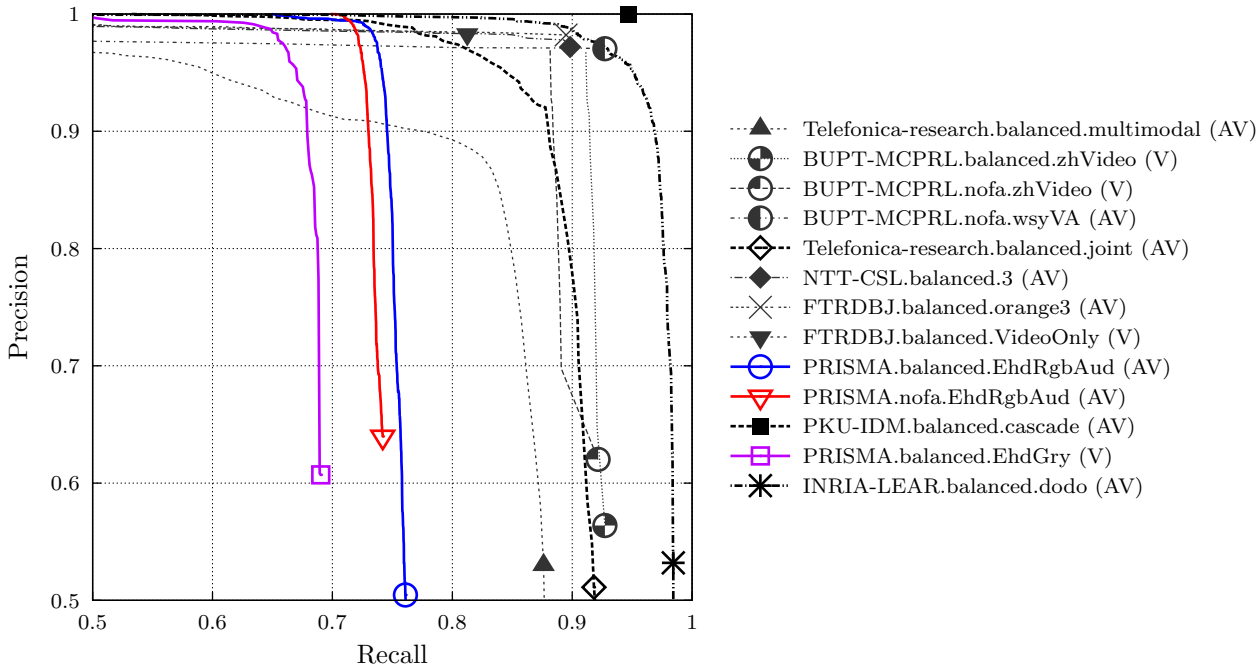
In summary, the combination of distances could have achieved high detection performance, however the approximate search was not properly configured in order to profit from the potentially high detection performance.

11.6.2 TRECVID 2011

Figure 11.8 depicts the precision/recall curves for Runs with high detection performance. Additionally, Appendix C lists R_{P1} and $R_{P.5}$ values for all submissions at TRECVID 2011. We compare the results in two scenarios: Runs type V and Runs type AV.

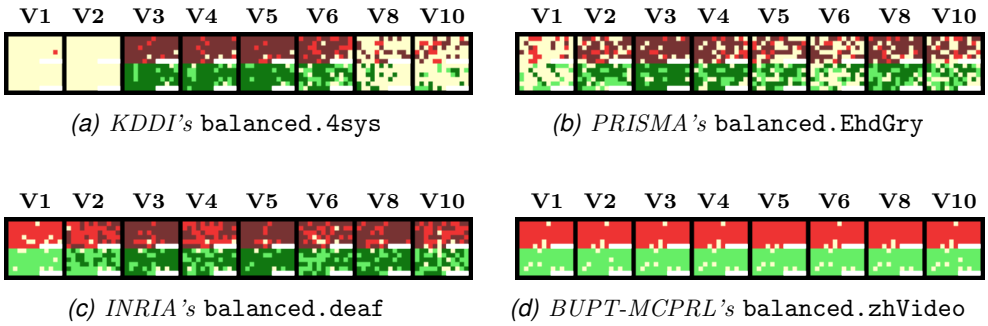


(a) Precision/Recall curves for submissions with high R_{P1}. Data shown in Table C.6 on page 203. Additional markers have been added to each curve (at an arbitrary position) to make them more visible.



(b) Precision/Recall curves for submissions with high $R_{P.5}$. The marker shows the $R_{P.5}$ on each precision/recall curve. Data shown in Table C.8 on page 205.

Figure 11.8 – Precision/Recall curves for selected submissions to TRECVID 2011.



(a) KDDI's `balanced.4sys` *(b) PRISMA's* `balanced.EhdGry`

(c) INRIA's `balanced.deaf` *(d) BUPT-MCPRL's* `balanced.zhVideo`

Figure 11.9 – Copy detections at R_{P1} and $R_{P.5}$ at TRECVID 2011. See Figure 11.7 for an explanation of the figure.

Runs Type V

We analyze the detection performance of four submissions Runs type V: our `nofa.EhdGry`, KDDI's `balanced.4sys`, INRIA's `balanced.deaf`, and BUPT-MCPRL's `balanced.zhVideo`. Figure 11.9 details the performance at every query video, comparing the performance at R_{P1} (darker red/green) and $R_{P.5}$ (lighter red/green), divided by visual transformation.

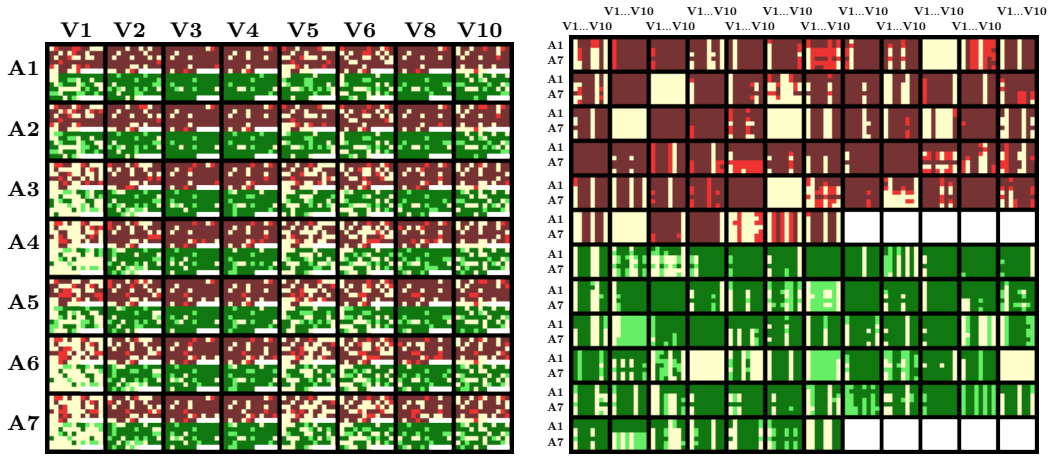
PRISMA's `nofa.EhdGry` achieves the highest detection performance at R_{P1} detecting 59% of copies before reporting the first false alarm. The improvement compared with our previous participation, where `nofa.ehdNgryhst` achieved R_{P1} 24%, is mainly due to better choice of approximate search parameters. When a similar amount of correct and incorrect detections are possible, `nofa.EhdGry` achieved $R_{P.5}$ 69% correct detections.

KDDI's `balanced.4sys` [Uchida et al., 2011] computes global descriptors (DCT descriptors at different frame sizes) and a search based on lookup tables. The Run achieved R_{P1} 45% and $R_{P.5}$ 55%, hence it was outperformed by `nofa.EhdGry`. In fact, `balanced.4sys` achieves a very high performance at quality transformations (**V3** to **V6**), but a very low performance at postproduction transformation.

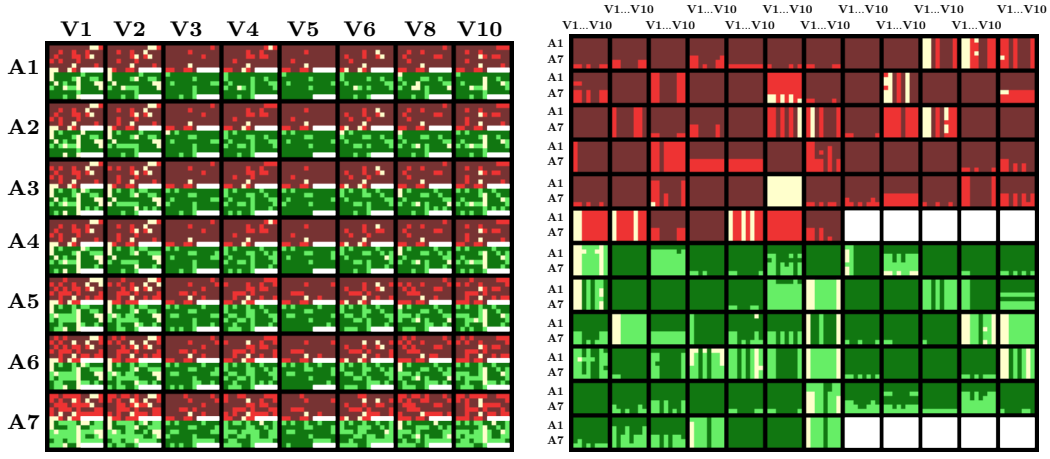
INRIA's `balanced.deaf` [Ayari et al., 2011] computes CS-LBP descriptors (an alternative to SIFT descriptors) and a codebook with hamming embedding. That submission achieved R_{P1} 54%, hence it is outperformed by `nofa.EhdGry`, but it achieves a strong performance at $R_{P.5}$ detecting 94% copies.

BUPT-MCPRL's `balanced.zhVideo` [Zhao et al., 2011b] combines SIFT descriptors and different global descriptors (binary patterns, color correlogram, HOG). It failed at detecting copies without false alarms, but it achieved a strong $R_{P.5}$ 93% at the balance scenario.

In summary, the results for `nofa.EhdGry` confirm the good performance at detecting copies without false alarms, and that a combined distance is able to detect copies either for quality and postproduction transformation, but it the balanced scenario its detection performance is lower than CBVCD system based on local descriptors.

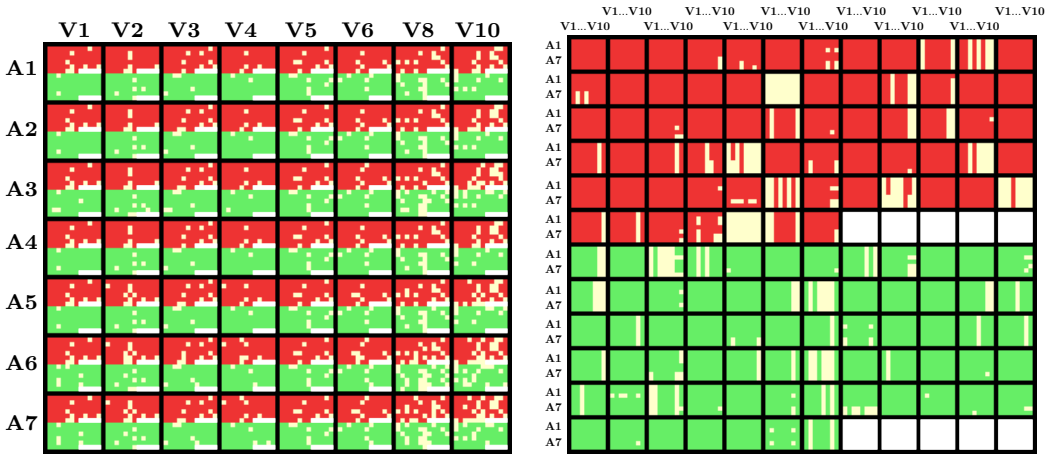


(a) PRISMA's `balanced.EhdRgbAud`

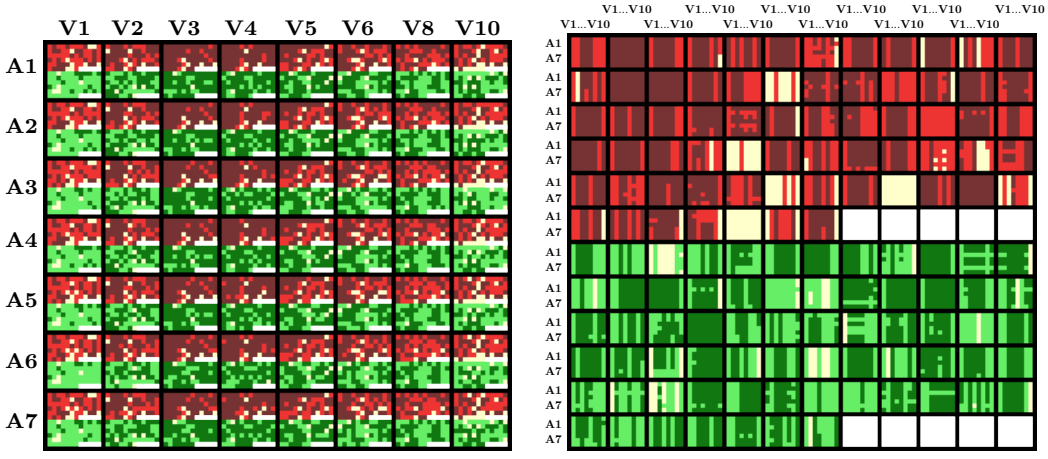


(b) CRIM's `balanced.V48A66T58B`

Figure 11.10 – Copy detections at R_{P1} and $R_{P.5}$ at TRECVID 2011. On the left side the queries are grouped by a+v transformation. On the right side the queries are grouped by base video.



(a) Telefonica's `balanced.multimodal`



(b) Telefonica's `balanced.joint`

Figure 11.11 – Results achieved by the fusion between our system into Telefonica's system. (a) Basal performance of Telefonica's system. (b) Performance of fusion between Telefonica's and our system.

Runs Type AV

Regarding the NoFA scenario, PKU-IDM's `balanced.cascade` [Jiang et al., 2011] achieved the highest R_{P1} with 94.7% correct detections without false alarms. Paradoxically, it was submitted to the balanced profile. Thereafter, our `nofa.EhdRgbAud` achieves the second highest R_{P1} with 69.9% correct detections without false alarms. A slightly lower result is achieved by CRIM's submissions [Gupta et al., 2011] with R_{P1} 69.6%, which performs a late fusion of acoustic and global descriptors and linear scan implemented in GPU.

Regarding the balanced scenario, INRIA's `balanced.dodo` [Ayari et al., 2011] achieves the highest $R_{P.5}$ with 98.4% correct detections. Thereafter, both PKU-IDM's `balanced.cascade` and BUPT-MCPRL's `balanced.wsyVA` achieve $R_{P.5}$ 94.7%, while CRIM achieved $R_{P.5}$ 94.4%. Our submission `balanced.EhdRgbAud` achieves a much lower result with $R_{P.5}$ 76.1%.

Figure 11.10 shows a detailed comparison between `balanced.EhdRgbAud` and CRIM's `balanced.V48A66T58B`. The figure summarizes the detection performance for each one of the 7504 a+v queries: 67 base queries with a full copy (red pixels) plus 67 base queries with a copy excerpt (green pixel), and each base video with the combination between eight visual transformations (from **V1** to **V10**) and seven acoustic transformations (from **A1** to **A8**). A dark red/green represents a correct detection at R_{P1}, a light red/green represents a correct detection at $R_{P.5}$, yellow represents an undetected copy neither at R_{P1} nor $R_{P.5}$. The figure organizes queries according to two different criteria: left-side figure groups queries by a+v transformation, thus revealing their impact on detection effectiveness; right-side figure groups queries by base video, thus revealing the influence of actual audiovisual information on detection effectiveness.

The figure shows that `balanced.EhdRgbAud` do not perform well for transformations **V1**, **V5** and **V6**. In those cases the audio information slightly improves the performance for transformations **A1** and **A2**. These results evince the selected fusion highly favors the visual information, which is due to the reduction in the weight applied to acoustic distance (described in Section 11.5.1).

On the other hand, CRIM's `balanced.V48A66T58B`, which uses a late fusion that considers global and acoustic descriptors, shows a similar performance at R_{P1} but a much higher performance at $R_{P.5}$. This is result of more appropriate global descriptors and fusion technique. The quality of the descriptors becomes evident when `balanced.EhdRgbAud` fully detects 10 base videos at R_{P1} (i.e., to detect the base video in every a+v transformation), while `balanced.V48A66T58B` fully detects 36 base videos. The late fusion enables to detect copies using just one modality (a copy can be detected in either visual or audio track), while the fusion of distances mainly detects copies on both modalities at the same time. Additionally, the use of linear scans also contributes to CRIM's high performance: in fact, NTT-SCL's `balanced.3` [Mukai et al., 2011] uses the same global descriptor as CRIM but resolves the searches with lookup tables and codebooks and it achieves a lower $R_{P.5}$ 89.8%.

Figure 11.11 shows a comparison between Telefonica's `balanced.multimodal` and `balanced.joint`. `balanced.multimodal` is a Run that makes a late fusion between local descriptors and audio. `balanced.joint` includes a late fusion with the result of PRISMA's copy detection based on global descriptors. The figure reveals the improvement at R_{P1} is evident (from 0% to 53.3%). However, the improvement at $R_{P.5}$ is rather marginal (from 87.6% to 91.8%). Therefore, the inclusion of global descriptors has a large impact on the number of detections without false alarms, but it has small impact at detecting new copies that are not detectable by local descriptors.

In summary, the fusion of distances is an alternative to combine global and acoustic de-

scriptors which improves the detection performance without producing false alarms. However, in a balanced scenario the fusion of distances only has small impact on detection performance.

11.6.3 Evaluation of preprocessing task

As shown in Chapter 7, the preprocessing task has a big impact on global descriptors detection performance. To estimate its impact, we assume query videos with camcording (**V1**) or PIP (**V2**) become undetectable when the preprocessing task is not present. We then calculate the precision/recall curves for our submissions type V when all the detections for query videos with transformations **V1** or **V2** have been discarded (assuming the non-preprocessed videos would not generate false alarms).

