VISTO: a New CBIR System for Vector Images

TANIA DI MASCIO    DANIELE FRIGIONI    LAURA TARANTINO

Dipartimento di Ingegneria Elettrica e dell’Informazione
Università degli Studi dell’Aquila
I-67040 Monteluco di Roio, L’Aquila - Italy
{tania, frigioni, laura}@ing.univaq.it

Abstract

In this paper, we present the main features of VISTO (Vector Image Search Tool), a new Content-Based Image Retrieval (CBIR) system for vector images. Though unsuitable for photo-realistic imagery, vector graphics are continually becoming more advanced and diffused. In fact, vector images are made up of individual, scalable objects defined by mathematical equations rather than pixels as in the case of raster images. This makes vector images fully scalable, resolution independent, not restricted to rectangular shape, allowing layering and editable/searchable text. Notwithstanding this increasing interest, all the CBIR systems proposed in the literature deal with raster images, while the research area concerning CBIR systems for vectorial images is quite new. To the best of our knowledge, VISTO is the first CBIR system for vector images proposed in the literature, and it has been designed for the retrieval of vector images in SVG (Scalable Vector Graphics) format. The contribution of this paper is twofold: we first describe the main characteristics of VISTO from the engine and the interface point of view, highlighting the differences with respect to CBIR systems for raster images known in the literature, and then evaluate VISTO’s engine from an experimental point of view within an advanced high quality 2D animation environment supporting cartoon episodes management.

Keywords: Content-based image retrieval, vector images, experimental evaluation.

1 Introduction

In computer graphics there are two types of data models: raster and vectorial (see [17] for a comprehensive discussion about these models). In the raster data model an image consists of a matrix of individual pixels, each characterized by its color and brightness. The main parameter of a raster image (also called bitmap image) is the resolution, which represents the number of pixels per inch. The resolution, when combined to the size (width and height in inches), determines the total number of pixels of an image. For example, a 2 by 3 inch image with a resolution of 300 pixels per inch contains $(2 \cdot 300) \cdot (3 \cdot 300) = 540,000$ pixels. The more pixels there are in an image, the more detail the image can be displayed with. Hence, if we double the resolution of an image, then the amount of storage for that image increases of a factor of four. In the vectorial data model, the basic units of information are points, lines and polygons. Each of these units is simply composed as a set of one or more co-ordinate points; for example, a line is a collection of related points, and a polygon is a collection of related lines. Generally speaking, a vector image is a collection of curves, as for example Bezier curves (see e.g., [17]). The use of mathematical equations of these curves
enables vector graphics to be scaled and manipulated repeatedly without distortion. In addition, operations such as geometrical transformation or visualization can be performed quite fast. The nature of the vectorial data model allows to modify the resolution of an image without changing the amount of storage for that image.

While the visualization quality of vector images is better than raster images in the representation of simple draws (e.g., clipart), the visualization quality of raster images is better than vector graphic in the representation of complex images (e.g., photos). In fact, from a visualization point of view, vector images are considered as vectorial draws rather then images. For this reason, while it is easy to convert a vectorial draw into a raster image, it is very complex to realize the opposite process. In Figure 1, we show the same image in raster data model (left) and in vectorial data model (right) in order to highlight the better quality of the vectorial draw. Furthermore, zooming operations do not modify the quality of vector images as it can be verified in Figure 2, where we show the effect of zooming on a particular of the same image (a) in the vectorial (b) and the raster (c) data models. From the storage dimension point of view, generally, the vectorial data model is better than the raster; this characteristic is strictly bind to the nature of data models. Another advantage of the vectorial data model is the possibility of storing, into the vectorial structure of the vector image file, additional and different information (e.g., keyword), differently from the raster image that can contain only the classical image information.

Figure 1: the same image in raster data model (left) and in vectorial data model (right).

From the popularity point of view, until recently, raster images ruled the roost on the World Wide Web, but the growing popularity of new vectorial-based web design programs, such as Macromedia Flash, is changing this trend. Moreover, the new format SVG (Scalable Vector Graphic) [57], promises to bring vector graphics to ordinary web pages soon. The W3C organization that sets standards for HTML, has standards for the SVG format, and popular browsers such as Netscape Navigator and Internet Explorer have plug-ins that can allow readers to use SVG graphics. It is worth noting that vector images are widely used since long time in many other application domains, as for example cartoons animation.

Hence, though unsuitable for photo-realistic imagery, vector graphics are continually becoming more advanced and diffused. In fact, as above mentioned, vector images are made up of individual, scalable objects defined by mathematical equations rather than pixels. This makes vector
images fully scalable, resolution independent, not restricted to rectangular shape, allowing layering and editable/searchable text. Notwithstanding this increasing interest, the great majority of content-based image retrieval (CBIR) systems proposed in the literature deal with raster images (we summarize the main features of these systems in Section 2, while for a complete survey we refer to [54]). On the contrary, the research area concerning CBIR systems for vector images is quite new. In fact, as pointed out in [43] “to the best of authors’ knowledge, there are no proposals (neither theoretical, nor applicative) that try to solve CBIR when images are represented in a vectorial data model”.

In this paper, we make a step toward this direction and propose a new CBIR system dealing with vector images, which is called VISTO (Vector Image Search TOol) and has been preliminary studied in [26, 27, 28, 29, 30]. To the best of our knowledge, VISTO is the unique CBIR system for vector images known in the literature, and it has been designed for the retrieval of vector images in SVG format (this was initially motivated by the fact that our first application domain was an environment for the production of 2D animation supporting cartoon episodes management [55], that clearly requires the vectorial data model). The contribution of the paper is twofold: we first describe the main characteristics of VISTO from the engine and the interface point of view, and then we evaluate the engine of VISTO from an experimental point of view.

1. From the engine point of view, VISTO includes a search engine, and a feature extraction module. For retrieval purposes a vector image is discretized within VISTO to be viewed as an inertial system in which material points are associated with descriptors obtained by discretization. This leads to a representation of the vector image that is invariant to translation, rotation, and scaling. To support requirements of different application domains, the VISTO engine offers a variety of moment sets as well as different metrics for similarity computation.

2. From the interface point of view, VISTO includes a graphical interface providing query-by-

Figure 2: effect of zooming on a particular of the same image (a) in vectorial (b) and in raster (c) data model.
sketch and query-by-example interaction with query results, and analysis of result’s quality. The interface of VISTO is designed for two classes of users: application domain users and researchers in the field of multimedia. Application domain users can use both query-by-sketch and query-by-example to search collections. Researchers can test, tune, and compare moment set and metrics. The interface helps in the selection of criteria and parameters necessary to tune the system to a specific application domain.

3. From the experimental point of view, we present an accurate experimental evaluation of VISTO. The experimental evaluation of retrieval systems is a critical part in the process of continuously improving the existing retrieval metrics. While researchers in text image retrieval have long been using a sophisticated set of tools for user-based evaluation, this does not yet apply to image retrieval [18].

We validate VISTO within an advanced high quality 2D animation environment supporting cartoon episodes management. Hence, cartoonists are the application domain users. It is very common that cartoonists reuse animation scenes and frames from previous episodes into new episodes. Possibilities for scene reuse usually stem from the memory of the animators, with little or no computational aid. Efficient archival and searching of animation material is hence appropriate.

In this framework, we performed our experiments as follows. We considered four different image categories and collected 400 images in a database, 100 for each category. The result of the searching process is a ranking of database images based on a metric obtained as weighted combination of the first seven moments of the inertial system. We consider the two measures, Precision and Recall, in order to evaluate the effectiveness and the efficiency of VISTO. The effectiveness of our system is demonstrated by the fact that the behavior of Precision vs Recall curves is always descendent. Furthermore, we demonstrate that VISTO’s efficiency increases with the decrease of the recall degree. We have also studied the discriminating power of different metrics implemented in our system. The first important observation is that Euclidean metric and City Block metric produce the same results. Finally, we have highlighted that there is a functional dependence between the discriminating power of metrics and the image category.

The paper is organized as follows. In Section 2, we give a summary of the state of the art concerning CBIR systems for raster images and propose a comparison between VISTO and the CBIR systems for raster images proposed in the literature. In Section 3, we describe the application domain within which VISTO engine has been validated. In Section 4, we summarize the main features of the search engine of VISTO. In Section 5, we describe the main features of the interface of VISTO. In Section 6, we report on our experimental evaluation of VISTO. Finally, in Section 7, we draw some conclusions and highlight future research directions.

2 Related Works

In this section we give a summary on the state of the art concerning existing CBIR systems for raster images. The goal of this section is to highlight the fact that, notwithstanding we deal with the vectorial model, our system has many characteristics in common with the existing systems, while it improves some of their aspects concerning the user interface and the interaction with it.
2.1 General framework

The main common characteristic between our system and existing systems is the architecture of framework. In Figure 3, we show a simplified framework of a generic CBIR system, highlighting the aspects related to the User Interface and the Engine System.

The Users Interface, and in particular the Query Interface Module, supports users in the query process; the user formulates a query, drawing an image or choosing an existing one, and she/he selects the database of images to be used as a search repository. Images are processed by the CBIR Engine that calculates the similarity ranking shown by the Query Result Module of the User Interface.

The Engine System is the main component of the overall framework. The goal of this component is to extract features from the query image and the database images, to represent visual features images, and to calculate comparison between query image representation and database images representation. As shown in Figure 3, in the CBIR Engine System we identify different components, in particular:

- The Feature Process component extracts visual features (for example, color, texture, and shape) from images using the Feature Extraction Module, and it creates, by the Feature Representation Module, a representation of visual features (for example, color histograms, shape vectors).

- The Comparison Process component computes images similarity by the Similarity Computation Module using retrieval techniques (for example, feature distance), and it creates by the
Ranking Module a ranking vector representing the query image result.

- The Indexing Repository contains the set of visual features descriptors of the database images (for example, set of color histograms, set of shape vectors).

- The Query Image Indexed contains the visual features descriptor of the query image (for example, one color histogram, one shape vector).

For a more detailed description of the general framework of CBIR, the reader should refer to [15].

2.2 Survey results

In the last years, a large amount of CBIR systems for raster images have been proposed in the literature (for a complete survey, see [54]). Most of them are accessible through the Web and provide demos to check their characteristics. We have collected the majority of the systems with demos and studied their main components: the Engine System and the User Interface. Table 1 summarizes the main features of the collected CBIR systems, and consists of nine columns highlighting General Aspects (Columns 1–3), Engine Aspects (Columns 4–6), and Interface Aspects (Columns 7–9). The General Aspects includes the name of the system, the data model supported and the application domain.

Aspects related to the Engine System are represented in the columns called:

- Visual Features (shape, color, and texture),
- Feature Representation (e.g., color histogram [35, 50]),
- Retrieval Techniques (e.g., feature distance [8]).

Aspects related to the User Interface are reported in Columns 7–9 using the following technical characteristics:

- Interface type (web-like or windows-like),
- Query type (ex = query by example or im = query by image),
- Draw window (yes or no).

All the CBIR systems listed in Table 1 support the raster model (see Column 2). This depends on the fact that, at the moment, the main application domain (Column 3) of the CBIR is the search of photos, and that the data model more suitable for the representation of photos is the raster model. Moreover, by observing the characteristics of the CBIR systems listed in Table 1, we can deduce many important considerations which we present categorized into engine system aspects and user interface aspects.

Engine system aspects. As mentioned in Section 1, we focus on aspects related to visual feature, feature representation and retrieval techniques (Column 4, 5, and 6 in Table 1). In the most recent survey on CBIR, Wang [8] writes “most systems perform features extraction as a pre-processing step, obtaining global image features like color histogram or local descriptors like shape and texture.”