Figure 11.12 shows the precision/recall curves for four submissions type V. In TRECVID 2010 submissions, the reversion of camcording had a little impact on effectiveness (about 0.01 points in recall) and the reversion of PIP had a bigger impact (about 0.08 points in recall). In fact, TRECVID query collection comprises 12.5% videos of each visual transformation, thus the preprocess helped detect about two thirds of all the queries with PIP. In TRECVID 2011, we improved our implementation for the preprocessing task, thus the reversion of camcording and PIP had a bigger impact on effectiveness (about 0.04 and 0.09 points in recall, respectively).

These results show discarding the preprocess would worsen the detection performance because global descriptors are highly affected by camcording and PIP transformations. In fact, a decrease of 0.1 points in R_{P1} and $R_{P.5}$ would highly impact the performance of our Runs (see Table B.6 on page 194 and Table C.6 on page 203). On the other hand, without detection and reversion of transformations, the number of query videos will be reduced, hence there will be more room for improving the accuracy of the approximate search without increasing the total search time.

11.7 Summary

In this chapter we reviewed our participation in the CCD evaluation at TRECVID 2010 and 2011.

In our participation at TRECVID 2010 we submitted four Runs. To create these Runs we used video preprocessing, spatio-temporal global descriptors, weighting by max-ρ, approximate search with pivots, and temporal coherence between nearest neighbors. The Runs showed competitive performance, especially for detecting copies without false alarms, outperforming most of the systems that only used visual information.

In our participation at TRECVID 2011 we again submitted four Runs. Two of them used global descriptors, and the other two included a combination of global and acoustic descriptors. The results showed a big improvement with respect to the previous participation, mainly due to a better implementation for the preprocessing task, better parameters for descriptors and distance function, and a Two-step search which mixes the approximate search and exact search. The Runs achieved a good performance, outperforming many systems, especially for detections without false alarms.

These results show that global descriptors can achieve competitive performance compared with other state-of-the-art systems (which are usually based on local descriptors), in particular, for

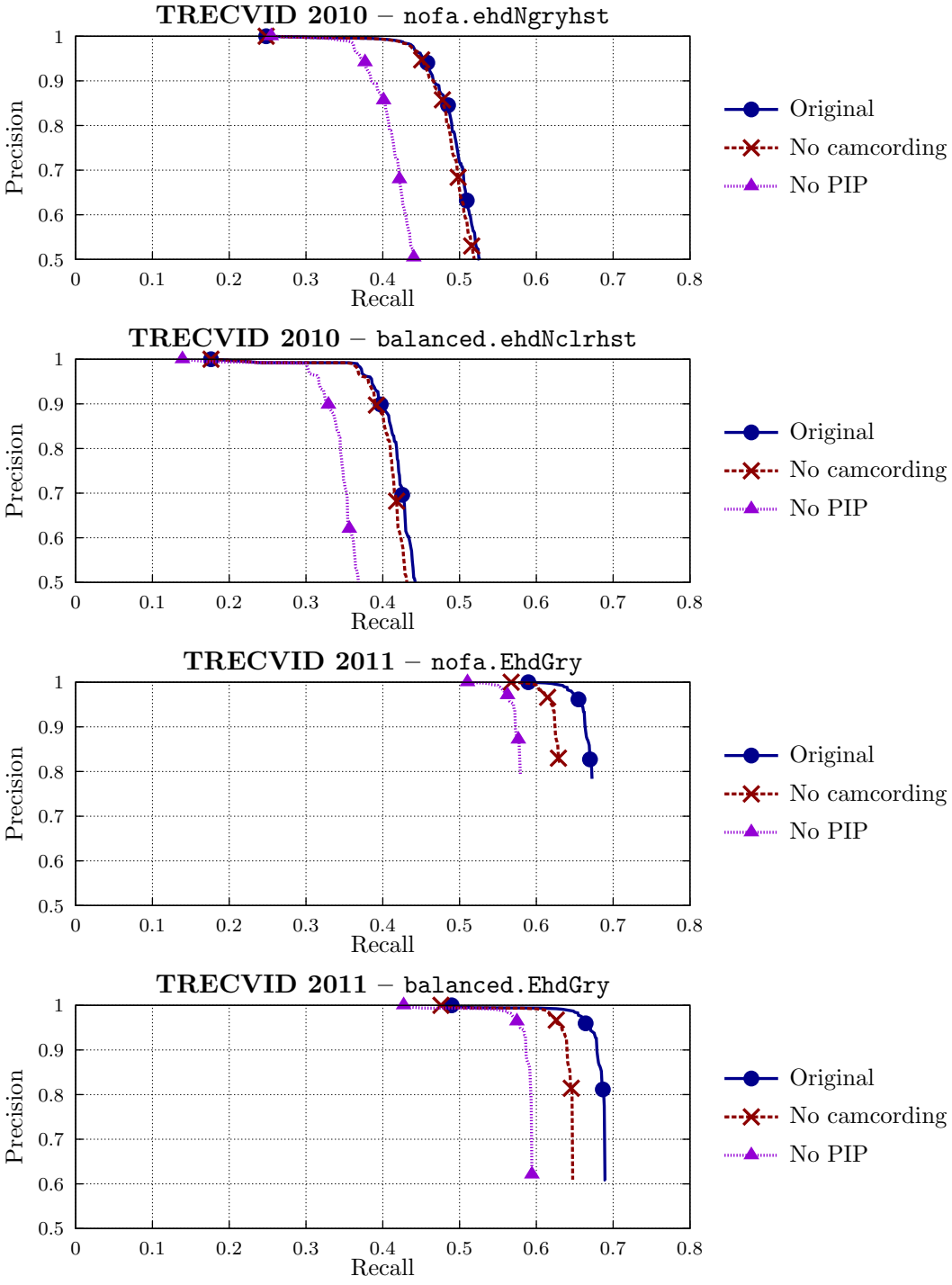


Figure 11.12 – Precision/Recall curves for the original Run, and for a Run discarding every detection from queries with either camcording or PIP transformations.

detection without false alarms. The distance fusion between global and acoustic descriptors can achieve a high performance.

Chapter 12

Conclusions

In this thesis, we have detailed a novel CBVCD system, called P-VCD. It is founded on the metric space approach and uses many novel algorithms and techniques in order to achieve high effectiveness and efficiency. P-VCD has been released as an Open Source Project under the GNU General Public License version 3.0 (GPLv3). Most of the source code used in this thesis can be freely downloaded from its website[1].

The system has been evaluated using the MUSCLE-VCD-2007 dataset (which is a medium-size dataset), and by participating in the CBVCD evaluation at TRECVID 2010 and 2011 (which uses large datasets). P-VCD shows high performance in both evaluations. In the case of MUSCLE-VCD-2007, it outperforms the best published result for the dataset, achieving the maximum detection effectiveness. In the case of TRECVID, it shows competitive performance with other state-of-the-art CBVCD systems. Therefore, this thesis exhibits an approach with high potential that we believe makes a valuable contribution to the CBMIR area.

12.1 Summary of the main contributions

Our proposed approach for CBVCD consists of five main tasks: Preprocessing, Video Segmentation, Feature Extraction, Similarity Search, and Copy Localization. The main contributions of this thesis, divided by task, can be summarized as follows:

- Regarding the Preprocessing task:
  - We have shown that the preprocessing of videos can improve detection effectiveness. In particular, we presented two processes: one for quality normalization and one for reversion of transformations, both producing a big improvement in the detection effectiveness.
- Regarding the Video Segmentation and Feature Extraction tasks:
  - We have reviewed different global descriptors, performed experimental evaluations of them using different segmentations, analyzed the results, and have defined some guidelines to design a good descriptor.

[1] P-VCD: `http://sourceforge.net/projects/p-vcd/`

  - We have developed a spatio-temporal extension for frame-based global descriptors showing that the spatio-temporal description improves the quality of the descriptors without affecting the similarity search, and that rather simple global descriptors can achieve high detection performance. In particular, the descriptors based on orientation of edges achieve high detection effectiveness.
  - We have proposed an acoustic descriptor that can be used with the metric space approach. The descriptor achieves high detection performance and can be seamlessly combined with global descriptors.

- Regarding the effectiveness of the Similarity Search task:
  - We have shown that the combination of distances at the similarity search can improve the effectiveness of the search. We developed three novel techniques to automatize the selection of weights in the combination: the α-normalization, the weighting by max-ρ, and the weighting by max-τ. We analyzed these algorithms showing they enable automatically setting good weights without requiring the use of training data.
  - We have shown that the spatio-temporal combined distance can improve the effectiveness of the search. In particular, we have shown that the spatio-temporal combination of the distances between the Edge Histogram descriptor and the novel acoustic descriptor can achieve the maximum effectiveness for MUSCLE-VCD-2007, i.e., to detect all the copies without giving any false alarms. This result outperforms the state-of-the-art systems evaluated on this dataset.
  - We have analyzed the use of some common non-metric distances in order to improve the effectiveness. We have concluded the evaluated non-metric distances present only small gains at the cost of decreasing the efficiency, and therefore the metric distance L_1 achieves the best tradeoff between effectiveness and efficiency.

- Regarding the efficiency of the Similarity Search task:
  - We have developed the Approximate Search with Pivots, which uses static pivots to estimate and discard most of the distance evaluations. We have shown that this approximate search shows a convenient *effectiveness-versus-efficiency* trade-off. We have tested this search in MUSCLE-VCD-2007 and TRECVID datasets with good results.
  - We have developed an approach to apply the Approximate Search with Pivots to local descriptors. We have shown that this approximate search can outperform the traditional search approach for local descriptors in both effectiveness and efficiency.
  - We have developed the Two-step search, which enables the use of complex combined distances in large datasets. We have tested this technique to perform a combined audio-visual search in the TRECVID dataset with satisfactory results.
  - We have developed the Snake Table, which uses previous query objects as dynamic pivots to resolve exact searches. We have defined the snake distribution, and we have shown experimentally that query objects in a CBVCD system fit that distribution. We have shown the snake distribution enables a novel approach to index metric spaces with high intrinsic dimensionality.

- Regarding the Copy Localization task:
  - We have developed a voting algorithm that determines the boundaries of a copy by analyzing the objects retrieved by the similarity search. Experimentally we have shown it can improve its effectiveness when it considers more than one similar object by weighting the votes according to rank positions.

Regarding our participation in the CCD evaluation at TRECVID, we have validated that our approach for CBVCD and the developed techniques are indeed relevant to the current state-of-the-art.

12.2 Benefits and drawbacks of the proposed solution

The results presented in this thesis prove the proposed solution can successfully address high effectiveness and efficiency. However, we must note there exist alternatives that can be more suitable than our approach in some situations. For instance, in Section 3.4 we discuss advantages and disadvantages between vector spaces and metric spaces.

In general, the benefits and drawbacks of the proposed solution are related to the effectiveness-versus-efficiency tradeoff presented in the form "complex-vs-simple similarity measure". By simple similarity measures we refer to distances that compare vectors directly by their dimensions, like traditional Minkowski distances. By complex similarity measures we refer to distances that may combine distances between descriptors, compute temporal correlation, or even compute the solution for an optimization problem. In general, the former distances are the basis for vector spaces, while the latter are the basis for the metric spaces. The following list intends to summarize the strengths and weaknesses of the proposed approach:

- Pro: A complex similarity measure can achieve higher effectiveness than simple distance functions (see Chapter 8). In fact, complex distances ease the combination of descriptors, the usage of variable-length descriptors, multimodal descriptors, and even non-vectorial descriptors, which may improve the effectiveness of a system.
- Pro: The metric approach enables to improve the efficiency for complex similarity measures, as long as they satisfy the metric properties. Therefore, even complex distances can use metric indexes to outperform the linear scan. Moreover, videos usually show similarity between consecutive frames which can easily be used by a metric index to improve efficiency (see Snake Table in Chapter 9).
- Con: A complex similarity measure usually is computationally expensive and produces a space with high intrinsic dimensionality. Therefore, in scenarios involving large amounts of data (which is a common case in video domain), the lightweight measures (like Euclidean distance) are sometimes the only affordable functions that can be used. In those cases, highly time-efficient techniques (like data projections or space divisions) show much higher scalability than our approach. In fact, based on the results shown in this thesis, a medium-size dataset (like MUSCLE-VCD-2007) can successfully be addressed by our approach, a large dataset (like TRECVID's CCD) can be addressed with satisfactory results, but a very large dataset (like the dataset described by Poullot et al. [2010]) is far beyond the possibilities of our approach (yet).
- Con: The similarity measure must satisfy the metric properties in order to apply the metric approach. Unfortunately, some convenient techniques (like adaptive weights, partial similarity, and even the minimum of two distances) may yield to break some metric property. Particularly to video domain, the triangle inequality hinders the definition of a function for detecting copies either in the visual track or in the audio track. In that situation, a similarity measure that does not (always) satisfy the metric properties or a late fusion of partial results may be considered.

- Pro: The global efficiency of a system can generally be improved by investing in better hardware. Additionally, better hardware also enables to compute more descriptors and reduce the search approximation, which may improve the effectiveness, but to a limited extent. On the other hand, enhancing similarity models can yield to improvements in effectiveness that could have never been reached by hardware enhancement. In other words, the lower efficiency that is shown by metric spaces can be overcome by just improving the hardware, while the low effectiveness shown by simple similarity models cannot be overcome just by better hardware.

12.3 Trends for future research

In this section we outline the open issues and research trends based on the results presented in this thesis:

- Regarding the effectiveness of the search:
  - We plan to study and develop an "universal normalization", i.e., given a distance d, to define the normalized function $\hat{d}$ as $\hat{d}(x, y) = F_d(d(x, y))$, where F_d is the cumulative distribution (see Section 8.1.1). This universal normalization is like the α-normalization with dynamic weights. The $\hat{d}$ is bounded to $[0, 1]$ and thus it can be used to normalize and combine any distance. However, in general, $\hat{d}$ does not satisfy the triangle inequality. More work is needed in order to study $\hat{d}$, its properties, benefits, and drawbacks.
  - During the experimental analysis of the weighting by max-ρ, we concluded that the weights that maximize MAP do not coincide with the weights that maximize ρ. In fact, the analysis showed configurations that can achieve both high effectiveness and high efficiency (see Figure 8.3 on page 97). We plan to study the characteristics of those configurations and develop criteria to locate them.
  - During the discussion of effectiveness versus efficiency at Section 8.1.2, we stated that a perfectly discriminant function d', which returns a small value to the correct objects and a large constant value to the irrelevant objects, would achieve maximum effectiveness and also a near-infinite intrinsic dimensionality. However, if d' returns uniformly distributed values to the irrelevant objects (instead of a constant value), the intrinsic dimensionality would decrease while maintaining the maximum effectiveness. Therefore, it is possible to improve the efficiency of a distance without compromising its effectiveness. We plan to develop this idea in order to define distances that achieve both high effectiveness and efficiency.
  - Using the approach of weighted combination of distances, we successfully combined distances from global descriptors and acoustic descriptors. This combination assumes that the copy exists in both modalities, i.e., in the visual and audio tracks at the same time. However, when the copy exists in only one modality, the combined function may not work as expected. We plan to address the detection of copies that exist only in one modality.
  - Another open problem is to successfully combine global, acoustic, and local descriptors in a single distance function. In preliminary experiments, we tested the weighted combination between L_1 and Matches (see Section 7.5), however the result was unsatisfactory due to their different properties and behavior, hence a different distance for local descriptors is needed. In particular, we plan to test a distance using a dense sampling of local descriptors, and a distance based on global summarizations of local descriptors, like BOW or glocal descriptors.

  - The algorithms we have developed to automatically combine distances (i.e., α-normalization weighting by max-ρ, and weighting by max-τ) are general enough to be applied to other CBMIR problems. We plan to use them in different scenarios with different descriptors and objectives in order to test their behavior. In particular, we plan to use them in a content-based image retrieval system and a 3D object retrieval.
  - In this thesis we have developed an early fusion at the similarity search (see Section 4.6.1). In the case of classification and semantic indexing, the common approach for the early fusion is at the content description, i.e., the descriptors from multimodal sources are combined in a unique descriptor which is the input to the classifier. We plan to develop the "distance fusion" approach, which is to calculate the distance for each descriptor, and to classify the vector of distances. Hence, this approach proposes to perform the classification in a "distances space" instead of the descriptors space.
  - In Section 9.3.3 we compare the linear combination of distances and the fusion of nearest neighbors, and we showed the fusion of nearest neighbors outperforms the linear combination in both effectiveness and efficiency. The benefits come from discarding irrelevant objects before combining the distances. We plan to continue the research on developing similarity measures based on distance aggregation of partial nearest neighbors.

- Regarding the efficiency of the search:
  - At Section 9.1.5 we described a sort of paradox that occurs when improving the distance in the approximate search: a distance with higher effectiveness usually implies a search space with higher intrinsic dimensionality, in turn, this produces a decrease in the quality of the estimator function, which may lead to a decrease in the effectiveness of the approximate search. This effect can be seen in Figure 9.1 on page 113, where the approximations of **KF** outperform the approximations of **EH**, despite than an exact search of **EH** outperforms **KF**. More work is needed in order to analyze this effect.
  - In the experimental evaluation of the approximate search with pivots for local descriptors at Section 9.1.6, we declared there is a limit for performing an approximate search with high effectiveness. That limit is given by the number of SIFT descriptors, which in turn depends on the size of the dataset. We plan to address the issue of scaling the approximate search beyond that limit. In this case, it is necessary to evaluate other local descriptors in order to determine the one that produces a search space with lower intrinsic dimensionality than SIFT.
  - The approximate search with pivots uses the maximum lower bound as an estimator. We have tested other simple estimators (like minimum upper bound, and the average of lower and upper bounds) with unsatisfactory results. Depending on the properties of the actual distance to evaluate, there may be other properties to exploit in the definition of the estimation. We plan to study this issue.
  - The Snake Table can achieve high efficiency on metric spaces with high intrinsic dimensionality if the queries fit a snake distribution. However, if the query set does not fit a snake distribution, the Snake Table may not achieve satisfactory results. In some cases, there may be a reordering of the query objects that improves the snake distribution. We have produced some advances in this topic [Barrios et al., 2013].