The color is the mostly used visual feature in CBIR systems (Column 4 in Table 1). The motivation comes from the application domain; in fact, in the search of photos, the color is much more descriptive than the shape and the texture, which can be easily derived from the color. Several color features representation techniques have been applied in Image Retrieval, notably,
<table>
<thead>
<tr>
<th>Name</th>
<th>Data model</th>
<th>Application domain</th>
<th>Visual Features</th>
<th>Feature Representation</th>
<th>Retrieval Techniques</th>
<th>Interface type</th>
<th>Query type</th>
<th>Draw window</th>
</tr>
</thead>
<tbody>
<tr>
<td>AltaVista [6]</td>
<td>raster</td>
<td>Internet image search</td>
<td>sh, co, tx</td>
<td>color vector</td>
<td>HLS of feature relevance</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
</tr>
<tr>
<td>Amore [34]</td>
<td>raster</td>
<td>Arthur - art retrieval</td>
<td>sh, co</td>
<td>color blob, color</td>
<td>square distance, Euclidean distance</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
</tr>
<tr>
<td>C-bird [23]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>color vector</td>
<td>Euclidean distance</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
</tr>
<tr>
<td>DrawSearch [44]</td>
<td>raster</td>
<td>Internet image search</td>
<td>sh, co, tx</td>
<td>color vector, shape</td>
<td>weight distances</td>
<td>windows-like</td>
<td>im</td>
<td>yes</td>
</tr>
<tr>
<td>ExLIB [9]</td>
<td>raster</td>
<td>Internet image search</td>
<td>sh, co, tx</td>
<td>feature distances</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Excalibur [10]</td>
<td>raster</td>
<td>Yahoo! and Infoseek search engines</td>
<td>co</td>
<td>brightness distance</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>FOCUS [7]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>region of color</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>ImageFinder [11]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>Boltzman machine</td>
<td>windows-like</td>
<td>im</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>ImageRetro [39]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>Minkovsky metric</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>ImageRover [40]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co, tx</td>
<td>segmentation</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>ImageScape [22]</td>
<td>raster</td>
<td>Internet image search</td>
<td>sh, co, tx</td>
<td>color histogram</td>
<td>Gauss filter</td>
<td>ex</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td>IRS? [20]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>MDS color</td>
<td>Euclidean distance</td>
<td>web-like</td>
<td>ex, im</td>
<td>no</td>
</tr>
<tr>
<td>MARS [36]</td>
<td>raster</td>
<td>Fowler Museum of Cultural History</td>
<td>sh, co, tx</td>
<td>color histogram</td>
<td>weight Euclidean distance</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
</tr>
<tr>
<td>MetaSeek [32]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>distance feature</td>
<td>web-like</td>
<td>ex, im</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>MIR [33]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>Euclidean distance</td>
<td>web-like</td>
<td>im</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>PicToSeek [14]</td>
<td>raster</td>
<td>Internet image search</td>
<td>sh</td>
<td>color vector</td>
<td>distance feature</td>
<td>web-like</td>
<td>ex, im</td>
<td>no</td>
</tr>
<tr>
<td>Pic2Seek [37]</td>
<td>raster</td>
<td>Internet image search</td>
<td>sh</td>
<td>color vector</td>
<td>distance feature</td>
<td>web-like</td>
<td>ex, im</td>
<td>no</td>
</tr>
<tr>
<td>QBIC [38]</td>
<td>raster</td>
<td>Hermitage - art retrieval</td>
<td>sh, co, tx</td>
<td>color layout</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>Sale [12]</td>
<td>raster</td>
<td>Internet image search</td>
<td>region of color</td>
<td>join of distance feature</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>SIMBA [45]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>distance feature</td>
<td>web-like</td>
<td>ex</td>
<td>im</td>
<td>no</td>
</tr>
<tr>
<td>Simplicity [46]</td>
<td>raster</td>
<td>Internet image search</td>
<td></td>
<td>distance feature</td>
<td>web-like</td>
<td>ex, im</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>VisualSeek [56]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td>color relevance</td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
<tr>
<td>WebSeek [59]</td>
<td>raster</td>
<td>Internet image search</td>
<td>co</td>
<td></td>
<td>web-like</td>
<td>ex</td>
<td>no</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Main characteristics of CBIR systems for raster images.
color histograms (see, e.g., [35, 50]), color sets (see, e.g., [48]), and Color Moments (see, e.g., [49]). In particular, in the moment-based approach, any color distribution can be characterized by its moments, with most of the information concentrated in the first, second, and third central moments (mean, variance, and skewness).

The shape, as visual feature, is generally required to be invariant to translation, rotation, and scaling. The shape features representation techniques can be classified into two categories: boundary-based and region-based. The former uses only the outer boundary of the shape while the latter uses the entire shape region. The most successful representatives for these two categories are Fourier Descriptor and Moment Invariants [19, 41] (we will further discuss on the use of moments in Section 4 and in the Appendix). The main idea of Fourier Descriptor is to use the Fourier transformed boundary as the shape feature. For example, Rui et al. [41] propose a modified Fourier Descriptor that is both robust to noise and invariant to geometric transformation. The main idea of Moment Invariants is to use region-based moments, which are invariant to transformations, as shape feature are. Based on [19] many improved versions appeared in the literature during the course of the years. Yang and Albrecht [61] propose a fast method of computing moments in binary images. In [21] algorithms are discussed to systematically generate and search for a given geometry’s invariants. In some recent review papers (see, e.g., [1]), the performance of boundary based representations, region based representations, and combined representations are compared. Their experiments show that combined representations outperform the simple representations.

Finally, the texture, as visual feature, contains important information about the structural arrangement of surfaces and their relationship to the surrounding environment. Currently, most texture feature representation techniques are based on the wavelet transform, originally proposed in [3]. For example, in [47], the authors use, as texture representation, statistics measures (mean and variance) extracted from the wavelet sub-bands. In [25] several kinds of wavelet transform are used to evaluate the texture image annotation.

Once a decision on the visual feature extraction and the feature representation techniques has been made, how to steer them towards accurate image retrieval is the next issue. A large number of retrieval techniques, as for example the feature distance [8] have relied on the assumption that the image feature space is a manifold embedded in Euclidean space. Clustering, as retrieval techniques, has been applied to image retrieval in order to help improve interface design, visualization, and result pre-processing [4]. A statistical approach involving the well known Wald-Wolfowitz test [58] for comparing non-parametric multivariate distributions has been used as retrieval techniques, it is applied for color image retrieval [51], representing images as sets of vectors in the RGB-space. Finally, a number of probabilistic approaches have been proposed in the last few years as retrieval techniques. The idea is to integrate feature selection, feature representation, and similarity measure into a combined Bayesian formulation, is used to minimize the probability of retrieval error (see [53] for more details).

User interface aspects. We define the terms web-like and windows-like to categorize the Interface type (Column 7 in Table 1). While the web-like interface (see, e.g., Blobworld [2] and MARS [36]) strictly follows the web environment, the design of a Windows™ interface (see, e.g., ImageFinder [11] and Netra [24]) reproduces the classical windows application. Most of the systems studied support a web-like interface. This is motivated by the fact that the great majority of the CBIR systems realized up to now is dedicated to the search of images over the Web. This has led to the creation of interfaces very similar to the Web search engines interfaces.

The supported Query type (Column 8 in Table 1) is mainly the query by example, where the user aims at extracting images of a given category (in other words the retrieval system should
discriminate among categories). The query by image responses to a query typically formulated by an example drawing; using this type of query formulation the user aims at find similar images (partially identical image, or a partially identical object in the image) in the image repository. In order to support the query by example formulation, most of the systems (among others C-Bird [23]) put in the main interface a series of images representing different categories of search. The search is started after the selection of one of the categories. Examples of systems that support both query by example and query by image are AltaVista Photofinder [6], Amore [34], and PictoSeek [14].

An Input Draw window (Column 9 in Table 1) represents the possibility to draw an image as input of the search process. This characteristic is given only by two of the considered systems: DrawSearch [44], and ImageScape [22]. Again the application domain motivates this aspect. In fact, if the search of photos is the main concern, then the necessity to draw is not very important.

**Discussion.** Besides the use of raster images, the systems of Table 1 have other common characteristics, and in particular:

- the lack of a feature extraction module directly accessible from the interface;
- the lack of a module for the visualization of graphics concerning the results of the performed operations;
- the lack of a software module dedicated to tune the system.

All these aspects highlight the necessity to have a complete interaction with users in order to improve the precision and the recall of the overall system; the interaction with users is the main aspect that we considered in the design of our project.

**2.3 VISTO design choices**

In what follows we summarize the main characteristics of VISTO, the system proposed in this paper, and compare them with systems listed in Table 1. In particular, we first describe the common characteristics of VISTO and the systems of Table 1, and then we highlight the differences.

- **Common characteristics**
  - Our system is bound to the application domain as a large number of the considered CBIR systems, but the modules are easily extendable to other contexts.
  - Our system uses, as feature representation, moments containing visual feature of images, as a large number of the considered CBIR systems.

- **Differences**
  - Our system deals with vectorial images; in fact, as above mentioned, our application domain is the cartoons animation that requires the vectorial format.
  - Our system uses the shape as the main feature and not the color, as in many of the considered systems; in fact, in the context of cartoons animation the shape is much more important and representative than the color, and also it well identifies objects.
  - Our system gives the possibility to set retrieval techniques (metrics or moments set) by interacting with the interface and by choosing among five different metrics and three sets of moments. None of the systems of Table 1 gives this opportunity.
– Our system allows to perform query by image and query by example, and introduces also a semantic level for the search. Most of the systems of Table 1 do not give this opportunity.
– The interface of our system is windows-like; this choice was motivated by the fact that our system is a desktop application and not a web application.
– The interface of our system has a dedicated software for the vectorial draw; in fact in our application context, the presence of a draw area is so basic for the cartoonist in the searching process of scenes already existing in the database, that we decided to introduce them in the input panel.

3 Application Domain

Concerning the application domain, the proposed CBIR engine has been validated in an advanced high quality 2D animation environment supporting cartoon episodes management. In this case application domain users are represented by cartoonists. It is very common that cartoonists reuse animation scenes and frames from previous episodes into new episodes. Possibilities for scene reuse usually stem from the memory of the animators, with little or no computational aid. Efficient archival and searching of animation material is hence appropriate.

The creation of cartoons is based on two fundamental aspects: episode realization and animation. While the former aspect is related with the authors skill, the latter, nowadays, is related to a specific technology. In general, authors prefer to use traditional paper for realization, because the new technologies do not assure them the artistic signature, thus not guaranteeing high quality artistic drawing. To solve this problem and to offer an environment as close as possible to the natural one, the system in [55] supports an interactive slate. This input device is composed by a digital display and an electronic, pressure sensitive, pen. This technology, based on new intelligent pen systems, protects every natural element of a paint realized in the traditional environment.

Generally, to create an episode it is necessary to animate a large number of scenes, in turn composed by a number of frames. The traditional realization of a scene utilizes an appropriate device, called rostrum (see Figure 4).
A single frame is shot by a camera perpendicular to a series of parallel transparent trays, each carrying a slide with one or more objects of the frame (e.g., background, characters, etc). Therefore a single scene is composed by one fixed background and one or more animated characters. Each of these animated characters is formed by a fixed part (the profile of the character) and a moving part (the face of the character) to simulate, for example, talk actions. Hence images in our domain often possess few characteristics, and often belong to well-defined categories (background, characters, faces, etc). Therefore we identify four categories that well represent our application domain (see Figure 5): background (BK), characters or persons (PE), faces (FA), and not classified (NC) for different objects.

Cartoonists may want to retrieve an image by providing a sketch of it, or to retrieve images similar to an example one, or to retrieve images of given categories. It is therefore appropriate that the retrieval system be able to answer two types of query of two level of abstractions: while the level 1 comprises retrieval by primitive features (such as shape), in response to a query formulated by an example drawing (query by image), the level 2 comprises retrieval by logical features, aimed at extracting images of a given category (query by example); in other words, the retrieval system should discriminate among categories.

4 The Engine: our proposal

In this section we summarize the main features of the search engine of VISTO (a preliminary version of the engine was presented in [26, 30]). The dataflow scheme of our CBIR engine proposal is depicted in Figure 6, and it is similar to the generic framework described in Section 2. Given a query image, database images are ranked based on the similarity with the input image, so that more relevant images are returned first in the query result. The processing hence requires a Feature Process to associate images with descriptors representing visual features, and a Comparison Process to evaluate distances between descriptors (the similarity between any two images is computed as the similarity between the two corresponding descriptors). It is worth remember that the purpose of image processing in image retrieval is to enhance aspects in the image data relevant to the query,
and to reduce the remaining aspects.

Figure 6: the system architecture.

Without loss of generality, our engine deals with shape extraction [13], since shape adequately identifies and classifies images typical of the application domains so far considered (treatment of other visual features is however a direct generalization of the shape case). Moreover, the shape representation is required to be invariant to translation, rotation, and scaling. These affine transformations are to be regarded as applied to a selected point belonging to the image and representative for the image.

Our approach is to consider the image like an inertial system and to use the center of mass as selected point. The inertial system is obtained by the Feature Extraction Module by discretizing the vectorial image, and by associating material points with basic elements obtained by the discretization process. The origin of the inertial system is then moved to the center of mass (formulas (1) and (2)), to which transformation can be applied (see Figure 7).

$$
x_{CM} = \frac{\sum_{i=0}^{n} m_i x_i}{\sum_{i=0}^{n} m_i}
$$

$$
y_{CM} = \frac{\sum_{i=0}^{n} m_i y_i}{\sum_{i=0}^{n} m_i}
$$

For a generic material point $i$, in formulas (1) and (2) $m_i$ represents the mass of $i$, that is the length of strokes obtained after the discretization process, and $(x_i, y_i)$ represents the coordinates of $i$.

Once an image has been transformed into an inertial system, the natural way to represent image shape is to exploit the inertial systems characteristics, which provide useful information about the image. In fact, the first four central moments are synthesis index of distribution as highlighted in second column of Table 2 (see Section A.1 in the Appendix for the computation of the moments).

In our context, the average of the inertial system represents the dimension of image: low average means image poor in strokes, high average means image rich in strokes. The variance of the inertial system represents how image center of mass area is composed: low variance means image center of
Figure 7: creation of the inertial system; (a) initial vector image, (b) discretized image, (c) image as inertial system

<table>
<thead>
<tr>
<th>Moment Order</th>
<th>Moment Name</th>
<th>Moment Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>first order</td>
<td>distribution average</td>
<td>image “dimension”</td>
</tr>
<tr>
<td>second order</td>
<td>distribution variance</td>
<td>“type” of center of mass area</td>
</tr>
<tr>
<td>third order</td>
<td>distribution skew</td>
<td>symmetry of image</td>
</tr>
<tr>
<td>fourth order</td>
<td>distribution kurtosis</td>
<td>“composition type” of image</td>
</tr>
</tbody>
</table>

Table 2: moments description

mass area poor in strokes, high variance means image center of mass area rich in strokes. The skew of the inertial system represents the symmetry of images: high value of skew means low symmetry of image, low value of skew means high symmetry of image. Finally, kurtosis of the inertial system represents how image is composed, high kurtosis means image poor in empty areas, low kurtosis means image rich in empty areas.