- Regarding other aspects of the thesis:
  - The preprocessing of videos improves the detection effectiveness. The quality normalization improves the effectiveness with a minimal impact in search times. However, the detection and reversion of transformations highly affects the efficiency of the system.

Moreover, the proposed reversion methods are specific to TRECVID datasets, hence more work is needed in order to determine realistic transformation to revert and to develop implementations that do not harm the efficiency of the system.

- The benchmark we performed for global descriptors included many descriptors and techniques, however there are many more techniques and configurations that were not tested. We plan to extend this evaluation in order to include other descriptors from the MPEG-7 standard, other color spaces, and other parameters for frame zoning and quantization.
- We presented an approach to extend frame-based global descriptors into spatio-temporal global descriptors. An open issue is the generalization of this approach to local descriptors. This generalization would increase the quality of local descriptors for videos, without needing to define a complex spatio-temporal local descriptor.
- P-VCD has been released as an Open Source project. This software uses command line interface, hence it requires experienced users. We plan to work on this project in order to improve the documentation and provide an end-user interface.

Bibliography

C. Aggarwal and P. Yu. The IGrid index: reversing the dimensionality curse for similarity indexing in high dimensional space. In *Proc. of the 6h ACM SIGKDD int. conf. on Knowledge Discovery and Data mining (KDD)*, pages 119–129. ACM, 2000.

C. Aggarwal, A. Hinneburg, and D. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Proc. of the 8th int. conf. on Database Theory (ICDT)*, pages 420–434. Springer, 2001.

G. Amato, F. Falchi, and C. Gennaro. Geometric consistency checks for knn based image classification relying on local features. In *Proc. of the intl. workshop on Similarity Search and Applications (SISAP)*, pages 81–88. ACM, 2011.

A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Communications of the ACM*, 51(1):117–122, 2008.

X. Anguera, P. Obrador, T. Adamek, D. Marimon, and N. Oliver. Telefonica research content-based copy detection trecvid submission. In *Proc. of TRECVID*. NIST, 2009a.

X. Anguera, P. Obrador, and N. Oliver. Multimodal video copy detection applied to social media. In *Proc. of the 1st SIGMM workshop on Social media (WSM)*, pages 57–64. ACM, 2009b.

X. Anguera, T. Adamek, D. Xu, and J. M. Barrios. Telefonica research at trecvid 2011 content-based copy detection. In *Proc. of TRECVID*. NIST, 2011a.

X. Anguera, J. M. Barrios, T. Adamek, and N. Oliver. Multimodal fusion for video copy detection. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 1221–1224. ACM, 2011b.

S. Avila, N. Thome, M. Cord, E. Valle, and A. Araujo. Bossa: Extended bow formalism for image classification. In *Proc. of the int. conf. on Image Processing (ICIP)*, pages 2909–2912. IEEE, 2011.

M. Ayari, J. Delhumeau, M. Douze, H. Jégou, D. Potapov, J. Revaud, C. Schmid, and J. Yuan. Inria@trecvid'2011: Copy detection & multimedia event detection. In *Proc. of TRECVID*. NIST, 2011.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., 1999.

H. Bai, Y. Dong, W. Liu, L. Wang, C. Huang, and K. Tao. France telecom orange labs (beijing) at trecvid 2011: Content-based copy detection. In *Proc. of TRECVID*. NIST, 2011.

J. M. Barrios. Content-based video copy detection. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 1141–1142. ACM, 2009.

J. M. Barrios and B. Bustos. Text-based and content-based image retrieval on flickr: Demo. In *Proc. of the int. workshop on Similarity Search and Applications (SISAP)*, pages 156–157. IEEE, 2009.

J. M. Barrios and B. Bustos. Content-based video copy detection: Prisma at trecvid 2010. In *Proc. of TRECVID*. NIST, 2010.

J. M. Barrios and B. Bustos. P-VCD: A pivot-based approach for content-based video copy detection. In *Proc. of the IEEE int. conf. on Multimedia and Expo (ICME).*, pages 1–6. IEEE, 2011a.

J. M. Barrios and B. Bustos. Automatic weight selection for multi-metric distances. In *Proc. of the int. workshop on Similarity Search and Applications (SISAP)*, pages 61–68. ACM, 2011b.

J. M. Barrios and B. Bustos. Competitive content-based video copy detection using global descriptors. *Multimedia Tools and Applications*, 62(1):75–110, 2013.

J. M. Barrios, B. Bustos, and X. Anguera. Combining features at search time: Prisma at video copy detection task. In *Proc. of TRECVID*. NIST, 2011.

J. M. Barrios, B. Bustos, and T. Skopal. Snake table: A dynamic pivot table for streams of k-nn searches. In *Proc. of the int. workshop on Similarity Search and Applications (SISAP)*, pages 25–39. Springer, 2012.

J. M. Barrios, B. Bustos, and T. Skopal. Analyzing and dynamically indexing the query set. To appear in *Information Systems*, 2013.

A. Basharat, Y. Zhai, and M. Shah. Content based video matching using spatiotemporal volumes. *Journal of Computer Vision and Image Understanding*, 110(3):360—377, 2008.

M. Batko, P. Kohoutkova, and D. Novak. Cophir image collection under the microscope. In *Proc. of the intl. workshop on Similarity Search and Applications (SISAP)*, pages 47–54. IEEE, 2009.

H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

J. S. Beis and D. G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *IEEE conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1000–1006. IEEE, 1997.

D. N. Bhat and S. K. Nayar. Ordinal measures for image correspondence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):415–423, 1998.

C. Böhm, S. Berchtold, and D. A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, 33(3):322–373, 2001.

J. S. Boreczky and L. A. Rowe. Comparison of video shot boundary detection techniques. *Journal of Electronic Imaging*, 5(2):122–128, 1996.

Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Proc. of the intl. conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2559–2566. IEEE, 2010.

G. Bradski and A. Kaehler. *Learning OpenCV*. O'Reilly Media, Inc., 2008.

S. Brin. Near neighbor search in large metric spaces. In *Proc. of the int. conf. on Very Large Databases (VLDB).*, pages 574–584. Morgan Kauffman, 1995.

B. Bustos. Index structures for similarity search in multimedia databases. *Ph.D. thesis, Department of Computer and Information Science, University of Konstanz*, 2006.

B. Bustos and T. Skopal. Dynamic similarity search in multi-metric spaces. In *Proc. of the int. workshop on Multimedia Information Retrieval (MIR)*, pages 137–146. ACM, 2006.

B. Bustos, G. Navarro, and E. Chávez. Pivot selection techniques for proximity searching in metric spaces. *Pattern Recognition Letters*, 24(14):2357–2366, 2003.

B. Bustos, O. Pedreira, and N. Brisaboa. A dynamic pivot selection technique for similarity search. In *Proc. of the int. workshop on Similarity Search and Applications (SISAP)*, pages 105–112. IEEE, 2008.

J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, 2001.

L. Chen and X. Lian. Efficient similarity search in nonmetric spaces with local constant embedding. *IEEE Transactions on Knowledge and Data Engineering*, 20(3):321–336, 2008.

M. Cherubini, R. de Oliveira, and N. Oliver. Understanding near-duplicate videos: A user-centric approach. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 35–44. ACM, 2009.

S. Cheung and A. Zakhor. Efficient video similarity measurement with video signature. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(1):59–74, 2003.

O. Chum, J. Philbin, M. Isard, and A. Zisserman. Scalable near identical image and shot detection. In *Proc. of the int. conf. on Image and Video Retrieval (CIVR)*, pages 549–556. ACM, 2007.

P. Ciaccia and M. Patella. Searching in metric spaces with user-defined and approximate distances. *ACM Transactions on Database Systems*, 27(4):398–437, 2002.

P. Ciaccia, M. Patella, and P. Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *Proc. of the int. conf. on Very Large Databases (VLDB).*, pages 426–435. Morgan Kauffman, 1997.

Cisco Systems Inc. *Global IP Traffic Forecast and Methodology, 2006–2011*, jan 14, 2008. White paper.

Cisco Systems Inc. *Cisco Visual Networking Index: Forecast and Methodology, 2011–2016*, may 30, 2012. White paper.

B. Coskun, B. Sankur, and N. Memon. Spatio-temporal transform based video hashing. *IEEE Transactions on Multimedia*, 8(6):1190–1208, 2006.

T. Deselaers, T. Weyand, and H. Ney. Image retrieval and annotation using maximum entropy. In *CLEF Workshop 2006*, pages 725–734. Springer, 2007.

M. Douze, A. Gaidon, H. Jegou, M. Marszałek, and C. Schmid. Inria lear's video copy detection system. In *Proc. of TRECVID*. NIST, 2008.

S. Eickeler and S. Müller. Content-based video indexing of tv broadcast news using hidden markov models. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, pages 2997–3000 vol.6. IEEE, 1999.

C. Faloutsos and K.-I. Lin. Fastmap: a fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In *Proc. of the int. conf. on Management of data (SIGMOD)*, pages 163–174. ACM, 1995.

B. Fauvet, P. Bouthemy, P. Gros, and F. Spindler. A geometrical key-frame selection method exploiting dominant motion estimation in video. In *Proc. of the 3th int. conf. on Image and Video Retrieval (CIVR)*, pages 419—427. Springer, 2004.

J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, 1977.

N. Gengembre and S.-A. Berrani. A probabilistic framework for fusing frame-based searches within a video copy detection system. In *Proc. of the int. conf. on Image and Video Retrieval (CIVR)*, pages 211–220. ACM, 2008.

K.-S. Goh, B. Li, and E. Chang. DynDex: A dynamic and non-metric space indexer. In *Proc. of the 10th ACM int. conf. on Multimedia (ACMMM)*, pages 466–475. ACM, 2002.

R. C. Gonzalez and R. E. Woods. *Digital Image Processing (Third Edition)*. Prentice-Hall, Inc., 2007.

X. Guo, Y. Chen, W. Liu, Y. Mao, H. Zhang, K. Zhou, L. Wang, Y. Hua, Z. Zhao, Y. Zhao, and A. Cai. Bupt-mcprl at trecvid 2010. In *Proc. of TRECVID*. NIST, 2010.

V. Gupta, G. Boulianne, and P. Cardinal. Crim's content-based audio copy detection system for trecvid 2009. In *Proc. of the int. workshop on Content-Based Multimedia Indexing (CBMI)*, pages 1–6. IEEE, 2010.

V. Gupta, P. D. Z. Varcheie, L. Gagnon, and G. Boulianne. Crim at trecvid-2011: Content-based copy detection using nearest-neighbor mapping. In *Proc. of TRECVID*. NIST, 2011.

A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proc. of the int. conf. on Management of data (SIGMOD)*, pages 47–57. ACM, 1984.

J. Haitsma and T. Kalker. A highly robust audio fingerprinting system. In *Proc. of the int. symp. on Music Information Retrieval (ISMIR)*, pages 107–115. ISMIR, 2002.

A. Hampapur and R. Bolle. Comparison of distance measures for video copy detection. In *Proc. of the IEEE int. conf. on Multimedia and Expo (ICME)*, pages 737–740. IEEE, 2001.

A. Hampapur, T. Weymouth, and R. Jain. Digital video segmentation. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 357–364. ACM, 1994.

A. Hanjalic. Shot-boundary detection: unraveled and resolved? *IEEE Transactions on Circuits and Systems for Video Technology*, 12(2):90–105, 2002.

C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of the Alvey Vision Conference*, pages 147—-151. The Plessey Company, 1988.

G. Hjaltason and H. Samet. Ranking in spatial databases. In *Proc. of the 4th int. symp. on Spatial Databases (SSD)*, pages 83–95. Springer, 1995.

P. Howarth and S. Rüger. Fractional distance measures for content-based image retrieval. In *Proc. of the 27th european conf. on IR Research (ECIR)*, pages 447–456. Springer, 2005.

M. Hradiš, I. Řezníček, D. Bařina, A. Vlček, and P. Zemčík. Brno university of technology at trecvid 2010 sin, ccd. In *Proc. of TRECVID*. NIST, 2010.

M. Hradiš, I. Řezníček, K. Behúň, and L. Otrusina. Brno university of technology at trecvid 2011 sin, ccd. In *Proc. of TRECVID*. NIST, 2011.

K. Iwamoto, E. Kasutani, and A. Yamada. Image signature robust to caption superimposition for video sequence identification. In *Proc. of the int. conf. on Image Processing (ICIP)*, pages 3185–3188. IEEE, 2006.

H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proc. of the european conf. on Computer Vision (ECCV)*, pages 304–317. Springer, 2008.

H. Jégou, M. Douze, and C. Schmid. Packing bag-of-features. In *Proc. of the IEEE int. conf. on Computer Vision (ICCV)*, pages 2357–2364. IEEE, 2009.

H. Jégou, M. Douze, G. Gravier, C. Schmid, and P. Gros. Inria lear-texmex: Video copy detection task. In *Proc. of TRECVID*. NIST, 2010.

M. Jiang, S. Fang, Y. Tian, T. Huang, and W. Gao. Pku-idm @ trecvid 2011 cbcd: Content-based copy detection with cascade of multimodal features and temporal pyramid matching. In *Proc. of TRECVID*. NIST, 2011.

W. Jiang, C. Cotton, S.-F. Chang, D. Ellis, and A. C. Loui. Short-term audio-visual atoms for generic video concept classification. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 5–14. ACM, 2009.

A. Joly, C. Frélicot, and O. Buisson. Robust content-based video copy identification in a large reference database. In *Proc. of the int. conf. on Image and Video Retrieval (CIVR)*, pages 414–424. Springer, 2003.

A. Joly, O. Buisson, and C. Frélicot. Content-based copy retrieval using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, 9(2):293–306, 2007.

A. Joly, J. Law-To, and N. Boujemaa. Inria-imedia trecvid 2008: Video copy detection. In *Proc. of TRECVID*. NIST, 2008.

Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Proc. of the intl. conf. on Computer Vision and Pattern Recognition (CVPR)*, pages II–506 – II–513 Vol.2. IEEE, 2004.

C. Kim. Content-based image copy detection. *Signal Processing: Image Communication*, 18(3): 169–184, 2003.

C. Kim and B. Vasudev. Spatiotemporal sequence matching for efficient video copy detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(1):127–132, 2005.

S. Kutluk and B. Gunsel. Itu mspr trecvid 2010 video copy detection system. In *Proc. of TRECVID*. NIST, 2010.

G. Langelaar, I. Setyawan, and R. Lagendijk. Watermarking digital image and video data. a state-of-the-art overview. *Signal Processing Magazine*, 17(5):20–46, 2000.

J. Law-To, O. Buisson, V. Gouet-Brunet, and N. Boujemaa. Robust voting algorithm based on labels of behavior for video copy detection. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 835–844. ACM, 2006.

J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In *Proc. of the int. conf. on Image and Video Retrieval (CIVR)*, pages 371–378. ACM, 2007a.

J. Law-To, A. Joly, and N. Boujemaa. MUSCLE-VCD-2007: A live benchmark for video copy detection, 2007b. `http://www-rocq.inria.fr/imedia/civr-bench/`.

S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. of the intl. conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178 Vol.2. IEEE, 2006.

D.-D. Le, S. Poullot, M. Crucianu, X. Wu, M. Nett, M. E. Houle, and S. Satoh. National institute of informatics, japan at trecvid 2009. In *Proc. of TRECVID*. NIST, 2009.

D.-D. Le, S. Poullot, X. Wu, B. Nouvel, and S. Satoh. National institute of informatics, japan at trecvid 2010. In *Proc. of TRECVID*. NIST, 2010.

D.-D. Le, C.-Z. Zhu, S. Poullot, V. Q. Lam, D. A. Duong, and S. Satoh. National institute of informatics, japan at trecvid 2011. In *Proc. of TRECVID*. NIST, USA, 2011.

S. Lee and C. D. Yoo. Robust video fingerprinting for content-based video identification. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(7):983–988, 2008.

M. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications and Applications*, 2(1):1–19, 2006.

B. Li, E. Chang, and C.-T. Wu. DPF - a perceptual distance function for image retrieval. In *Proc. of the 2002 int. conf. on Image Processing (ICIP)*, pages II–597–II–600. IEEE, 2002.

Y. Li, L. Mou, M. Jiang, C. Su, X. Fang, M. Qian, Y. Tian, Y. Wang, T. Huang, and W. Gao. Pku-idm @ trecvid 2010: Copy detection with visual-audio feature fusion and sequential pyramid matching. In *Proc. of TRECVID*. NIST, 2010.

Y. Liang, B. Cao, J. Li, C. Zhu, Y. Zhang, C. Tan, G. Chen, C. Sun, J. Yuan, M. Xu, and B. Zhang. Thu-img at trecvid 2009. In *Proc. of TRECVID*. NIST, 2009.

Z. Liu, E. Zavesky, D. Gibbon, B. Shahraray, and P. Haffner. AT&T research at trecvid 2007. In *Proc. of TRECVID*. NIST, 2007.

Z. Liu, T. Liu, D. Gibbon, and B. Shahraray. Effective and scalable video copy detection. In *Proc. of the int. conf. on Multimedia Information Retrieval (MIR)*, pages 119–128. ACM, 2010a.

Z. Liu, E. Zavesky, N. Sawant, and B. Shahraray. At&t research at trecvid 2010. In *Proc. of TRECVID*. NIST, 2010b.

Z. Liu, E. Zavesky, N. Zhou, and B. Shahraray. At&t research at trecvid 2011. In *Proc. of TRECVID*. NIST, 2011.

D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

D. Lowe. SIFT demo program (version 4), 2005. `http://www.cs.ubc.ca/~lowe/keypoints/siftDemoV4.zip`.

B. S. Manjunath, J.-R. Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6):703–715, 2001.

Y. Meng, E. Chang, and B. Li. Enhancing DPF for near-replica image recognition. In *IEEE conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 416–423. IEEE, 2003.