In conclusion, we can consider the first four invariant central moments and in general all invariant central moments and the eccentricity and the axes, representative for the inertial system (formulas of eccentricity and axis are given in Section A.1); then feature descriptors, created in the Feature Representation Module, are vectors containing values of invariant central moments. In the literature, different invariant central moments sets have been proposed [52], differing in the way the moments are computed (see, e.g, [60, 61]). At the present implementation stage, our engine includes the following moments set:

- Hu moments [19]
- Zernike moments [5]
- Bamieh moments [61]
The detailed mathematical descriptions of the three moment sets are given in Sections A.2, A.3 and A.4.

Concerning the similarity computation, different metrics have been proposed in the literature [16]. All metrics need a vector of coefficients to adequately weigh the individual values of descriptor vectors. Our CBIR engine can be tuned to the discrimination requirements of a selected application domain by choosing a moment set and a metric (along with a vector of weights) among those included in the system. At the present implementation stage, our engine includes the following metrics:

- Chebyshev distance (CH) [61]
- City Block distance (CY) [61]
- Cross Correlation distance (CC) [5]
- Discrimination Cost distance (DC) [5]
- Euclidean distance (EU) [61]

For further details and formulas about metrics see Section A.5.

It has to be noticed that a certain degree of uncertainty is typical of similarity search, and, in particular, two types of errors may occur:

- Images that do not exactly satisfy the user’s query are returned in the query result;
- Images that do satisfy the user’s query are not returned in the query results.

Engine tuning hence requires quality result indicators to establish the effectiveness of a search, i.e., to evaluate, on one hand, how well the engine satisfy the domain requirements, and, on the other hand, the extent of the retrieval process errors. Let us denote as $N_{Rel\_Im\_Ret}$ the number of returned relevant images, as $N_{Rel\_Im}$ the total number of relevant images in the collection, and as $N_{Im\_Ret}$ the total number of images returned. The following two quantities are often used to measure the effectiveness of a search [16]:

- **Precision** = $N_{Rel\_Im\_Ret} / N_{Im\_Ret}$
- **Recall** = $N_{Rel\_Im\_Ret} / N_{Rel\_Im}$

In other words, Precision is the fraction of the retrieved documents which is relevant and Recall is the fraction of the relevant images which has been retrieved. Obviously, the relevant images are images belonging to the same category. In Section 6, we will use these two measures to evaluate the retrieval quality of our system.

5 The Interface: our proposal

The interface of VISTO was designed to help two different types of users described in Section 1: the application domain users represented by cartoonists and researchers in the field of multimedia represented by engineers. The main task of the cartoonist is retrieving an image by providing a sketch of it, or retrieving images similar to an example one, or retrieving images of given categories. The main task of engineers is obviously tuning the engine in an interactive way, based on system feedback.
To support the different tasks of these types of users, the interface includes a structured main window and chart viewers. The former, always displayed on the screen, is used to handle user’s input actions and to visualize results and a first system feedback. The latter are invoked to visualize additional system feedback to favor an in-depth tuning analysis. Then, while cartoonists use only the main window, engineers use both main window and chart viewers.

![Figure 8: the main window of VISTO’s interface.](image)

**The main window.** The main window of the interface is partitioned into three display panels, spatially organized into three rows (see Figure 8):

1. The upper panel accepts user input actions, and it is in turn divided into three panels: from left to right, the image processing request panel, the parameters setting panel, and a tool palette, respectively.

   (a) The image processing request panel requires the user to first provide an image, either by sketching it (clicking on the Draw image button), or by specifying a file containing it (clicking on the Choose image button), and then to activate the desired function: the Search Engine or the Feature Extraction functions. While cartoonists may want to use the input image as an example image for a search, engineers may want to use the input image for a search, or for an in-depth analysis (i.e., the engineers want to classify it with the support of the system).

   (b) The parameter setting panel accepts input actions to set parameters of either the Search Engine function or the Feature Extraction function.
If the user chooses the Search Engine function, s/he needs to specify which set of moments is to be used as descriptor vector, which weights to be assigned to the chosen set, and which metrics are to be applied in the similarity computation. The rank visualized in the middle panel is that related to the choice made in the column thumbnails of the parameter setting panel.

If the user chooses the Feature Extraction function, s/he needs only to specify which sets of moments are to be used as descriptor vectors (more than one set can be chosen). Typically, the cartoonist is not interested in the use of this panel, which is instead fundamental for the engineer, whose main purpose is the tuning process of the engine.

(c) The tool palette shows three buttons: clicking on the first the search engine process or the feature extraction process starts; clicking on the Back button all processes are reset; clicking on the Exit button, the system is closed.

2. The middle panel displays the query image selected and a scrollable list of the images retrieved due to a query, ranked by similarity as discussed in Section 4, along with progression bars giving echoes of the system processing. The panel is also used by users (cartoonists and engineers) to provide relevance feedback. Images in the list may be selected (clicking twice on it) to be provided as target image in a new search, in an incremental querying process.

![Figure 9: the chart viewers: a log file ranking.](image)

3. The lower panel, mainly used by engineers, provides an initial indication on the search effectiveness, by displaying, both in tabular and in chart form, figures about Precision and Recall and obtained with the selected metrics. The tools in the palette on the right side of
the panel are used to invoke interactive log tables and charts (displayed by the *chart viewers*) visualizing more detailed and less aggregated data, which favor an in-depth analysis of the image processing, by engineers.

**Chart viewers.** As discussed in Section 2, in order to evaluate the Precision and Recall measures, it is useful that the engineers could inspect detailed log data (see Figure 9) that account for the role of moments, metrics and weights in the similarity computation. Several charts can be visualized and manipulated, e.g., to make comparisons among moment sets (see Figure 10), or to appreciate the discriminating power of individual moment in a given set. For example, Figure 11 depicts the values of the first order of the Hu’s moment set for a small-sized sample database: peaks in the chart correspond to images of the background category, thus highlighting the role of the first order moment in the identification of background images. Information of this type are useful to determine adequate weight vectors to efficiently answer queries of level 2 (logical level, see Section 3).

### 6 Experiments and Discussions

The experimental evaluation of retrieval systems is a critical part in the process of continuously improving the existing retrieval metrics. While researchers in text image retrieval have long been using a sophisticated set of tools for user-based evaluation, this does not yet apply to image retrieval [18]. In this section we present an experimental study of the VISTO’s engine.
For the present study we use the two measures, mentioned in Section 4, **Precision** and **Recall** in order to evaluate the **effectiveness** and the **efficiency** of our system and to examine the **discriminating power** of implemented metrics (Euclidean, Discrimination Cost, City Block, and Cross Correlation).

### 6.1 Experiments

We conducted a preliminary experiment in order to discover the effectiveness and the efficiency of our system and to examine the discriminating power of implemented metrics. We then fixed the Hu’s set as moments set and we assigned to each moment of Hu’s set the weight \([1, 1, 1, 1, 1, 1, 1]\), to test the system in the worst case.

We tested our system on a 933MHz CPU Pentium III PC with 256MB RAM. We use images falling in the four different categories BacKground, PErson, FAces, and Not Classified, described in Section 3 and denoted from now on as BK, PE, FA, and NC, respectively. We use a database, from now on denoted as \(DB\), of 400 images, 100 for each category.

We performed experiments by submitting to our CBIR system a set of nine queries (see Table 3) on images randomly selected from \(DB\). We say that an image \(j\) of \(DB\) is **relevant** for a query \(Q\) on image \(i\), denoted as \(Q(i)\), if and only if \(j\) and \(i\) belong to the same category. For instance, for query 1 in Table 3 (\(Q(BKTest5)\)), the relevant images are all the images in \(DB\) belonging to the BK category. Further, given a query \(Q(i)\), we denote as \(RQ(i)\) the set of images of \(DB\) relevant to \(Q(i)\).

As a result to a query \(Q(i)\), the system **produces a set**, denoted as \(AQ(i)\), containing all images.
Table 3: list of query images.

<table>
<thead>
<tr>
<th>Query</th>
<th>Image’s Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BKTest5</td>
<td>BacKground</td>
</tr>
<tr>
<td>2</td>
<td>BKTest6</td>
<td>BacKground</td>
</tr>
<tr>
<td>3</td>
<td>PEdog1</td>
<td>PErson</td>
</tr>
<tr>
<td>4</td>
<td>PEpotato1</td>
<td>PErson</td>
</tr>
<tr>
<td>5</td>
<td>FAbye</td>
<td>FAce</td>
</tr>
<tr>
<td>6</td>
<td>FAface4</td>
<td>FAce</td>
</tr>
<tr>
<td>7</td>
<td>NCflower</td>
<td>Not Classified</td>
</tr>
<tr>
<td>8</td>
<td>NCrain</td>
<td>Not Classified</td>
</tr>
<tr>
<td>9</td>
<td>NCcircel</td>
<td>Not Classified</td>
</tr>
</tbody>
</table>

of DB ranked by similarity with respect to \(i\); in other words, the first image of the set \(A_{Q(i)}\) is the most relevant image for the query \(Q(i)\). The system shows the firsts images of \(A_{Q(i)}\), in customary descendant order, but it does not necessarily show all images of \(A_{Q(i)}\).

Given a query \(Q(i)\) and his set of results \(A_{Q(i)}\), we define:

- \(Sh_{Q(i)}\) as the set of first elements of \(A_{Q(i)}\) shown by the system;
- \(R_{deg}(Sh_{Q(i)})\) as the recall degree of \(Q(i)\), that is the percentage of images of the set \(A_{Q(i)}\) that the system shows; formally:

\[
R_{deg}(Sh_{Q(i)}) = \left(\frac{|Sh_{Q(i)}|}{|A_{Q(i)}|}\right)\%.
\]

For instance, given a query \(Q(i)\), if \(Sh_{Q(i)}\) is 200 (in other words the system shows the first 200 elements of \(A_{Q(i)}\)), then the recall degree of \(Q(i)\) is 50%.

Furthermore, as seen in Section 4, given a query \(Q(i)\), the notions of Precision and Recall of \(Q(i)\), are hence the following:

- \(\text{Precision}_{Q(i)} = \frac{|R_{Q(i)} \cap A_{Q(i)}|}{|Sh_{Q(i)}|}\)
- \(\text{Recall}_{Q(i)} = \frac{|R_{Q(i)} \cap A_{Q(i)}|}{|R_{Q(i)}|}\)

These formulas highlight that the Precision vs Recall curves depend from \(|R_{Q(i)}|\) and \(|Sh_{Q(i)}|\). In our experiments, while the value of \(|R_{Q(i)}|\) is fixed (it is always equal to 100, because, as above mentioned, our DB is composed of 100 images per category), the value \(|Sh_{Q(i)}|\) and then the value \(R_{deg}(Sh_{Q(i)})\) are not fixed. In fact, as already mentioned, they depend from the system. For our experiments, we depicted three sets of Precision vs Recall curves, in particular:

1. set I: the Precision vs Recall curves for each of the nine queries in Table 3 and for each metric of our system, fixing \(R_{deg}(Sh_{Q(i)}) = 100\% (|Sh_{Q(i)}| = 400)\);
2. set II: the Precision vs Recall curves for each of the nine queries in Table 3 and for each metric of our system, fixing \(R_{deg}(Sh_{Q(i)}) = 50\% (|Sh_{Q(i)}| = 200)\);
3. set III: the Precision vs Recall curves for each of the nine queries in Table 3 and for each metric of our system, choosing \(|Sh_{Q(i)}| = t\), where \(t\) is the position in \(A_{Q(i)}\) of the unique element of \(A_{Q(i)}\) such that the Recall\(_{Q(i)}\) is 50%; in this case \(R_{deg}(Sh_{Q(i)})\) is variable.
Each of these sets is composed of $9 \times 5 = 45$ curves (5 metrics per 9 queries), where each curve is the representation of a sample of Precision and Recall values, obtained by fixing the query $Q(i)$ and the metric $M$; this sample is denoted as $S_{Q(i)}(M)$.

Before conducting experiments on sets I, II, and III, we have performed a preliminary ANOVA (ANalysis Of Variance) test [12] to statistically estimate the consistency of samples $S_{Q(i)}(M)$. After that, we have conducted the following experiments:

1. we first evaluated the retrieval effectiveness of our system, studying the behavior of the Precision vs Recall curves of set I; the results of these experiments are shown in Figures 12 and 13;

2. we conducted a critical study of the retrieval efficiency of our system making an in depth analysis of the Precision vs Recall curves of sets II and III; the results of these experiments are shown in Figure 14;

3. we studied the discriminating power of metrics using the values of Precision and Recall obtained in experiments. The resulting histograms are shown in Figure 15.

The same experiments have been performed also on a database of 2000 images, where each image has been assigned to one of the 4 considered categories. The size of the categories varies from 220 to 1020. We have noticed that the results are basically the same of the previous set of experiments, and hence we do not report on them.

6.2 Discussion

The results of the preliminary ANOVA test, carrying out from the samples used, meet the expectation, by showing that samples $S_{Q(i)}(M)$ are statistically consistent (for more details see [31]). These results allow us to study the effectiveness and the efficiency of our system and to conduct an in-depth study about the discriminating power of metrics.

To verify the effectiveness of our system, we evaluated the retrieval performance of all metrics $M$ over all queries $Q(i)$ in Table 3 averaging the Precision of all metrics at each Recall level of set I. Graphic (a) of Figure 12 depicts the results: analyzing these curves we can observe their descendent behavior; as well described in [18] this behavior demonstrate the effectiveness of our
Figure 13: Precision vs Recall for each category; (a) Precision vs Recall of Euclidean Distance for the Q1 and Discrimination Cost for Q2; (b) Precision vs Recall of Euclidean Distance for the Q3, Q4; (c) Precision vs Recall of Euclidean Distance for the Q5, Q6; (d) Precision vs Recall of Euclidean Distance for the Q7, Q8, Q9.