L. Micó and J. Oncina. A constant average time algorithm to allow insertions in the LAESA fast nearest neighbour search index. In *Proc. of the int. conf. on Pattern Recognition (ICPR)*, pages 3911–3914. IEEE, 2010.

M. Micó, J. Oncina, and E. Vidal. A new version of the nearest-neighbour approximating and eliminating search algorithm (AESA) with linear preprocessing time and memory requirements. *Pattern Recognition Letters*, 15(1):9–17, 1994.

H.-S. Min, W. D. Neve, and Y. M. Ro. Towards using semantic features for near-duplicate video detection. In *Proc. of the IEEE int. conf. on Multimedia and Expo (ICME)*, pages 1364–1369. IEEE, 2010.

H.-S. Min, J. Y. Choi, W. D. Neve, and Y. M. Ro. Bimodal fusion of low-level visual features and high-level semantic features for near-duplicate video clip detection. *Signal Processing: Image Communication*, 26(10):612–627, 2011.

M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *Proc. of the int. conf. on Computer Vision Theory and Application (VISSAPP)*, pages 331–340. INSTICC Press, 2009.

R. Mukai, T. Kurozumi, K. Hiramatsu, T. Kawanishi, H. Nagano, and K. Kashino. Ntt communication science laboratories at trecvid 2010 content-based copy detection. In *Proc. of TRECVID*. NIST, 2010a.

R. Mukai, T. Kurozumi, K. Hiramatsu, T. Kawanishi, H. Nagano, and K. Kashino. Ntt communication science laboratories at trecvid 2010 content-based copy detection. In *Proc. of TRECVID*. NIST, 2010b.

R. Mukai, T. Kurozumi, T. Kawanishi, H. Nagano, and K. Kashino. Ntt communication science laboratories at trecvid 2011 content-based copy detection. In *Proc. of TRECVID*. NIST, 2011.

M. Naito, K. Matsumoto, M. Shshibori, K. Kita, M. Cuturi, T. Matsui, S. Sato, K. Hoashi, F. Sugaya, , and Y. Nakajima. Shot boundary detection and high-level feature extraction experiments for trecvid 2006. In *Proc. of TRECVID*. NIST, 2006.

A. Natsev, J. R. Smith, M. Hill, G. Hua, B. Huangy, M. Merlery, L. Xie, H. Ouyangz, and M. Zhoux. Ibm research trecvid-2010 video copy detection and multimedia event detection system. In *Proc. of TRECVID*. NIST, 2010.

X. Naturel and P. Gros. A fast shot matching strategy for detecting duplicate sequences in a television stream. In *Proc. of the int. workshop on Computer Vision meets Databases (CVDB)*, pages 21–27. ACM, 2005.

R. Ng and J. Han. CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering*, 14(5):1003–1016, 2002.

C.-W. Ngo, S.-A. Zhu, H.-K. Tan, W.-L. Zhao, and X.-Y. Wei. Vireo at trecvid 2010: Semantic indexing, known-item search, and content-based copy detection. In *Proc. of TRECVID*. NIST, 2010.

M. Patella and P. Ciaccia. Approximate similarity search: A multi-faceted problem. *Journal of Discrete Algorithms*, 7(1):36–48, 2009.

S. Poullot, O. Buisson, and M. Crucianu. Z-grid-based probabilistic retrieval for scaling up content-based copy detection. In *Proc. of the int. conf. on Image and Video Retrieval (CIVR)*, pages 348–355. ACM, 2007.

S. Poullot, M. Crucianu, and O. Buisson. Scalable mining of large video databases using copy detection. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 61–70. ACM, 2008.

S. Poullot, O. Buisson, and M. Crucianu. Scaling content-based video copy detection to very large databases. *Multimedia Tools and Applications*, 47(2):279–306, 2010.

G. Roth, R. Laganière, M. Bouchard, I. Lakhmiri, and T. Janati. Viva lab - university of ottawa at trecvid 2009 content based copy detection. In *Proc. of TRECVID*. NIST, 2009.

V. Roth, J. Laub, J. Buhmann, and K.-R. Müller. Going metric: Denoising pairwise data. In *Proc. of the int. conf. on Neural Information Processing Systems (NIPS)*, pages 817–824. MIT Press, 2002.

A. H. Rouhi and J. A. Thom. Rmit university at trecvid 2011 content-based copy detection. In *Proc. of TRECVID*. NIST, 2011.

Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

Y. Rubner, J. Puzicha, C. Tomasi, and J. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, 84(1):25–43, 2001.

T. Sakata, N. Matozaki, K. Kise, and M. Iwamura. Osaka prefecture university at trecvid 2011. In *Proc. of TRECVID*. NIST, 2011.

A. Saracoğlu, E. Esen, T. K. Ateş, B. O. Acar, Ü. Zubari, E. C. Ozan, E. Özalp, A. A. Alatan, and T. Çiloğlu. Content based copy detection with coarse audio-visual fingerprints. In *Proc. of the int. workshop on Content-Based Multimedia Indexing (CBMI)*, pages 213–218. IEEE, 2009.

A. Saracoğlu, E. Esen, M. Soysal, T. K. Ateş, B. Loğoğlu, M. Tekin, T. Karadeniz, M. Sevinç, H. Sevimli, B. O. Acar, E. C. Ozan, D. O. Onur, S. Selçuk, A. A. Alatan, and T. Çiloğlu. Tübitak uzay at trecvid 2010: Content-based copy detection and semantic indexing. In *Proc. of TRECVID*. NIST, 2010.

S. Satoh, M. Takimoto, and J. Adachi. Scene duplicate detection from videos based on trajectories of feature points. In *Proc. of the int. conf. on Multimedia Information Retrieval (MIR)*, pages 237–244. ACM, 2007.

J. Shao, H. T. Shen, and X. Zhou. Challenges and techniques for effective and efficient similarity search in large video databases. *Proc. of the VLDB Endowment*, 1(2):1598–1603, 2008.

E. Shechtman and M. Irani. Space-time behavior based correlation. In *Proc. of the intl. conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 405–412 Vol.1. IEEE, 2005.

H. T. Shen, X. Zhou, Z. Huang, J. Shao, and X. Zhou. Uqlips: A real-time near-duplicate video clip detection system. In *Proc. of the int. conf. on Very Large Data Bases (VLDB)*, pages 1374–1377. VLDB Endowment, 2007.

H. T. Shen, J. Shao, Z. Huang, Y. Yang, J. Song, J. Liu, and X. Zhu. Uqmsg experiments for trecvid 2011. In *Proc. of TRECVID*. NIST, 2011.

J. Shi and C. Tomasi. Good features to track. In *Proc. of the intl. conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600. IEEE, 1994.

M. Shishibori, M. Ohnishi, Y. Tanioka, and K. Kita. Instance search and content-based copy detection experiments for trecvid 2011. In *Proc. of TRECVID*. NIST, 2011.

J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. of the IEEE int. conf. on Computer Vision (ICCV)*, pages 1470–1477. IEEE, 2003.

T. Skopal. On fast non-metric similarity search by metric access methods. In *Proc. of the 10th int. conf. on Extending Database Technology (EDBT)*, pages 718–736. Springer, 2006.

T. Skopal. Unified framework for fast exact and approximate search in dissimilarity spaces. *ACM Transactions on Database Systems*, 32(4):29–47, 2007.

T. Skopal and B. Bustos. On nonmetric similarity search problems in complex domains. *ACM Computing Surveys*, 43(4):1–34, 2011.

T. Skopal and J. Lokoč. NM-tree: Flexible approximate similarity search in metric and non-metric spaces. In *Proc. of the 19th int. workshop on Database and Expert Systems Applications (DEXA)*, pages 312–325. Springer, 2008.

T. Skopal, J. Lokoč, and B. Bustos. D-cache: Universal distance cache for metric access methods. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):868–881, 2012.

A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and TRECVid. In *Proc. of the int. workshop on Multimedia Information Retrieval (MIR)*, pages 321–330. ACM, 2006.

A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

C. G. Snoek, M. Worring, and A. W. Smeulders. Early versus late fusion in semantic video analysis. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 399–402. ACM, 2005.

C. Sun, J. Li, B. Zhang, and Q. Zhang. Thu-img at trecvid 2010. In *Proc. of TRECVID*. NIST, 2010.

M. Swanson, M. Kobayashi, and A. Tewfik. Multimedia data-embedding and watermarking technologies. *Proc. of the IEEE*, 86(6):1064–1087, 1998.

R. Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.

H.-K. Tan, C.-W. Ngo, R. Hong, and T.-S. Chua. Scalable detection of partial near-duplicate videos by visual-temporal consistency. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 145–154. ACM, 2009.

C. Tomasi and T. Kanade. Detection and tracking of point features. *Carnegie Mellon University Technical Report CMU-CS-91-132*, 1991.

Y. Uchida, S. Sakazawa, M. Agrawal, and M. Akbacak. Kddi labs and sri international at trecvid 2010: Content-based copy detection. In *Proc. of TRECVID*. NIST, 2010.

Y. Uchida, K. Takagi, and S. Sakazawa. Kddi labs at trecvid 2011: Content-based copy detection. In *Proc. of TRECVID*. NIST, 2011.

K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. Evaluation of color descriptors for object and scene recognition. In *Proc. of the intl. conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

J. van Gemert, J.-M. Geusebroek, C. Veenman, and A. Smeulders. Kernel codebooks for scene categorization. In *Proc. of the european conf. on Computer Vision (ECCV)*, pages 696–709. Springer, 2008.

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms (version 0.9.14), 2008. `http://www.vlfeat.org/`.

E. Vidal. New formulation and improvements of the nearest-neighbour approximating and eliminating search algorithm (AESA). *Pattern Recognition Letters*, 15(1):1–7, 1994.

G. Willems, T. Tuytelaars, and L. V. Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In *Proc. of the european conf. on Computer Vision (ECCV)*, pages 650––663. Springer, 2008a.

G. Willems, T. Tuytelaars, and L. V. Gool. Spatio-temporal features for robust content-based video copy detection. In *Proc. of the int. conf. on Multimedia Information Retrieval (MIR)*, pages 283–290. ACM, 2008b.

R. B. Wolfgang, C. I. Podilchuk, and E. J. Delp. Perceptual watermarks for digital images and video. *Proc. of the IEEE*, 87(7):1108–1126, 1999.

X. Wu, A. G. Hauptmann, and C.-W. Ngo. Practical elimination of near-duplicates from web video search. In *Proc. of the int. conf. on Multimedia (ACMMM)*, pages 218—227. ACM, 2007.

M.-C. Yeh, C.-Y. Hsu, and C.-S. Lu. Ntnu-academia sinica at trecvid 2010 content based copy detection. In *Proc. of TRECVID*. NIST, 2010.

E. Younessian, X. Anguera, T. Adamek, N. Oliver, and D. Marimon. Telefonica research at trecvid 2010 content-based copy detection. In *Proc. of TRECVID*. NIST, 2010.

J. Yuan, Z. Guo, L. Lv, W. Wan, T. Zhang, D. Wang, X. Liu, C. Liu, S. Zhu, D. Wang, Y. Pang, N. Ding, Y. Liu, J. Wang, X. Zhang, X. Tie, Z. Wang, H. Wang, T. Xiao, Y. Liang, J. Li, F. Lin, B. Zhang, J. Li, W. Wu, X. Tong, D. Ding, Y. Chen, T. Wang, and Y. Zhang. THU and ICRC at trecvid 2007. In *Proc. of TRECVID*. NIST, 2007.

P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach (Advances in Database Systems)*. Springer, 2005.

W.-L. Zhao, D. Borth, and T. M. Breuel. Participation at trecvid 2011 semantic indexing & content-based copy detection tasks. In *Proc. of TRECVID*. NIST, 2011a.

Z. Zhao, Y. Zhao, X. Guo, Y. Chen, Y. Hua, W. Wang, C. Liu, S. Wu, H. Zhang, L. Wang, Y. Mao, A. Cai, and M. Zhai. Bupt-mcprl at trecvid 2011*. In *Proc. of TRECVID*. NIST, 2011b.

Y. Zhuang, Y. Rui, T. Huang, and S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. In *Proc. of the int. conf. on Image Processing (ICIP)*, pages 866–870. IEEE, 1998.

Y. Zhuang, Y. Rui, and T. Huang. Video key frame extraction by unsupervised clustering and feedback adjustment. *Journal of Computer Science and Technology*, 14(3):283–287, 1999.

Appendix A

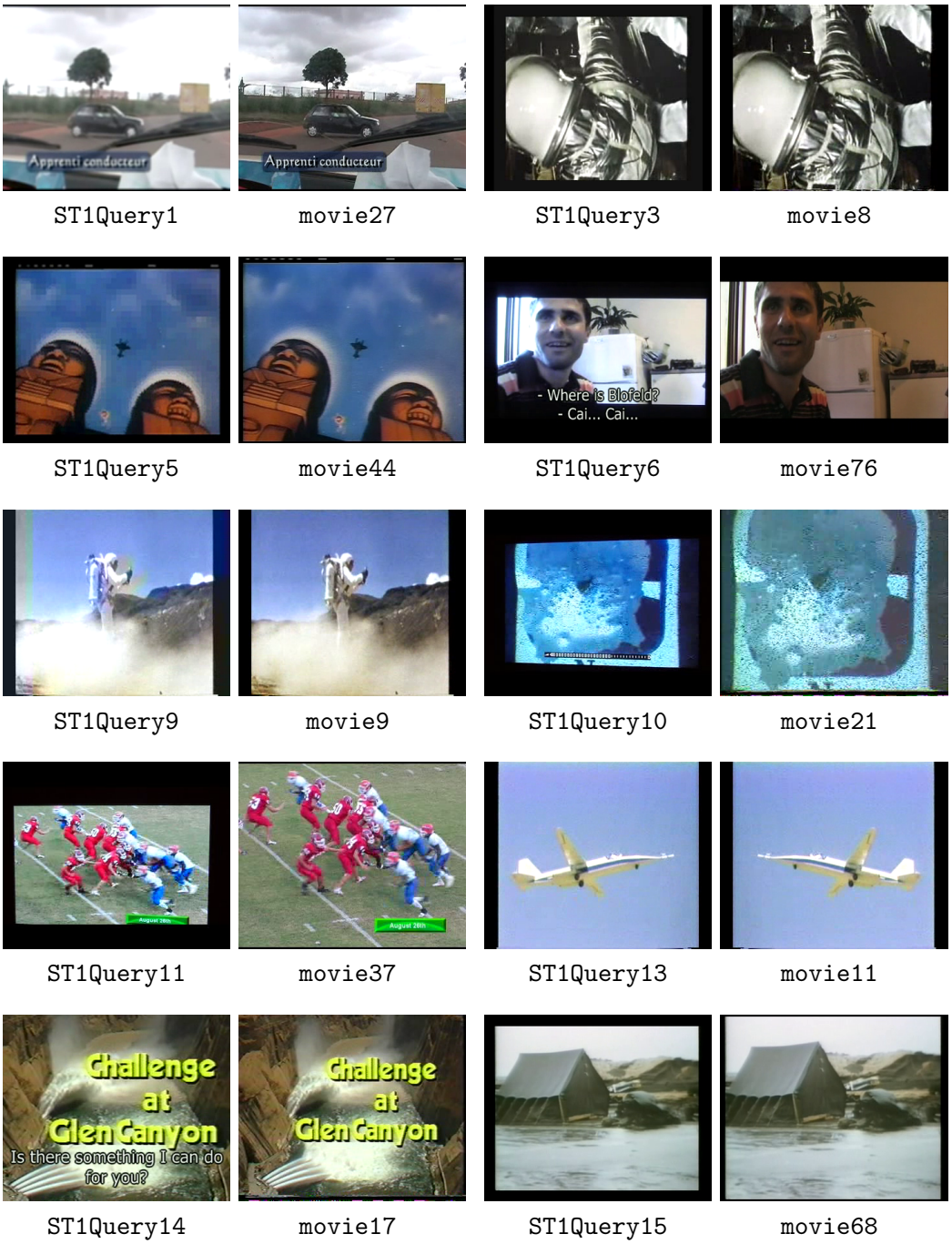
MUSCLE-VCD-2007 ground-truth

Query		Reference		Copy Length	Transformations
Video	Copy Start	Video	Copy Start		
ST1Query1	00:00.0	movie27	00:00.0	06:58.3	Color adjustment, blur.
ST1Query2	—	—	—	—	—
ST1Query3	00:00.0	movie8	00:00.0	06:19.1	Reencoding, color adjustment, crop.
ST1Query4	—	—	—	—	—
ST1Query5	00:00.0	movie44	00:01.4	07:43.8	Strong reencoding, resize.
ST1Query6	00:00.0	movie76	00:09.3	06:02.1	Frontal camcording, subtitles, live audio recording.
ST1Query7	—	—	—	—	—
ST1Query8	—	—	—	—	—
ST1Query9	00:00.0	movie9	00:00.0	09:13.4	Colors adjustment.
ST1Query10	00:00.0	movie21	00:00.7	11:34.0	Non frontal camcording, live audio recording.
ST1Query11	00:00.0	movie37	00:21.2	11:46.9	Frontal camcording, live audio recording.
ST1Query12	—	—	—	—	—
ST1Query13	00:00.0	movie11	00:00.0	17:27.3	Flip.
ST1Query14	00:00.0	movie17	00:02.4	26:19.4	Zoom, subtitles.
ST1Query15	00:00.0	movie68	00:05.4	42:59.4	Resize.