Furthermore, in order to study the quality of the effectiveness for each category, we calculated the polynomial interpolation curves, represented by the dashed lines in Figure 13; studying the behavior of these curves, we observe a concavity change around the 60% of Recall in curves depicted in graphics (b) and (d). This shows that images from the PE and NC categories are in some sense more difficult to retrieve, at least for our system when the degree of recall is 100%.

We further investigated this phenomenon by conducting an in depth analysis using sets II and III. Figure 14 depicts the results: it is worth noting that there are no concavity change in the graphics (b), (c), and (d) and in the graphic (a) the concavity change is shifted around the 70% of Recall. No concavity changes in these graphics mean a complete descendent behavior of curves considered and this result confirms our hypothesis: the efficiency of our system depends from the recall degree. Therefore we can assume that our CBIR system increases in efficiency with the decrease of $R_{deg}$.
Figure 14: Precision vs Recall for the PE and NC categories; (a) Precision vs Recall (set II) of Euclidean Distance for the Q3, Q4; Precision vs Recall (set II) of Euclidean Distance for the Q7, Q8, Q9; (c) Precision vs Recall (set III) of Euclidean Distance for the Q3, Q4; (d) Precision vs Recall (set III) of Euclidean Distance for the Q7, Q8, Q9. In these graphics we highlighted the $|\text{Sh}_Q(i)|$.

Finally, consulting histograms in Figure 15 representing average values of Precision at different level of Recall for each category, we can study the discriminating power of different metrics implemented in our system. The first important observation is that: Euclidean metric and City Block metric produce the same results, furthermore:

- BK category: the Discrimination Cost offers a great performance in both the cases depicted in Figure 15; others metrics have approximatively the same discriminating power;
- PE category: while the Euclidean metric and the City Block metric produce the better results, the lower discriminating power is performed by the Cross Correlation metric;
- FA category: the Euclidean metric, and the City Block metric, produce the better results;
- NC category: both in the case depicted in Figure 15, there are no significant differences in discriminating power of different metrics performed.
In conclusion, by the above described experiments we have demonstrated that the engine of VISTO works well, and we proved the effectiveness and the efficiency of our system, especially if the degree of recall is a value around the 50%, which increases if $R_{deg}$ decreases. Furthermore, we have discovered an equivalence in performance between the Euclidean and the City Block metrics. Finally, we have highlighted that there is a functional dependence between the discriminating power of metrics and the image category.

7 Conclusions and future works

In this paper we presented the main features of VISTO, a CBIR system designed for the retrieval of vector images in SVG format. To the best of our knowledge, VISTO is the first Content-Based Image Retrieval system for vector images proposed up to now in the literature. We have first described the main characteristics of VISTO from the engine and the interface point of view, highlighting the differences with respect to CBIR systems for raster images known in the literature. In particular, while sharing with other systems the general architecture framework and the use of moments as feature representations, VISTO differs in the following aspects:

- Use of the shape as main feature.
- Possibility to interactively set the retrieval techniques (metrics or moments set) to tune the engine for different application domains (none of the studied systems gives this opportunity).
- A modality of search at semantic level, in addition to query by image and query by example (most of the studied systems do not give this opportunity).

Then we have evaluated the VISTO engine from an experimental point of view within an advanced high quality 2D animation environment supporting cartoon episodes management. In particular, we proved that:

- The VISTO engine guarantees effectiveness and the efficiency of retrieval.
- There is an equivalence in performance between the Euclidean and the City Block metrics.
- There is a functional dependence between the discriminating power of metrics and the image category.
The following research directions deserve further investigation:

- The evaluation of VISTO within different application domains, with the aim of testing the retrieval efficiency and the discriminating power of the prototype in other domains.
- Re-engineering VISTO with the aim of designing a web application.

References


[33] MIR. http://www.cedar.buffalo.edu/MMIR.


A Appendix: formulas

In this appendix we define the formulas for concepts introduced in Section 4. Our engine deals with shape extraction [13], since shape adequately identifies and classifies images typical of the application domains. Moreover, the shape representation is required to be invariant to translation, rotation, and scaling. These affine transformations are to be regarded as applied to a selected point belonging to the image and representative for the image. Our approach is to consider an image like an inertial system and to use the center of mass as selected point. The inertial system is obtained in the Feature Extraction Module discretizing the vectorial image, and associating material points with basic elements obtained by the discretization process. The origin of the inertial system is then moved to the center of mass, to which transformation can be applied.

A.1 The inertial system

An inertial system is composed of material points (in other word, an inertial system is a distribution of material points); the center of mass is calculated using (3) and (4), where, for a generic material point \( i \), \( m_i \) represent the mass of \( i \) and \((x_i, y_i)\) represent coordinates of \( i \).

\[
x_{CM} = \frac{\sum_{i=0}^{n} m_i x_i}{\sum_{i=0}^{n} m_i} \tag{3}
\]

\[
y_{CM} = \frac{\sum_{i=0}^{n} m_i y_i}{\sum_{i=0}^{n} m_i} \tag{4}
\]

Once defined an image as an inertial system, the natural way to represent image shape is to exploit the inertial systems characteristics, which provide useful information about the image. We use inertial moments set as image feature descriptors. The moments set considered are the Hu’s moment set, Zernike’s moment set, and Bamieh’s moment set. In order to calculate these moment’s set values we may introduce invariant central moment concept.

We say that for a collection of \( n \) material points, \( m_{p,q} \) represents the invariant central moment of \( pq \) order; it is calculated using (5) where \( x \) and \( y \) represent the coordinates of \( n \) material points and \( x_{CM} \) and \( y_{CM} \) represent the center of mass coordinates calculated above.

\[
m_{p,q} = \sum_{p,q} (x - x_{CM})^p \cdot (y - y_{CM})^q \tag{5}
\]

It is worth noting that \( m_{p,q} \) is invariant with respect to translation, rotation, and scaling transformations, as our application domain requires.

Starting from these invariant central moments, we can describe a collection of \( n \) material points using the eccentricity concept of material points collection (6) and the axes concept of material points collection (7).

\[
eccentricity = \frac{m_{20} + m_{02} + \sqrt{(m_{20} + m_{02})^2 + 4m_{11}^2}}{m_{20} + m_{02} - \sqrt{(m_{20} + m_{02})^2 + 4m_{11}^2}} \tag{6}
\]
\[
\theta = \frac{1}{2} \tan^{-1} \left[ \frac{2m_{11}}{m_{20} - m_{02}} \right]
\] (7)

For an in-depth description of a collection of \( n \) material points, we can use the above mentioned invariant moment’s set:

- Hu moment set;
- Zernike moment set;
- Bamieh moment set;

### A.2 Hu’s moment set

In 1962 Hu formulated his theory about inertial moments [19]. He demonstrated that inertial moments described in Figure 16, and calculated using the invariant central moments shown in (5), are invariant to affine transformation.

1. \( H M_1 = m_{20} + m_{02} \)
2. \( H M_2 = (m_{20} - m_{02})^2 + 4m_{11}^2 \)
3. \( H M_3 = (m_{30} - 3m_{12})^2 + (3m_{21} - m_{03})^2 \)
4. \( H M_4 = (m_{30} + 3m_{12})^2 + (m_{21} + m_{03})^2 \)
5. \begin{align*}
HM_5 &= (m_{30} - 3m_{12})(m_{30} + m_{12}) \left[ (m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2 \right] + \\
&\quad + (3m_{21} m_{03})(m_{21} + m_{03}) \left[ 3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2 \right]
\end{align*}
6. \( HM_6 = (m_{20} - m_{02}) \left[ (m_{30} + m_{12})^2 - (m_{21} + m_{03})^2 \right] + 4m_{11}(m_{30} + m_{12})(m_{21} + m_{03}) \)
7. \begin{align*}
HM_7 &= (3m_{21} - m_{30})(m_{30} + m_{12}) \left[ (m_{30} + m_{12})^2 - 3(m_{21} + m_{03})^2 \right] + \\
&\quad + (3m_{12} m_{30})(m_{21} + m_{03}) \left[ 3(m_{30} + m_{12})^2 - (m_{21} + m_{03})^2 \right]
\end{align*}

Figure 16: Hu’s moment set.

### A.3 Zernike’s moment set

Reviewing Hu’s moment set, Zernike proposed a new version of invariant moments based on orthogonal polynomials (see, e.g., [5]). Zernicke represented the invariant moments as functions of complex polynomials described in (8) and (9), where \((n - |m|)\) is constant and \(|m| < n\).

\[
R_{mn}(\rho) = \sum_{s=0}^{(n-|m|)/2} (-1)^s \frac{(n-s)!}{s!(\frac{n+|m|}{2} - s)!}\left(\frac{n-|m|}{2} - s\right)!\] (8)
\[ V_{mn}(x, y) = V_{mn}(\rho \cos \theta, \rho \sin \theta) = R_{mn}(\rho)^{\text{im}} \theta \] (9)

A generic Zernike moment, shown in (10) is calculated developing polynomials defined in (8) and (9) on a particular circumference \( x^2 + y^2 = 1 \). The complete list of Zernike moment is shown in Figure 17

\[ A_{mn} = \frac{n + 1}{\pi} \sum_x \sum_y f(x, y) V_0(x, y) \] (10)

1. \( ZM_1 = \frac{3}{\pi}[2(m_{20} + m_{02}) - 1] \)
2. \( ZM_2 = \frac{9}{\pi}[\rho^2 - 4m_{11}^2] \)
3. \( ZM_3 = \frac{16}{\pi}[(m_{03} - 3m_{21})^2 + (m_{30} - 3m_{12})^2] \)
4. \( ZM_4 = \frac{144}{\pi}[(m_{03} + m_{21})^2 + (m_{12} + m_{30})^2] \)
5. \( ZM_5 = \frac{13824}{\pi^3} \left\{ (m_{03} - 3m_{21})(m_{03}
\quad + m_{21})^2 - 3(m_{12} + m_{30})^2 \right\} - \)
\quad + (m_{30} - 3m_{12})(m_{30} + 3m_{12}) \left\{ (m_{03} + m_{12})^2 - 3(m_{21} + m_{03})^2 \right\} \)
6. \( ZM_6 = \frac{1864}{\pi^4} (m_{02} - m_{20}) \left\{ (m_{30} + m_{12})^2 - (m_{21} + m_{03})^2 \right\} + \)
\quad + 4m_{11}(m_{03} + m_{21})(m_{30} + m_{12})

Figure 17: Zernike’s moment set.

A.4 Bamieh’s moment set

Successively, Bamieh defined his own invariant moment set in order to characterize images (see, e.g., [61]). These formulas are shown in Figure 18.

1. \( BM_1 = m_{02}m_{20} - m_{11}^2 \)
2. \( BM_2 = (m_{03}m_{30} - m_{21}m_{12})^2 - 4(m_{03}m_{12} - m_{21}^2)(m_{21}m_{30} - m_{12}^2) \)
3. \( BM_3 = m_{40}m_{04} - 4m_{31}m_{13} + 3(m_{22} \)
4. \( BM_4 = m_{41}m_{22}m_{04} - 2m_{31}m_{22}m_{13} - m_{40}m_{13}m_{04}m_{31} - m_{22}^3 \)

Figure 18: Bamieh’s moment set.
A.5 Metrics

Concerning the similarity computation, different metrics have been proposed in the literature [16]. All metrics need a vector of coefficients to adequately weigh the individual values of descriptor vectors. Our CBIR engine includes the following metrics.

- The EUclidean distance (EU) [61]. The *Euclidean distance* is the well known metrics implemented in CBIR systems. Given two vectors $F_{Pi}$ and $F_{Ii}$ representing respectively the $N$ vector image feature components and $w_i$ representing the $N$ component weigh vector, the Euclidean distance $EU$ is expressed in (11).

$$EU = \sqrt{\sum_{i=1}^{N} w_i (F_{Pi} - F_{Ii})^2}.$$  \hspace{1cm} (11)

- The Cross Correlation distance (CC) [5]. Given two vectors $F_{Pi}$ and $F_{Ii}$ representing respectively the $N$ vector image feature components and $w_i$ representing the $N$ component weigh vector, the Cross Correlation distance $CC$ is expressed in (12).

$$CC = \frac{\sum_{i=0}^{N} w_i (F_{Pi}F_{Ii})}{\sqrt{\sum_{j=1}^{N} w_j F_{Pj}^2 \sum_{k=1}^{N} w_k F_{Ik}^2}}.$$  \hspace{1cm} (12)

- Discrimination Cost distance (DC) [5]. Given two vectors $F_{Pi}$ and $F_{Ii}$ representing respectively the $N$ vector image feature components, the Discrimination Cost distance $DC$ is expressed in (13).

$$DC = \sum_{j=1}^{N} \left[ \frac{\max(F_{Pj},F_{Ij})}{\min(F_{Pj},F_{Ij})} - 1 \right]^2.$$  \hspace{1cm} (13)

- CitY Block distance (CY) [61]. Given two vectors $F_{Pi}$ and $F_{Ii}$ representing respectively the $N$ vector image feature components, the CitY Block distance $CY$ is expressed in (14).

$$CY = \sum_{i=1}^{N} |F_{Pi} - F_{Ii}|.$$  \hspace{1cm} (14)

- The CHebyschev distance (CH) [61]. Given two vectors $F_{Pi}$ and $F_{Ii}$ representing respectively the $N$ vector image feature components, the CHebyschev distance $CH$ 15

$$CH = \max_{i=1}^{N} |F_{Pi} - F_{Ii}|.$$  \hspace{1cm} (15)