Table A.1 – Ground-truth of ST1 collection in MUSCLE-VCD-2007 dataset. Start times and copy length in format [minutes]:[seconds].[d]

Query		Reference		Copy Length	Transformations
Video	Copy Start	Video	Copy Start		
ST2Query1	01:02.6	movie30	05:51.0	00:38.4	Blur, lack of audio track.
ST2Query1	02:52.6	movie55	19:12.0	01:14.6	Blur.
ST2Query1	07:51.3	movie33	01:31.0	00:52.8	Blur.
ST2Query1	09:11.6	movie38	15:06.9	01:49.3	Blur.
ST2Query1	11:22.5	movie43	38:39.6	01:25.2	Blur.
ST2Query1	13:30.7	movie50	01:23.0	00:40.8	Blur.
ST2Query2	00:40.2	movie98	08:04.0	01:25.1	Insertion of moving caption.
ST2Query2	03:30.6	movie20	03:35.0	00:39.6	Change of color, sharpness.
ST2Query2	04:30.7	movie27	01:43.0	01:04.0	Vertical deformation, letter box, contrast.
ST2Query2	06:21.4	movie26	15:07.0	02:02.7	Insertion of logo, vertical shift, contrast.
ST2Query2	08:40.6	movie89	08:06.0	00:36.5	Blur.
ST2Query2	10:36.4	movie82	06:30.0	00:56.1	Insertion of logo, contrast.
ST2Query2	12:41.0	movie59	13:39.0	01:43.2	Zoom, vertical deformation.
ST2Query2	16:28.2	movie13	04:51.0	00:40.9	Crop, change of color.
ST2Query3	01:10.6	movie46	31:36.0	00:55.2	Blur, flip.
ST2Query3	03:27.6	movie15	05:45.0	00:38.9	Blur, vertical deformation.
ST2Query3	04:56.4	movie16	40:38.0	00:22.4	Crop, insertion of logo, blur.
ST2Query3	06:18.5	movie18	00:55.0	00:30.4	Change of gamma.
ST2Query3	07:56.1	movie99	48:27.0	00:27.8	Noise, brightness.
ST2Query3	10:02.2	movie65	07:29.0	00:44.2	Brightness, change of gamma, vertical shift, subtitles.
ST2Query3	11:22.9	movie23	04:25.0	00:29.3	Crop, Change of color, vertical deformation.

Table A.2 – Ground-truth of ST2 collection in MUSCLE-VCD-2007 dataset. Start times and copy length in format [minutes]:[seconds].[d]



ST1Query1 movie27 ST1Query3 movie8

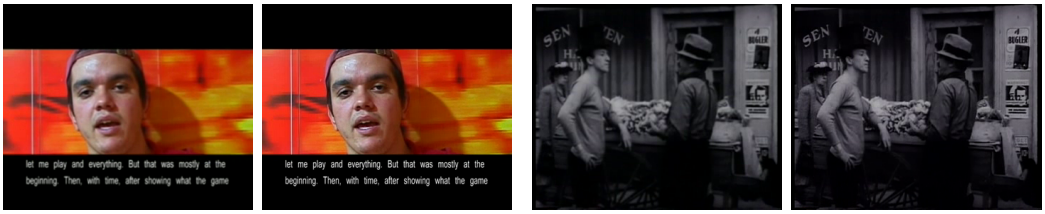
ST1Query5 movie44 ST1Query6 movie76

ST1Query9 movie9 ST1Query10 movie21

ST1Query11 movie37 ST1Query13 movie11

ST1Query14 movie17 ST1Query15 movie68

Figure A.1 – Copies in ST1 collection. On the left a frame from query video and on the right the corresponding frame from reference video.



ST2Query1 movie30 ST2Query1 movie55

ST2Query1 movie33 ST2Query1 movie38

ST2Query1 movie43 ST2Query1 movie50

Figure A.2 – Copy excerpts in `ST2Query1`. On the left a sample from query video and on the right the corresponding frame from reference video.

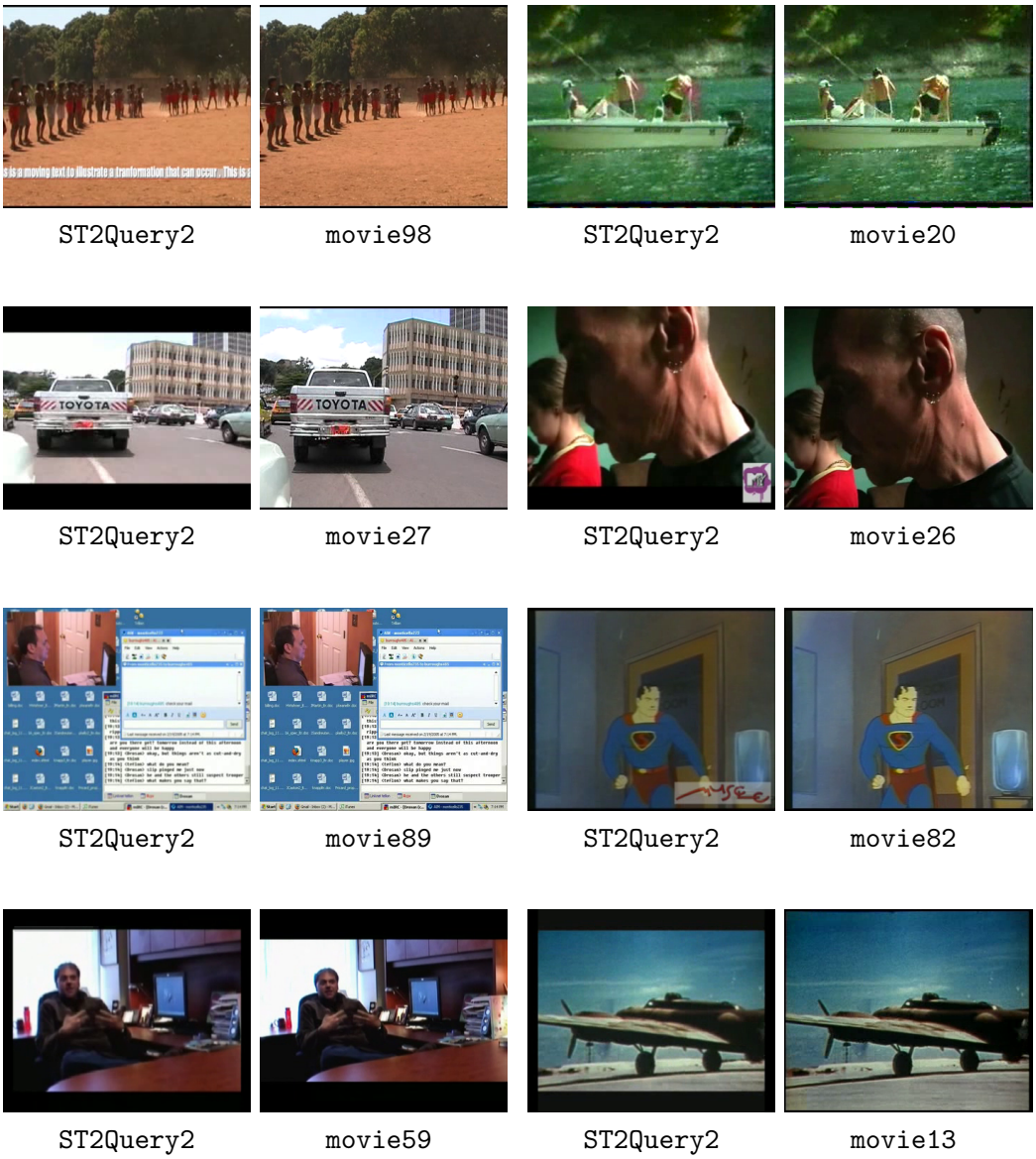


Figure A.3 – Copy excerpts in `ST2Query2`. On the left a sample from query video and on the right the corresponding frame from reference video.

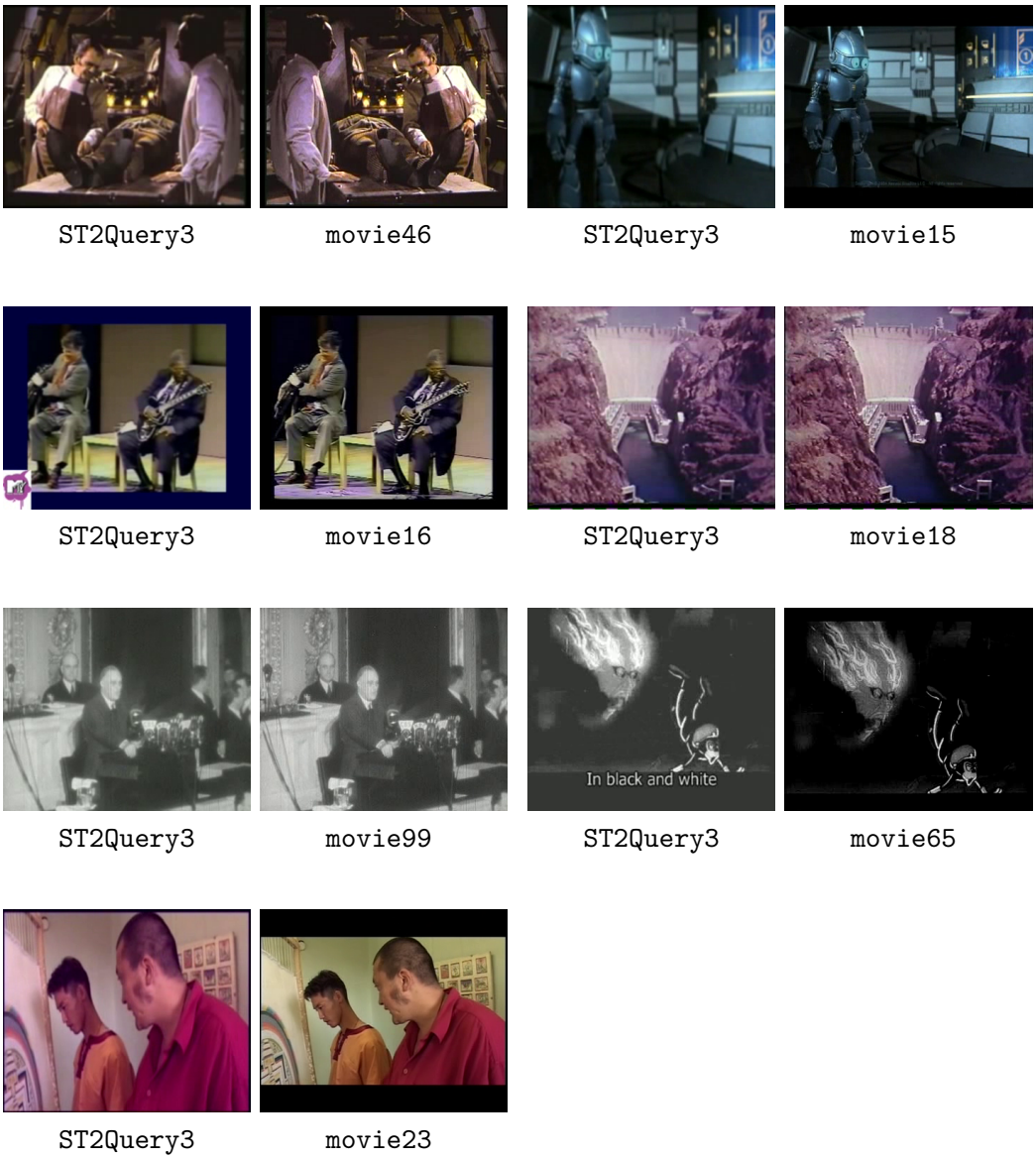


Figure A.4 – Copy excerpts in `ST2Query3`. On the left a sample from query video and on the right the corresponding frame from reference video.

Appendix B

Results at TRECVID 2010

Team 2010	Organization	Location	Paper
asahikasei	Asahikasei Co.	Asia	—
ATTLabs	AT&T Labs - Research	NorthAm	[Liu et al., 2010b]
brno	Brno University of Technology	Europe	[Hradiš et al., 2010]
BUPT-MCPRL	Beijing University of Posts and Telecommunications-MCPRL	Asia	[Guo et al., 2010]
CCU	National Chung Cheng University	Asia	—
IBM	IBM T. J. Watson Research Center	NorthAm	[Natsev et al., 2010]
IDARE	Shandong University	Asia	—
INRIA-LEAR-TEXMEX	INRIA-TEXMEX	Europe	[Jégou et al., 2010]
ITU_MSPR	Istanbul Technical University	Europe	[Kutluk and Gunsel, 2010]
KDDILabs-SRI	KDDI R&D Labs and SRI International	Asia	[Uchida et al., 2010]
NII	National Institute of Informatics	Asia	[Le et al., 2010]
NJU	Nanjing University	Asia	[Mukai et al., 2010a]
NTNU-Academia-Sinica	NTNU and Academia Sinica	Asia	[Yeh et al., 2010]
NTT-CSL	NTT Communication Science Laboratories-CSL	Asia	[Mukai et al., 2010b]
PKU-IDM	Peking University-IDM	Asia	[Li et al., 2010]
PRISMA	**University of Chile**	**SouthAm**	**[Barrios and Bustos, 2010]**
SYSU-GITL	Sun Yat-sen University - GITL	Asia	—
THU-IMG	Tsinghua University-IMG	Asia	[Sun et al., 2010]
TID	Telefonica Research	Europe	[Younessian et al., 2010]
TUBITAK_UZAY	TUBITAK - Space Technologies Research Institute	Europe	[Saracoğlu et al., 2010]
UNIBS	University of Brescia	Europe	—
VIREO	City University of Hong Kong	Asia	[Ngo et al., 2010]

Table B.1 – The 22 participant teams in CCD evaluation at TRECVID 2010.

#	Team	Run	Type	Avg. Optimal		Avg. Actual		Avg.
				NDCR	F1	NDCR	F1	MTP
1	PKU-IDM	perseus	AV	0.061	0.889	0.061	0.889	11925
2	PKU-IDM	kraken	AV	0.091	0.892	0.091	0.892	9631
3	NTT-CSL	3	AV	0.140	0.933	0.140	0.933	25
4	NTT-CSL	0	AV	0.158	0.934	0.158	0.934	25
5	INRIA-LEAR-TEXMEX	mouflon	AV	0.308	0.959	1017.305	0.957	537
6	KDDILabs-SRI	1	AV	0.343	0.961	13.982	0.847	84
7	KDDILabs-SRI	2	AV	0.373	0.964	13.986	0.848	84
8	INRIA-LEAR-TEXMEX	bouquetin	AV	0.465	0.946	1485.700	0.935	537
9	ATTLabs	1	AV	0.554	0.808	48.199	0.809	41
10	**PRISMA**	**ehdNghT10**	**V**	**0.611**	**0.828**	**40.753**	**0.846**	**128**
11	**PRISMA**	**ehdNgryhst**	**V**	**0.611**	**0.828**	**147.772**	**0.811**	**128**
12	IDARE	test	—	1.000	0.000	1.000	0.000	1592
13	ATTLabs	3	AV	10.050	0.822	105.473	0.799	41
14	IBM	gistG	V	13.986	0.775	80.890	0.773	14
15	IBM	gistGCsift	V	14.190	0.819	228.262	0.701	85
16	IBM	gistGC	V	14.211	0.818	469.092	0.801	50
17	nii	av	AV	17.915	0.927	306.224	0.898	10
18	asahikasei	VmainAsub	AV	18.166	0.758	274.309	0.761	225
19	brno	l3sl2	AV	27.747	0.452	880.447	0.565	1003
20	brno	l3sl	AV	29.663	0.468	553.523	0.536	640
21	nii	a	A	31.337	0.928	306.224	0.898	9
22	VIREO	srpeflip	V	40.974	0.444	2650.472	0.612	144
23	NJU	norank1	V	41.135	0.432	9503.509	0.469	94
24	THU-IMG	tortoise	V	54.495	0.596	22619	0.633	110
25	THU-IMG	tiger	V	54.503	0.599	15780	0.634	110
26	asahikasei	AmainVsub	AV	54.511	0.512	1429.169	0.656	407
27	TUBITAK_UZAY	aindexnf	A	61.969	0.384	658.254	0.917	4
28	NTNU-Academia-Sinica	2	V	67.885	0.505	1111.631	0.709	111
29	NJU	comp2	V	81.300	0.084	0.998	0.064	94
30	SYSU-GITL	sysuc2	V	108.07	0.000	13987	0.000	504
31	CCU	submission	AV	111.89	0.011	2412.019	0.027	81
32	TID	rawfusion	AV	143.78	0.921	210.709	0.920	601
33	ITU_MSPR	ITUMSPR2	AV	228.05	0.563	228.046	0.563	880
34	BUPT-MCPRL	SD	AV	535.41	0.868	535.411	0.868	27
35	BUPT-MCPRL	TF	AV	669.25	0.868	669.248	0.868	27
36	UNIBS	MF	V	25176	0.524	36057	0.520	12
37	UNIBS	SF	V	25390	0.774	36953	0.758	11

Table B.2 – Results for NoFA profile at TRECVID 2010. Values averaged for the 56 transformations. Submissions in descending order by Average Optimal NDCR.

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.977	15	0.631	10	0.269	10	0.262	10	0.769	13	0.708	15	0.562	8	0.708	12	0.611	10
A2	0.977	13	0.631	8	0.269	9	0.262	9	0.769	15	0.708	13	0.562	8	0.708	12	0.611	11
A3	0.977	12	0.631	7	0.269	8	0.262	8	0.769	13	0.708	12	0.562	7	0.708	11	0.611	8
A4	0.977	11	0.631	9	0.269	9	0.262	8	0.769	15	0.708	14	0.562	9	0.708	11	0.611	10
A5	0.977	16	0.631	14	0.269	11	0.262	10	0.769	17	0.708	18	0.562	11	0.708	15	0.611	13
A6	0.977	16	0.631	11	0.269	9	0.262	9	0.769	16	0.708	18	0.562	9	0.708	15	0.611	12
A7	0.977	17	0.631	13	0.269	9	0.262	9	0.769	17	0.708	18	0.562	11	0.708	15	0.611	12
Avg	0.977	12	0.631	9	0.269	8	0.262	8	0.769	15	0.708	15	0.562	8	0.708	12	**0.611**	**10**

(a) Optimal NDCR for `nofa.ehdNghT10`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.484	23	0.946	10	0.878	21	0.877	16	0.753	25	0.838	22	0.931	14	0.916	15	0.828	16
A2	0.484	23	0.946	9	0.878	23	0.877	15	0.753	25	0.838	22	0.931	13	0.916	15	0.828	15
A3	0.484	22	0.946	9	0.878	21	0.877	15	0.753	24	0.838	21	0.931	13	0.916	15	0.828	15
A4	0.484	21	0.946	9	0.878	20	0.877	15	0.753	25	0.838	20	0.931	14	0.916	15	0.828	14
A5	0.484	24	0.946	6	0.878	20	0.877	17	0.753	26	0.838	22	0.931	12	0.916	14	0.828	17
A6	0.484	23	0.946	5	0.878	21	0.877	15	0.753	26	0.838	21	0.931	11	0.916	14	0.828	15
A7	0.484	22	0.946	5	0.878	21	0.877	15	0.753	25	0.838	20	0.931	9	0.916	11	0.828	15
Avg	0.484	21	0.946	5	0.878	19	0.877	14	0.753	24	0.838	20	0.931	12	0.916	15	**0.828**	**14**

(b) Optimal F1 for `nofa.ehdNghT10`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.977	15	0.631	10	0.269	10	0.262	10	0.769	13	0.708	15	0.562	8	0.708	12	0.611	10
A2	0.977	13	0.631	8	0.269	9	0.262	9	0.769	15	0.708	13	0.562	8	0.708	12	0.611	11
A3	0.977	12	0.631	7	0.269	8	0.262	8	0.769	13	0.708	12	0.562	7	0.708	11	0.611	8
A4	0.977	11	0.631	9	0.269	9	0.262	8	0.769	15	0.708	14	0.562	9	0.708	11	0.611	10
A5	0.977	16	0.631	14	0.269	11	0.262	10	0.769	17	0.708	18	0.562	11	0.708	15	0.611	13
A6	0.977	16	0.631	11	0.269	9	0.262	9	0.769	16	0.708	18	0.562	9	0.708	15	0.611	12
A7	0.977	17	0.631	13	0.269	9	0.262	9	0.769	17	0.708	18	0.562	11	0.708	15	0.611	12
Avg	0.977	12	0.631	9	0.269	8	0.262	8	0.769	15	0.708	15	0.562	8	0.708	12	**0.611**	**10**

(c) Optimal NDCR for `nofa.ehdNgryhst`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.484	23	0.946	10	0.878	21	0.877	16	0.753	25	0.838	22	0.931	14	0.916	15	0.828	16
A2	0.484	23	0.946	9	0.878	23	0.877	15	0.753	25	0.838	22	0.931	13	0.916	15	0.828	15
A3	0.484	22	0.946	9	0.878	21	0.877	15	0.753	24	0.838	21	0.931	13	0.916	15	0.828	15
A4	0.484	21	0.946	9	0.878	20	0.877	15	0.753	25	0.838	20	0.931	14	0.916	15	0.828	14
A5	0.484	24	0.946	6	0.878	20	0.877	17	0.753	26	0.838	22	0.931	12	0.916	14	0.828	17
A6	0.484	23	0.946	5	0.878	21	0.877	15	0.753	26	0.838	21	0.931	11	0.916	14	0.828	15
A7	0.484	22	0.946	5	0.878	21	0.877	15	0.753	25	0.838	20	0.931	9	0.916	11	0.828	15
Avg	0.484	21	0.946	5	0.878	19	0.877	14	0.753	24	0.838	20	0.931	12	0.916	15	**0.828**	**14**

(d) Optimal F1 for `nofa.ehdNgryhst`

Table B.3 – Optimal NDCR and Optimal F1 for submissions `nofa.ehdNghT10` and `nofa.ehdNgryhst` at TRECVID 2010.
means the rank between 37 submissions.

#	Team	Run	Type	Avg. Optimal		Avg. Actual		Avg.
				NDCR	F1	NDCR	F1	MTP
1	PKU-IDM	perseus	AV	0.032	0.889	0.151	0.889	11925
2	PKU-IDM	kraken	AV	0.118	0.892	0.356	0.890	9631
3	NTT-CSL	1	AV	0.139	0.933	0.139	0.933	25
4	NTT-CSL	2	AV	0.140	0.933	0.140	0.933	25
5	INRIA-LEAR-TEXMEX	lapin	AV	0.198	0.649	15.652	0.646	537
6	KDDILabs-SRI	1	AV	0.297	0.960	0.413	0.960	84
7	KDDILabs-SRI	2	AV	0.317	0.961	2.366	0.954	84
8	ATTLabs	2	AV	0.356	0.803	0.818	0.802	41
9	ATTLabs	4	AV	0.398	0.800	1.346	0.788	41
10	nii	av	AV	0.485	0.897	0.611	0.898	10
11	INRIA-LEAR-TEXMEX	truite	AV	0.488	0.877	4.531	0.853	719
12	nii	a	A	0.519	0.902	0.611	0.898	9
13	TID	rawfusion	AV	0.529	0.921	1.196	0.913	601
14	**PRISMA**	**ehdNgryhst**	**V**	**0.597**	**0.820**	**9.057**	**0.723**	**128**
15	IBM	gistG	V	0.612	0.775	0.664	0.773	14
16	**PRISMA**	**ehdNclrhst**	**V**	**0.658**	**0.820**	**8.902**	**0.724**	**132**
17	TID	localvideo	V	0.741	0.941	4.829	0.932	441
18	VIREO	srpeflip	V	0.799	0.462	3.840	0.607	144
19	VIREO	srpe	V	0.839	0.475	3.472	0.608	72
20	TUBITAK_UZAY	aindexb	A	0.847	0.384	9.827	0.886	4
21	TUBITAK_UZAY	avb	AV	0.958	0.392	2.463	0.905	31
22	asahikasei	VmainAsub	AV	0.971	0.493	1.475	0.491	224
23	THU-IMG	dragon	V	0.987	0.723	28.967	0.692	110
24	THU-IMG	linnet	V	0.994	0.722	35.364	0.690	110
25	brno	l3sl2	AV	1.006	0.452	1.823	0.565	1003
26	brno	l3sl2X	AV	1.006	0.452	1.823	0.565	1003
27	asahikasei	AmainVsub	AV	1.028	0.469	5.123	0.622	407
28	NTNU-Academia-Sinica	1	V	1.032	0.505	3.051	0.704	111
29	NJU	comp2	V	1.077	0.084	59.086	0.489	94
31	NTNU-Academia-Sinica	3	AV	1.107	0.000	16.522	0.781	11
30	SYSU-GITL	sysuc1	V	1.107	0.000	21.919	0.000	227
32	CCU	submission	AV	1.111	0.000	3.715	0.000	81
33	TUBITAK_UZAY	sift1024b	V	1.120	0.000	19.020	0.650	26
34	ITU_MSPR	ITUMSPR1	AV	1.155	0.561	1.155	0.561	956
35	TID	rawlocaud	A	1.202	0.822	43.946	0.927	2167
36	BUPT-MCPRL	SD	AV	1.224	0.867	1.224	0.867	27
37	NJU	rank1	V	1.240	0.120	39.924	0.508	94
38	BUPT-MCPRL	TF	AV	1.918	0.867	1.918	0.867	27
39	IDARE	test	AV	10.428	0.265	253.035	0.300	3
40	UNIBS	MF	V	26.130	0.524	142.569	0.587	12
41	UNIBS	SF	V	26.377	0.774	144.287	0.744	11

Table B.4 – Results for Balanced profile at TRECVID 2010. Values averaged for the 56 transformations. Submissions in descending order by Average Optimal NDCR.

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.962	16	0.454	9	0.346	12	0.608	16	0.638	19	0.808	19	0.762	16	0.685	16	0.658	16
A2	0.962	16	0.454	7	0.346	11	0.608	14	0.638	16	0.808	19	0.762	17	0.685	14	0.658	14
A3	0.962	15	0.454	7	0.346	11	0.608	13	0.638	17	0.808	18	0.762	16	0.685	14	0.658	15
A4	0.962	13	0.454	7	0.346	11	0.608	15	0.638	17	0.808	18	0.762	15	0.685	15	0.658	15
A5	0.962	17	0.454	10	0.346	13	0.608	17	0.638	19	0.808	21	0.762	18	0.685	18	0.658	17
A6	0.962	17	0.454	9	0.346	13	0.608	17	0.638	19	0.808	20	0.762	17	0.685	17	0.658	17
A7	0.962	17	0.454	9	0.346	13	0.608	17	0.638	19	0.808	20	0.762	18	0.685	17	0.658	17
Avg	0.962	17	0.454	7	0.346	11	0.608	16	0.638	18	0.808	19	0.762	17	0.685	16	**0.658**	**16**

(a) Optimal NDCR for `balanced.ehdNclrhst`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.583	23	0.893	11	0.894	16	0.809	23	0.849	20	0.767	26	0.952	7	0.813	21	0.820	16
A2	0.583	23	0.893	13	0.894	16	0.809	22	0.849	20	0.767	24	0.952	7	0.813	21	0.820	16
A3	0.583	19	0.893	10	0.894	15	0.809	22	0.849	18	0.767	22	0.952	5	0.813	19	0.820	14
A4	0.583	20	0.893	10	0.894	16	0.809	25	0.849	19	0.767	23	0.952	7	0.813	21	0.820	15
A5	0.583	25	0.893	11	0.894	16	0.809	27	0.849	18	0.767	26	0.952	7	0.813	23	0.820	18
A6	0.583	22	0.893	10	0.894	14	0.809	23	0.849	19	0.767	25	0.952	6	0.813	22	0.820	17
A7	0.583	22	0.893	9	0.894	12	0.809	23	0.849	18	0.767	24	0.952	6	0.813	24	0.820	17
Avg	0.583	20	0.893	10	0.894	13	0.809	21	0.849	18	0.767	23	0.952	5	0.813	20	**0.820**	**15**

(b) Optimal F1 for `balanced.ehdNclrhst`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.977	18	0.515	11	0.269	10	0.262	11	0.777	20	0.708	18	0.562	14	0.708	17	0.597	14
A2	0.977	18	0.515	9	0.269	10	0.262	9	0.777	20	0.708	16	0.562	12	0.708	15	0.597	12
A3	0.977	17	0.515	9	0.269	8	0.262	8	0.777	19	0.708	17	0.562	10	0.708	15	0.597	11
A4	0.977	15	0.515	11	0.269	9	0.262	9	0.777	18	0.708	17	0.562	13	0.708	16	0.597	13
A5	0.977	19	0.515	14	0.269	11	0.262	9	0.777	21	0.708	19	0.562	15	0.708	19	0.597	15
A6	0.977	20	0.515	12	0.269	9	0.262	9	0.777	21	0.708	19	0.562	15	0.708	18	0.597	15
A7	0.977	20	0.515	11	0.269	10	0.262	9	0.777	20	0.708	19	0.562	14	0.708	18	0.597	15
Avg	0.977	19	0.515	9	0.269	9	0.262	9	0.777	20	0.708	18	0.562	14	0.708	17	**0.597**	**14**

(c) Optimal NDCR for `balanced.ehdNgryhst`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.484	24	0.888	14	0.878	18	0.877	15	0.747	24	0.838	21	0.931	13	0.916	12	0.820	17
A2	0.484	24	0.888	14	0.878	20	0.877	16	0.747	24	0.838	20	0.931	12	0.916	12	0.820	17
A3	0.484	21	0.888	13	0.878	19	0.877	13	0.747	22	0.838	18	0.931	11	0.916	12	0.820	15
A4	0.484	22	0.888	14	0.878	18	0.877	14	0.747	23	0.838	19	0.931	13	0.916	13	0.820	16
A5	0.484	26	0.888	13	0.878	19	0.877	17	0.747	26	0.838	21	0.931	12	0.916	12	0.820	19
A6	0.484	24	0.888	12	0.878	17	0.877	14	0.747	25	0.838	20	0.931	9	0.916	11	0.820	18
A7	0.484	23	0.888	12	0.878	17	0.877	14	0.747	25	0.838	19	0.931	9	0.916	11	0.820	18
Avg	0.484	22	0.888	13	0.878	16	0.877	13	0.747	23	0.838	18	0.931	10	0.916	9	**0.820**	**16**

(d) Optimal F1 for `balanced.ehdNgryhst`

Table B.5 – Optimal NDCR and Optimal F1 for submissions `balanced.ehdNclrhst` and `balanced.ehdNgryhst` at TRECVID 2010.
means the rank between 41 submissions.

#	Team	Profile	Run	Type	R_{P1}	Precision
1	PKU-IDM	nofa	perseus	AV	0.911	1
2	PKU-IDM	balanced	perseus	AV	0.885	1
3	PKU-IDM	nofa	kraken	AV	0.882	1
4	NTT-CSL	balanced	1	AV	0.836	1
5	NTT-CSL	balanced	2	AV	0.835	1
6	NTT-CSL	nofa	3	AV	0.835	1
7	PKU-IDM	balanced	kraken	AV	0.817	1
8	NTT-CSL	nofa	0	AV	0.816	1
9	INRIA-LEAR-TEXMEX	balanced	lapin	AV	0.711	1
10	INRIA-LEAR-TEXMEX	nofa	mouflon	AV	0.578	1
11	KDDILabs-SRI	nofa	1	AV	0.346	1
12	IBM	balanced	gistG	V	0.344	1
13	IBM	nofa	gistG	V	0.341	1
14	KDDILabs-SRI	balanced	2	AV	0.314	1
15	KDDILabs-SRI	nofa	2	AV	0.314	1
16	KDDILabs-SRI	balanced	1	AV	0.285	1
17	**PRISMA**	**nofa**	**ehdNghT10**	**V**	**0.248**	**1**
18	**PRISMA**	**nofa**	**ehdNgryhst**	**V**	**0.248**	**1**
19	**PRISMA**	**balanced**	**ehdNgryhst**	**V**	**0.204**	**1**
20	INRIA-LEAR-TEXMEX	nofa	bouquetin	AV	0.203	1
21	**PRISMA**	**balanced**	**ehdNclrhst**	**V**	**0.176**	**1**
22	INRIA-LEAR-TEXMEX	balanced	truite	AV	0.158	1
23	IBM	nofa	gistGCsift	V	0.132	1
24	IBM	nofa	gistGC	V	0.111	1
25	VIREO	balanced	srpeflip	V	0.018	1
26	VIREO	nofa	srpeflip	V	0.018	1
27	asahikasei	balanced	AmainVsub	AV	0.016	1
28	asahikasei	nofa	AmainVsub	AV	0.016	1
29	VIREO	balanced	srpe	V	0.015	1
30	NJU	nofa	norank1	V	0.010	1
31	TID	balanced	localvideo	V	0.009	1
32	nii	balanced	av	AV	0.008	1
33	nii	nofa	av	AV	0.008	1
34	ATTLabs	balanced	2	AV	0.004	1
35	ATTLabs	nofa	1	AV	0.004	1
36	asahikasei	balanced	VmainAsub	AV	0.002	1
37	asahikasei	nofa	VmainAsub	AV	0.002	1
38	NJU	balanced	comp2	V	0.002	1
39	NJU	nofa	comp2	V	0.002	1

continue in Table B.7...

Table B.6 – R_{P1} (maximum Recall with Precision 1) for all TRECVID 2010 submissions. Part 1 of 2.

#	Team	Profile	Run	Type	R_{P1}	Precision
continuation from Table B.6...						
40	NTNU-Academia-Sinica	balanced	1	V	0.001	1
41	NTNU-Academia-Sinica	nofa	2	V	0.001	1
42	ATTLabs	balanced	4	AV	0.000	1
43	ATTLabs	nofa	3	AV	0.000	1
–	brno	balanced	l3sl2	AV	—	—
–	brno	balanced	l3sl2X	AV	—	—
–	brno	nofa	l3sl	AV	—	—
–	brno	nofa	l3sl2	AV	—	—
–	BUPT-MCPRL	balanced	SD	AV	—	—
–	BUPT-MCPRL	balanced	TF	AV	—	—
–	BUPT-MCPRL	nofa	SD	AV	—	—
–	BUPT-MCPRL	nofa	TF	AV	—	—
–	CCU	balanced	submission	AV	—	—
–	CCU	nofa	submission	AV	—	—
–	IDARE	balanced	test	AV	—	—
–	IDARE	nofa	test	—	—	—
–	ITU_MSPR	balanced	ITUMSPR1	AV	—	—
–	ITU_MSPR	nofa	ITUMSPR2	AV	—	—
–	nii	balanced	a	A	—	—
–	nii	nofa	a	A	—	—
–	NJU	balanced	rank1	V	—	—
–	NTNU-Academia-Sinica	balanced	3	AV	—	—
–	SYSU-GITL	balanced	sysuc1	V	—	—
–	SYSU-GITL	nofa	sysuc2	V	—	—
–	THU-IMG	balanced	dragon	V	—	—
–	THU-IMG	balanced	linnet	V	—	—
–	THU-IMG	nofa	tiger	V	—	—
–	THU-IMG	nofa	tortoise	V	—	—
–	TID	balanced	rawfusion	AV	—	—
–	TID	balanced	rawlocaud	A	—	—
–	TID	nofa	rawfusion	AV	—	—
–	TUBITAK_UZAY	balanced	aindexb	A	—	—
–	TUBITAK_UZAY	balanced	avb	AV	—	—
–	TUBITAK_UZAY	balanced	sift1024b	V	—	—
–	TUBITAK_UZAY	nofa	aindexnf	A	—	—
–	UNIBS	balanced	MF	V	—	—
–	UNIBS	balanced	SF	V	—	—
–	UNIBS	nofa	MF	V	—	—
–	UNIBS	nofa	SF	V	—	—

Table B.7 – R_{P1} (maximum Recall with Precision 1) for all TRECVID 2010 submissions. Part 2 of 2.

#	Team	Profile	Run	Type	$R_{P.5}$	Precision
1	INRIA-LEAR-TEXMEX	balanced	lapin	AV	0.964	0.512
2	INRIA-LEAR-TEXMEX	balanced	truite	AV	0.961	0.520
3	PKU-IDM	balanced	perseus	AV	0.954	0.990
4	PKU-IDM	balanced	kraken	AV	0.947	0.976
5	INRIA-LEAR-TEXMEX	nofa	bouquetin	AV	0.943	0.608
6	BUPT-MCPRL	balanced	TF	AV	0.927	0.877
7	BUPT-MCPRL	balanced	SD	AV	0.925	0.919
8	BUPT-MCPRL	nofa	TF	AV	0.920	0.952
9	BUPT-MCPRL	nofa	SD	AV	0.918	0.961
10	PKU-IDM	nofa	perseus	AV	0.911	1.000
11	INRIA-LEAR-TEXMEX	nofa	mouflon	AV	0.907	0.590
12	PKU-IDM	nofa	kraken	AV	0.882	1.000
13	TID	balanced	rawfusion	AV	0.849	0.510
14	TID	nofa	rawfusion	AV	0.849	0.510
15	NTT-CSL	balanced	1	AV	0.836	1.000
16	NTT-CSL	balanced	2	AV	0.835	1.000
17	NTT-CSL	nofa	3	AV	0.835	1.000
18	ATTLabs	balanced	4	AV	0.818	0.574
19	NTT-CSL	nofa	0	AV	0.816	1.000
20	KDDILabs-SRI	balanced	2	AV	0.784	0.652
21	KDDILabs-SRI	nofa	2	AV	0.784	0.652
22	KDDILabs-SRI	balanced	1	AV	0.780	0.784
23	KDDILabs-SRI	nofa	1	AV	0.780	0.784
24	nii	balanced	av	AV	0.754	0.771
25	nii	nofa	av	AV	0.734	0.966
26	ATTLabs	balanced	2	AV	0.719	0.518
27	ATTLabs	nofa	3	AV	0.701	0.988
28	TUBITAK_UZAY	balanced	aindexb	A	0.681	0.519
29	TUBITAK_UZAY	nofa	aindexnf	A	0.681	0.519
30	nii	balanced	a	A	0.674	0.971
31	nii	nofa	a	A	0.674	0.971
32	TID	balanced	localvideo	V	0.672	0.515
33	ATTLabs	nofa	1	AV	0.637	0.995
34	TUBITAK_UZAY	balanced	avb	AV	0.560	0.797
35	VIREO	balanced	srpeflip	V	0.543	0.697
36	**PRISMA**	**balanced**	**ehdNgryhst**	**V**	**0.539**	**0.508**
37	VIREO	nofa	srpeflip	V	0.530	0.742
38	**PRISMA**	**nofa**	**ehdNghT10**	**V**	**0.526**	**0.501**
39	**PRISMA**	**nofa**	**ehdNgryhst**	**V**	**0.526**	**0.501**

continue in Table B.9...

Table B.8 – $R_{P.5}$ (maximum Recall with Precision greater of equal than 0.5) for all TRECVID 2010 submissions. Part 1 of 2.

#	Team	Profile	Run	Type	$R_{P.5}$	Precision
continuation from Table B.8...						
40	IBM	balanced	gistG	V	0.494	0.510
41	ITU_MSPR	balanced	ITUMSPR1	AV	0.491	0.914
42	IBM	nofa	gistG	V	0.486	0.558
43	VIREO	balanced	srpe	V	0.484	0.705
44	TID	balanced	rawlocaud	A	0.470	0.500
45	ITU_MSPR	nofa	ITUMSPR2	AV	0.466	0.967
46	IBM	nofa	gistGCsift	V	0.464	0.516
47	IBM	nofa	gistGC	V	0.449	0.564
48	**PRISMA**	**balanced**	**ehdNclrhst**	**V**	**0.442**	**0.501**
49	NTNU-Academia-Sinica	balanced	1	V	0.312	0.502
50	NTNU-Academia-Sinica	nofa	2	V	0.312	0.502
51	THU-IMG	nofa	tortoise	V	0.303	0.504
52	THU-IMG	nofa	tiger	V	0.298	0.500
53	THU-IMG	balanced	linnet	V	0.283	0.505
54	THU-IMG	balanced	dragon	V	0.271	0.500
55	asahikasei	balanced	VmainAsub	AV	0.112	0.512
56	asahikasei	nofa	VmainAsub	AV	0.112	0.577
57	brno	balanced	l3sl2	AV	0.084	0.518
58	brno	balanced	l3sl2X	AV	0.084	0.518
59	brno	nofa	l3sl2	AV	0.084	0.518
60	asahikasei	balanced	AmainVsub	AV	0.082	0.527
61	asahikasei	nofa	AmainVsub	AV	0.082	0.527
62	brno	nofa	l3sl	AV	0.052	0.517
63	NJU	nofa	norank1	V	0.049	0.515
64	NJU	balanced	comp2	V	0.003	0.750
65	NJU	nofa	comp2	V	0.003	0.750
–	CCU	balanced	submission	AV	—	—
–	CCU	nofa	submission	AV	—	—
–	IDARE	balanced	test	AV	—	—
–	IDARE	nofa	test	—	—	—
–	NJU	balanced	rank1	V	—	—
–	NTNU-Academia-Sinica	balanced	3	AV	—	—
–	SYSU-GITL	balanced	sysuc1	V	—	—
–	SYSU-GITL	nofa	sysuc2	V	—	—
–	TUBITAK_UZAY	balanced	sift1024b	V	—	—
–	UNIBS	balanced	MF	V	—	—
–	UNIBS	balanced	SF	V	—	—
–	UNIBS	nofa	MF	V	—	—
–	UNIBS	nofa	SF	V	—	—

Table B.9 – $R_{P.5}$ (maximum Recall with Precision greater of equal than 0.5) for all TRECVID 2010 submissions. Part 2 of 2.

Appendix C

Results at TRECVID 2011

Team 2011	Organization	Location	Paper
ATTLabs	AT&T Labs Research	NorthAm	[Liu et al., 2011]
brno	Brno University of Technology	Europe	[Hradiš et al., 2011]
BUPT-MCPRL	Beijing University of Posts and Telecommunications-MCPRL	Asia	[Zhao et al., 2011b]
CRIM-VISI	Computer Research Institute of Montreal - Vision & Imaging team	NorthAm	[Gupta et al., 2011]
FTRDBJ	France Telecom Orange Labs (Beijing)	Asia	[Bai et al., 2011]
IMP	Osaka Prefecture University	Asia	[Sakata et al., 2011]
INRIA-LEAR	INRIA-LEAR	Europe	[Ayari et al., 2011]
INRIA-TEXMEX	INRIA/IRISA	Europe	[Ayari et al., 2011]
ITU_MSPR	Istanbul Technical University	Europe	—
iupr-dfki	University of Kaiserslautern	Europe	[Zhao et al., 2011a]
KDDILabs	KDDILabs	Asia	[Uchida et al., 2011]
NTT-CSL	NTT Communication Science Laboratories-CSL	Asia	[Mukai et al., 2011]
PKU-IDM	Peking University-IDM	Asia	[Jiang et al., 2011]
PRISMA	**PRISMA-University of Chile**	**SouthAm**	**[Barrios et al., 2011]**
RMIT	RMIT University School of CS&IT	Australia	[Rouhi and Thom, 2011]
SYSU-GITL	Sun Yat-sen University - GITL	Asia	—
Telefonica.research	Telefonica Research	Europe	[Anguera et al., 2011a]
tokushima_U	Tokushima University	Asia	[Shishibori et al., 2011]
UQMSG	University of Queensland	Australia	[Shen et al., 2011]
USC-UTSA	USC Viterbi School of Engineering	NorthAm	—
XJTU	Xi'an Jiaotong University	Asia	—
ZJU_CS_IV	Zhejiang University	Asia	—

Table C.1 – The 22 participant teams in CCD evaluation at TRECVID 2011.

#	Team	Run	Type	Avg. Optimal		Avg. Actual		Avg.
				NDCR	F1	NDCR	F1	MTP
1	PKU-IDM	cascade	AV	0.078	0.950	0.080	0.950	179
2	INRIA-LEAR	dodo	AV	0.103	0.942	5.911	0.947	2079
3	CRIM-VISI	V48A66T160	AV	0.122	0.711	42.073	0.711	2792
4	CRIM-VISI	V48A66T60	AV	0.122	0.711	85.889	0.715	2792
5	INRIA-TEXMEX	tyche	AV	0.252	0.934	0.352	0.935	32848
6	**PRISMA**	**EhdRgbAud**	**AV**	**0.286**	**0.946**	**0.336**	**0.962**	**64**
7	NTT-CSL	0	AV	0.311	0.931	106.897	0.924	96
8	ATTLabs	1	AV	0.330	0.879	0.508	0.890	30
9	ATTLabs	3	AV	0.342	0.875	0.506	0.886	42
10	**PRISMA**	**EhdGry**	**V**	**0.374**	**0.938**	**0.419**	**0.956**	**50**
11	FTRDBJ	AudioOnly	A	0.410	0.887	76.678	0.880	4589
12	FTRDBJ	orange1	AV	0.511	0.927	219.410	0.917	4589
13	ZJU_CS_IV	bhgccd	V	0.544	0.952	0.545	0.952	801
14	BUPT-MCPRL	wsyVA	AV	0.587	0.895	404.343	0.892	62
15	ZJU_CS_IV	bgccd	V	0.626	0.956	0.627	0.956	445
16	IMP	Uvote	V	0.748	0.654	374.333	0.650	191
17	IMP	Wvote	V	0.773	0.658	107.503	0.680	184
18	KDDILabs	4sys	V	13.807	0.682	27.161	0.645	4
19	SYSU-GITL	videoonly1	AV	14.243	0.785	2723.846	0.693	727
20	brno	brnoccd	AV	23.812	0.709	197.237	0.705	1575
21	KDDILabs	base	V	27.236	0.683	0.720	0.732	1
22	KDDILabs	2sys	V	27.237	0.621	13.927	0.636	2
23	SYSU-GITL	videoonly2	V	27.590	0.773	3057.516	0.739	719
24	Telefonica-research	multimodal	AV	57.768	0.948	153.092	0.949	601
25	USC-UTSA	test	V	107.79	0.000	961696	0.280	4
26	UQMSG	mfh	AV	117.32	0.000	296.575	0.000	1
27	ITU_MSPR	ITUMSPR1	A	137.98	0.370	137.975	0.370	953
28	tokushima_U	ch4of12	A	275.33	0.930	854.986	0.915	110
29	BUPT-MCPRL	zhVideo	V	400.57	0.914	400.577	0.915	62
30	RMIT	VideoNOFA7	V	401.45	0.054	5687.435	0.391	142
31	RMIT	VideoNOFA8	V	401.45	0.054	3284.710	0.343	142
32	XJTU	1	V	9999	0.000	13830	0.000	50

Table C.2 – Results for Nofa profile at TRECVID 2011. Values averaged for the 56 transformations. Submissions in descending order by Average Optimal NDCR.

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.761	14	0.321	6	0.119	10	0.239	10	0.403	18	0.515	17	0.209	4	0.425	11	0.374	11
A2	0.761	14	0.321	5	0.119	9	0.239	10	0.403	17	0.515	17	0.209	6	0.425	11	0.374	11
A3	0.761	13	0.321	6	0.119	9	0.239	9	0.403	14	0.515	15	0.209	6	0.425	10	0.374	9
A4	0.761	13	0.321	7	0.119	8	0.239	10	0.403	14	0.515	17	0.209	6	0.425	10	0.374	10
A5	0.761	13	0.321	8	0.119	9	0.239	10	0.403	15	0.515	16	0.209	7	0.425	10	0.374	10
A6	0.761	13	0.321	7	0.119	9	0.239	10	0.403	15	0.515	16	0.209	7	0.425	10	0.374	10
A7	0.761	13	0.321	7	0.119	9	0.239	10	0.403	15	0.515	14	0.209	7	0.425	8	0.374	8
Avg	0.761	13	0.321	7	0.119	8	0.239	9	0.403	16	0.515	16	0.209	6	0.425	11	**0.374**	**10**

(a) Optimal NDCR for `nofa.EhdGry`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.887	15	0.945	9	0.966	4	0.949	8	0.952	7	0.924	11	0.934	12	0.944	7	0.938	10
A2	0.887	14	0.945	10	0.966	2	0.949	8	0.952	5	0.924	11	0.934	11	0.944	6	0.938	9
A3	0.887	16	0.945	9	0.966	2	0.949	8	0.952	5	0.924	9	0.934	10	0.944	6	0.938	8
A4	0.887	13	0.945	6	0.966	2	0.949	7	0.952	6	0.924	10	0.934	11	0.944	4	0.938	8
A5	0.887	14	0.945	5	0.966	2	0.949	6	0.952	3	0.924	10	0.934	9	0.944	6	0.938	7
A6	0.887	13	0.945	6	0.966	1	0.949	7	0.952	5	0.924	8	0.934	10	0.944	6	0.938	7
A7	0.887	9	0.945	5	0.966	2	0.949	7	0.952	3	0.924	8	0.934	10	0.944	5	0.938	7
Avg	0.887	14	0.945	8	0.966	2	0.949	6	0.952	4	0.924	10	0.934	10	0.944	5	**0.938**	**7**

(b) Optimal F1 for `nofa.EhdGry`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.358	8	0.216	2	0.075	6	0.142	7	0.284	12	0.299	9	0.104	2	0.254	6	0.217	6
A2	0.358	8	0.209	4	0.075	6	0.142	8	0.269	12	0.284	8	0.104	4	0.261	5	0.213	5
A3	0.575	10	0.284	5	0.112	8	0.149	8	0.418	16	0.366	11	0.172	4	0.358	6	0.304	6
A4	0.694	10	0.306	6	0.119	8	0.149	8	0.463	18	0.388	12	0.157	5	0.358	7	0.329	6
A5	0.493	11	0.239	6	0.104	7	0.142	9	0.358	14	0.336	11	0.127	4	0.291	6	0.261	7
A6	0.664	10	0.321	7	0.142	11	0.157	9	0.433	16	0.381	11	0.187	6	0.358	8	0.330	8
A7	0.716	11	0.321	7	0.142	10	0.172	9	0.478	16	0.403	10	0.179	6	0.366	7	0.347	7
Avg	0.551	10	0.271	6	0.110	7	0.150	8	0.386	15	0.351	11	0.147	4	0.321	5	**0.286**	**6**

(c) Optimal NDCR for `nofa.EhdRgbAud`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.927	7	0.948	7	0.977	1	0.964	2	0.928	14	0.931	9	0.974	1	0.949	4	0.950	5
A2	0.931	6	0.949	7	0.975	1	0.969	1	0.930	12	0.925	10	0.971	1	0.954	3	0.951	5
A3	0.890	15	0.951	5	0.969	1	0.958	2	0.940	12	0.923	10	0.969	1	0.958	2	0.945	6
A4	0.882	16	0.948	4	0.971	1	0.958	1	0.940	13	0.948	7	0.968	1	0.957	2	0.947	6
A5	0.904	10	0.942	7	0.972	1	0.960	1	0.931	13	0.933	7	0.973	1	0.962	1	0.947	5
A6	0.866	16	0.947	5	0.966	1	0.956	4	0.919	13	0.924	8	0.970	1	0.963	1	0.939	6
A7	0.876	13	0.944	6	0.969	1	0.962	1	0.936	11	0.940	6	0.964	1	0.957	2	0.944	5
Avg	0.897	12	0.947	5	0.971	1	0.961	1	0.932	13	0.932	8	0.970	1	0.957	2	**0.946**	**5**

(d) Optimal F1 for `nofa.EhdRgbAud`

Table C.3 – Optimal NDCR and Optimal F1 for submissions `nofa.EhdGry` and `nofa.EhdRgbAud` at TRECVID 2011.
means the rank between 32 submissions.

#	Team	Run	Type	Avg. Optimal NDCR	Avg. Optimal F1	Avg. Actual NDCR	Avg. Actual F1	Avg. MTP
1	PKU-IDM	cascade	AV	0.053	0.949	0.055	0.949	172
2	CRIM-VISI	V48A66T58B	AV	0.117	0.712	0.163	0.715	2792
3	CRIM-VISI	V48A66T65B	AV	0.117	0.712	0.159	0.715	2792
4	INRIA-LEAR	dodo	AV	0.144	0.942	0.217	0.944	2079
5	INRIA-TEXMEX	zozo	AV	0.194	0.929	0.348	0.936	32848
6	INRIA-TEXMEX	themis	AV	0.211	0.929	0.351	0.936	32848
7	NTT-CSL	1	AV	0.244	0.940	0.306	0.936	96
8	INRIA-LEAR	deaf	V	0.258	0.950	0.362	0.951	2041
9	**Telefonica-research**	**joint**	**AV**	**0.268**	**0.957**	**1.209**	**0.944**	**601**
10	NTT-CSL	2	AV	0.270	0.930	0.384	0.924	96
11	FTRDBJ	orange3	AV	0.287	0.920	0.340	0.917	4589
12	**PRISMA**	**EhdRgbAud**	**AV**	**0.300**	**0.955**	**8.462**	**0.935**	**64**
13	NTT-CSL	3	AV	0.309	0.943	0.474	0.935	96
14	ATTLabs	2	AV	0.317	0.879	0.492	0.889	30
15	ATTLabs	4	AV	0.330	0.876	0.509	0.887	42
16	FTRDBJ	VideoOnly	V	0.335	0.918	0.388	0.917	4589
17	INRIA-TEXMEX	audioonly	A	0.406	0.910	0.545	0.917	192
18	**PRISMA**	**EhdGry**	**V**	**0.412**	**0.938**	**3.716**	**0.913**	**50**
19	KDDILabs	4sys	V	0.471	0.682	0.490	0.645	4
20	BUPT-MCPRL	zhVideo	V	0.529	0.915	0.636	0.915	62
21	ZJU_CS_IV	bhgccd	V	0.544	0.952	0.545	0.952	801
22	BUPT-MCPRL	wsyVA	AV	0.578	0.892	1.213	0.891	62
23	Telefonica-research	multimodal	AV	0.610	0.947	3.595	0.907	601
24	ZJU_CS_IV	bgccd	V	0.626	0.956	0.627	0.956	445
25	Telefonica-research	mask	AV	0.662	0.729	1.085	0.708	2393
26	IMP	Wvote	V	0.681	0.652	0.888	0.637	184
27	IMP	Uvote	V	0.726	0.661	1.401	0.632	188
28	iupr-dfki	fsift	V	0.836	0.639	4.479	0.563	84
29	SYSU-GITL	videoonly1	AV	0.893	0.786	3.467	0.693	727
30	SYSU-GITL	videoonly2	V	0.909	0.773	3.757	0.739	719
31	brno	brnoccd	AV	0.911	0.800	2.317	0.744	1575
32	iupr-dfki	fsift2	V	0.962	0.581	6.100	0.569	84
33	tokushima_U	ch4of12	A	1.005	0.930	1.533	0.915	110
34	ITU_MSPR	ITUMSPR2	AV	1.125	0.417	1.125	0.417	953
35	UQMSG	mfh	AV	1.193	0.000	6.408	0.001	1
36	tokushima_U	chth	A	1.256	0.928	4.122	0.922	248
37	RMIT	VideoBal5	V	1.395	0.054	25.230	0.470	142
38	RMIT	VideoBal6	V	1.395	0.054	13.219	0.421	142
39	tokushima_U	chcode	A	1.989	0.929	7.120	0.917	120
40	XJTU	1	V	19.087	0.000	41.766	0.000	49
41	USC-UTSA	test	AV	19.089	0.242	1834	0.304	4

Table C.4 – Results for Balanced profile at TRECVID 2011. Values averaged for the 56 transformations. Submissions in descending order by Average Optimal NDCR.

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.761	23	0.313	11	0.104	8	0.323	15	0.495	23	0.607	23	0.293	15	0.403	17	0.412	18
A2	0.761	23	0.313	11	0.104	9	0.323	16	0.495	22	0.607	24	0.293	17	0.403	17	0.412	18
A3	0.761	22	0.313	10	0.104	7	0.323	12	0.495	21	0.607	23	0.293	14	0.403	13	0.412	14
A4	0.761	22	0.313	10	0.104	8	0.323	15	0.495	22	0.607	24	0.293	14	0.403	17	0.412	18
A5	0.761	21	0.313	10	0.104	7	0.323	14	0.495	21	0.607	21	0.293	15	0.403	16	0.412	17
A6	0.761	20	0.313	10	0.104	11	0.323	14	0.495	19	0.607	21	0.293	17	0.403	16	0.412	17
A7	0.761	21	0.313	10	0.104	7	0.323	13	0.495	20	0.607	21	0.293	14	0.403	14	0.412	15
Avg	0.761	22	0.313	10	0.104	7	0.323	14	0.495	21	0.607	22	0.293	16	0.403	16	**0.412**	**18**

(a) Optimal NDCR for `balanced.EhdGry`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.887	24	0.945	11	0.966	5	0.950	9	0.952	8	0.926	17	0.936	19	0.941	13	0.938	15
A2	0.887	23	0.945	10	0.966	3	0.950	10	0.952	9	0.926	17	0.936	19	0.941	12	0.938	13
A3	0.887	23	0.945	12	0.966	3	0.950	10	0.952	10	0.926	16	0.936	17	0.941	9	0.938	13
A4	0.887	22	0.945	9	0.966	3	0.950	8	0.952	9	0.926	17	0.936	19	0.941	9	0.938	13
A5	0.887	24	0.945	8	0.966	3	0.950	6	0.952	5	0.926	15	0.936	13	0.941	8	0.938	9
A6	0.887	22	0.945	10	0.966	3	0.950	8	0.952	7	0.926	14	0.936	16	0.941	9	0.938	10
A7	0.887	16	0.945	6	0.966	3	0.950	7	0.952	5	0.926	10	0.936	14	0.941	7	0.938	9
Avg	0.887	24	0.945	9	0.966	3	0.950	7	0.952	6	0.926	16	0.936	15	0.941	7	**0.938**	**11**

(b) Optimal F1 for `balanced.EhdGry`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#	NDCR	#
A1	0.358	12	0.194	2	0.075	5	0.134	4	0.343	19	0.313	16	0.119	2	0.246	8	0.223	6
A2	0.343	10	0.179	4	0.075	6	0.149	7	0.328	19	0.321	17	0.127	4	0.254	11	0.222	6
A3	0.575	13	0.269	8	0.112	10	0.209	8	0.455	20	0.388	15	0.187	5	0.328	10	0.315	11
A4	0.679	21	0.291	9	0.127	12	0.216	9	0.493	21	0.373	17	0.216	7	0.336	14	0.341	15
A5	0.470	14	0.216	6	0.104	7	0.187	8	0.425	19	0.373	17	0.164	5	0.284	10	0.278	11
A6	0.664	18	0.306	9	0.142	15	0.246	11	0.500	20	0.425	17	0.231	10	0.343	15	0.357	16
A7	0.701	19	0.306	9	0.127	12	0.224	9	0.515	22	0.448	17	0.224	9	0.358	13	0.363	14
Avg	0.541	16	0.252	9	0.109	9	0.195	8	0.437	20	0.377	17	0.181	5	0.307	11	**0.300**	**12**

(c) Optimal NDCR for `balanced.EhdRgbAud`

	V1		V2		V3		V4		V5		V6		V8		V10		Avg	
	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#	F1	#
A1	0.927	13	0.949	7	0.977	1	0.971	1	0.952	8	0.951	9	0.976	1	0.949	5	0.957	5
A2	0.932	11	0.950	6	0.976	1	0.976	1	0.960	4	0.951	9	0.973	1	0.954	2	0.959	1
A3	0.890	22	0.952	5	0.971	1	0.978	1	0.960	4	0.961	1	0.976	1	0.959	1	0.956	4
A4	0.887	22	0.949	6	0.974	1	0.971	1	0.956	5	0.948	9	0.973	1	0.957	2	0.952	6
A5	0.907	15	0.943	10	0.972	1	0.969	1	0.954	4	0.945	7	0.976	1	0.962	1	0.954	3
A6	0.866	23	0.947	7	0.970	1	0.978	1	0.958	5	0.958	2	0.978	1	0.963	1	0.952	4
A7	0.881	18	0.944	8	0.969	1	0.974	1	0.956	4	0.964	2	0.977	1	0.957	2	0.953	3
Avg	0.899	19	0.948	7	0.973	1	0.974	1	0.957	3	0.954	4	0.976	1	0.957	2	**0.955**	**3**

(d) Optimal F1 for `balanced.EhdRgbAud`

Table C.5 – Optimal NDCR and Optimal F1 for submissions `balanced.EhdGry` and `balanced.EhdRgbAud` at TRECVID 2011.
means the rank between 41 submissions.

#	Team	Profile	Run	Type	R_{P1}	Precision
1	PKU-IDM	balanced	cascade	AV	0.947	1
2	PKU-IDM	nofa	cascade	AV	0.922	1
3	**PRISMA**	**nofa**	**EhdRgbAud**	**AV**	**0.699**	**1**
4	CRIM-VISI	balanced	V48A66T58B	AV	0.696	1
5	CRIM-VISI	balanced	V48A66T65B	AV	0.696	1
6	CRIM-VISI	nofa	V48A66T160	AV	0.696	1
7	CRIM-VISI	nofa	V48A66T60	AV	0.696	1
8	INRIA-TEXMEX	balanced	zozo	AV	0.670	1
9	INRIA-TEXMEX	balanced	themis	AV	0.652	1
10	INRIA-TEXMEX	nofa	tyche	AV	0.652	1
11	**PRISMA**	**balanced**	**EhdRgbAud**	**AV**	**0.650**	**1**
12	ATTLabs	balanced	2	AV	0.598	1
13	**PRISMA**	**nofa**	**EhdGry**	**V**	**0.590**	**1**
14	ATTLabs	nofa	1	AV	0.580	1
15	ATTLabs	balanced	4	AV	0.567	1
16	ATTLabs	nofa	3	AV	0.552	1
17	INRIA-LEAR	balanced	deaf	V	0.542	1
18	INRIA-TEXMEX	balanced	audioonly	A	0.539	1
19	**Telefonica-research**	**balanced**	**joint**	**AV**	**0.533**	**1**
20	**PRISMA**	**balanced**	**EhdGry**	**V**	**0.490**	**1**
21	FTRDBJ	nofa	AudioOnly	A	0.483	1
22	ZJU_CS_IV	balanced	bhgccd	V	0.456	1
23	ZJU_CS_IV	nofa	bhgccd	V	0.456	1
24	KDDILabs	balanced	4sys	V	0.447	1
25	KDDILabs	nofa	4sys	V	0.447	1
26	KDDILabs	nofa	2sys	V	0.410	1
27	INRIA-LEAR	balanced	dodo	AV	0.409	1
28	INRIA-LEAR	nofa	dodo	AV	0.409	1
29	KDDILabs	nofa	base	V	0.382	1
30	ZJU_CS_IV	balanced	bgccd	V	0.374	1
31	ZJU_CS_IV	nofa	bgccd	V	0.374	1
32	BUPT-MCPRL	balanced	wsyVA	AV	0.361	1
33	BUPT-MCPRL	nofa	wsyVA	AV	0.361	1
34	FTRDBJ	nofa	orange1	AV	0.325	1
35	FTRDBJ	balanced	orange3	AV	0.314	1
36	FTRDBJ	balanced	VideoOnly	V	0.265	1
37	NTT-CSL	nofa	0	AV	0.261	1

continue in Table C.7...

Table C.6 – R_{P1} (maximum Recall with Precision 1) for all TRECVID 2011 submissions. Part 1 of 2.

#	Team	Profile	Run	Type	R_{P1}	Precision
continuation from Table C.6...						
38	NTT-CSL	balanced	1	AV	0.253	1
39	Telefonica-research	balanced	mask	AV	0.134	1
40	IMP	balanced	Uvote	V	0.132	1
41	IMP	balanced	Wvote	V	0.125	1
42	IMP	nofa	Wvote	V	0.125	1
43	IMP	nofa	Uvote	V	0.099	1
44	NTT-CSL	balanced	2	AV	0.033	1
45	NTT-CSL	balanced	3	AV	0.031	1
46	iupr-dfki	balanced	fsift	V	0.012	1
47	iupr-dfki	balanced	fsift2	V	0.012	1
48	brno	balanced	brnoccd	AV	0.002	1
49	brno	nofa	brnoccd	AV	0.002	1
–	BUPT-MCPRL	balanced	zhVideo	V	—	—
–	BUPT-MCPRL	nofa	zhVideo	V	—	—
–	ITU_MSPR	balanced	ITUMSPR2	AV	—	—
–	ITU_MSPR	nofa	ITUMSPR1	A	—	—
–	RMIT	balanced	VideoBal5	V	—	—
–	RMIT	balanced	VideoBal6	V	—	—
–	RMIT	nofa	VideoNOFA7	V	—	—
–	RMIT	nofa	VideoNOFA8	V	—	—
–	SYSU-GITL	balanced	videoonly1	AV	—	—
–	SYSU-GITL	balanced	videoonly2	V	—	—
–	SYSU-GITL	nofa	videoonly1	AV	—	—
–	SYSU-GITL	nofa	videoonly2	V	—	—
–	Telefonica-research	balanced	multimodal	AV	—	—
–	Telefonica-research	nofa	multimodal	AV	—	—
–	tokushima_U	balanced	ch4of12	A	—	—
–	tokushima_U	balanced	chcode	A	—	—
–	tokushima_U	balanced	chth	A	—	—
–	tokushima_U	nofa	ch4of12	A	—	—
–	UQMSG	balanced	mfh	AV	—	—
–	UQMSG	nofa	mfh	AV	—	—
–	USC-UTSA	balanced	test	AV	—	—
–	USC-UTSA	nofa	test	V	—	—
–	XJTU	balanced	1	V	—	—
–	XJTU	nofa	1	V	—	—

Table C.7 – R_{P1} (maximum Recall with Precision 1) for all TRECVID 2011 submissions. Part 2 of 2.

#	Team	Profile	Run	Type	$R_{P.5}$	Precision
1	INRIA-LEAR	balanced	dodo	AV	0.984	0.532
2	INRIA-LEAR	nofa	dodo	AV	0.952	0.982
3	PKU-IDM	balanced	cascade	AV	0.947	1.000
4	BUPT-MCPRL	balanced	wsyVA	AV	0.947	0.921
5	INRIA-LEAR	balanced	deaf	V	0.945	0.520
6	CRIM-VISI	balanced	V48A66T58B	AV	0.944	0.644
7	CRIM-VISI	balanced	V48A66T65B	AV	0.944	0.644
8	CRIM-VISI	nofa	V48A66T160	AV	0.944	0.644
9	CRIM-VISI	nofa	V48A66T60	AV	0.944	0.644
10	INRIA-TEXMEX	balanced	zozo	AV	0.944	0.961
11	BUPT-MCPRL	balanced	zhVideo	V	0.927	0.564
12	BUPT-MCPRL	nofa	wsyVA	AV	0.927	0.970
13	INRIA-TEXMEX	balanced	themis	AV	0.927	0.960
14	INRIA-TEXMEX	nofa	tyche	AV	0.927	0.960
15	BUPT-MCPRL	nofa	zhVideo	V	0.922	0.620
16	PKU-IDM	nofa	cascade	AV	0.922	1.000
17	**Telefonica-research**	**balanced**	**joint**	**AV**	**0.918**	**0.511**
18	NTT-CSL	balanced	3	AV	0.898	0.972
19	NTT-CSL	balanced	2	AV	0.898	0.981
20	FTRDBJ	balanced	orange3	AV	0.895	0.982
21	NTT-CSL	balanced	1	AV	0.892	0.985
22	NTT-CSL	nofa	0	AV	0.892	0.992
23	FTRDBJ	nofa	orange1	AV	0.890	0.983
24	Telefonica-research	balanced	multimodal	AV	0.876	0.530
25	Telefonica-research	nofa	multimodal	AV	0.876	0.530
26	FTRDBJ	balanced	VideoOnly	V	0.813	0.983
27	IMP	balanced	Uvote	V	0.768	0.852
28	**PRISMA**	**balanced**	**EhdRgbAud**	**AV**	**0.761**	**0.505**
29	IMP	balanced	Wvote	V	0.756	0.865
30	IMP	nofa	Wvote	V	0.756	0.865
31	IMP	nofa	Uvote	V	0.744	0.904
32	**PRISMA**	**nofa**	**EhdRgbAud**	**AV**	**0.742**	**0.640**
33	ATTLabs	balanced	2	AV	0.718	0.989
34	ATTLabs	nofa	1	AV	0.714	0.981
35	ATTLabs	nofa	3	AV	0.699	0.979
36	ATTLabs	balanced	4	AV	0.695	0.990
37	**PRISMA**	**balanced**	**EhdGry**	**V**	**0.690**	**0.607**

continue in Table C.9...

Table C.8 – $R_{P.5}$ (maximum Recall with Precision greater of equal than 0.5) for all TRECVID 2011 submissions. Part 1 of 2.

#	Team	Profile	Run	Type	$R_{P.5}$	Precision
continuation from Table C.8...						
38	INRIA-TEXMEX	balanced	audioonly	A	0.675	0.595
39	**PRISMA**	**nofa**	**EhdGry**	**V**	**0.673**	**0.784**
40	FTRDBJ	nofa	AudioOnly	A	0.601	0.991
41	Telefonica-research	balanced	mask	AV	0.574	0.599
42	iupr-dfki	balanced	fsift	V	0.565	0.667
43	KDDILabs	balanced	4sys	V	0.551	0.525
44	KDDILabs	nofa	4sys	V	0.551	0.525
45	iupr-dfki	balanced	fsift2	V	0.547	0.582
46	KDDILabs	nofa	2sys	V	0.495	0.583
47	KDDILabs	nofa	base	V	0.481	0.528
48	ZJU_CS_IV	balanced	bhgccd	V	0.456	1.000
49	ZJU_CS_IV	nofa	bhgccd	V	0.456	1.000
50	tokushima_U	balanced	ch4of12	A	0.376	0.518
51	tokushima_U	nofa	ch4of12	A	0.376	0.518
52	ZJU_CS_IV	balanced	bgccd	V	0.374	1.000
53	ZJU_CS_IV	nofa	bgccd	V	0.374	1.000
54	brno	balanced	brnoccd	AV	0.363	0.756
55	ITU_MSPR	balanced	ITUMSPR2	AV	0.329	0.912
56	ITU_MSPR	nofa	ITUMSPR1	A	0.324	0.971
57	SYSU-GITL	balanced	videoonly2	V	0.324	0.504
58	SYSU-GITL	nofa	videoonly2	V	0.299	0.586
59	SYSU-GITL	balanced	videoonly1	AV	0.270	0.502
60	SYSU-GITL	nofa	videoonly1	AV	0.262	0.537
61	tokushima_U	balanced	chth	A	0.249	0.522
62	tokushima_U	balanced	chcode	A	0.184	0.503
63	brno	nofa	brnoccd	AV	0.177	0.928
–	RMIT	balanced	VideoBal5	V	—	—
–	RMIT	balanced	VideoBal6	V	—	—
–	RMIT	nofa	VideoNOFA7	V	—	—
–	RMIT	nofa	VideoNOFA8	V	—	—
–	UQMSG	balanced	mfh	AV	—	—
–	UQMSG	nofa	mfh	AV	—	—
–	USC-UTSA	balanced	test	AV	—	—
–	USC-UTSA	nofa	test	V	—	—
–	XJTU	balanced	1	V	—	—
–	XJTU	nofa	1	V	—	—

Table C.9 – $R_{P.5}$ (maximum Recall with Precision greater of equal than 0.5) for all TRECVID 2011 submissions. Part 2 of 2.