INSTITUTO DE COMPUTAÇÃO
UNIVERSIDADE ESTADUAL DE CAMPINAS

A New Framework to Combine Descriptors for Content-based Image Retrieval

Ricardo da S. Torres *Alexandre X. Falcão*
Marcos André Gonçalves *Baoping Zhang*
Weiguo Fan *Edward A. Fox*

Technical Report - IC-05-021 - Relatório Técnico

September - 2005 - Setembro

A New Framework to Combine Descriptors for Content-based Image Retrieval

Ricardo da S. Torres* Alexandre X. Falcão* Marcos André Gonçalves†
Baoping Zhang ‡ Weiguo Fan‡ Edward A. Fox‡

Abstract

Methods that combine image database descriptors have strong influence on the effectiveness of content-based image retrieval (CBIR) systems. Although there are many combination functions described in the image processing literature, empirical evaluation studies have shown that those functions do not perform consistently well across different contexts (queries, image collections, users). Moreover, it is often very difficult for human beings to identify optimal combination functions for a particular application. In this paper, we propose a novel framework using *Genetic Programming* to combine image database descriptors for CBIR. Our framework is validated through several experiments involving two image databases and a specific domain, where the images are retrieved based on the shape of their objects.

1 Introduction

Advances in data storage and image acquisition technologies have enabled the creation of large image datasets. In order to deal with these data, it is necessary to develop appropriate information systems which can support different services. The focus of this paper is on content-based image retrieval (CBIR) systems [9, 19]. Basically, CBIR systems try to retrieve images similar to a user-defined specification or pattern (e.g., shape sketch, image example). Their goal is to support image retrieval based on content properties (e.g., shape, texture, and color). An image database can be indexed using different *descriptors* [24], which are characterized by: (i) a *feature extraction algorithm* that encodes image properties into a *feature vector*; and (ii) a *similarity measure* (distance function) that computes the similarity between two images as a function of the distance between their corresponding feature vectors. In the CBIR domain, a descriptor is considered more effective than another one, when it increases the number of relevant images returned, given an input query defined by a user.

*Institute of Computing, University of Campinas, Av. Albert Einstein, 1251, CEP 13084-851, Campinas, SP, Brasil.

†Department of Computer Science, Federal University of Minas Gerais, Belo Horizonte, MG, Brazil.

‡Department of Computer Science, Virginia Polytechnic Institute and State University, Blacksburg, VA, 24061-0106, USA.

Different descriptors encoding different or even the same image properties have been proposed to support image retrieval by content [24, 28]. These descriptors are commonly chosen in a domain-dependent fashion, and, generally, are combined in order to meet users' perception. For example, while one user may wish to retrieve images based on their color feature, another one may wish to retrieve images according to their texture properties. At a higher level, descriptors encoding different properties can be combined to support different perception criteria of different users. Many of these combination strategies are based on weights which assess the importance of each descriptor [16, 21].

This paper proposes a novel framework to combine image database descriptors, improving effectiveness in retrieval tasks. More specifically, we introduce a systematic and automatic discovery framework to aid the combination of descriptors. This framework is based on an artificial intelligence (AI) optimization technique, called *Genetic Programming (GP)*, which has been widely used in various design and data mining applications [5, 7, 13, 27].

Our solution relies on the creation of a *composite descriptor*, which is simply the combination of pre-defined descriptors using a GP technique. We employ GP to combine the similarity values obtained from each descriptor, creating a more effective fused similarity function. As far as we know, this approach is original and opens a new and productive field for investigation.

We validate the proposed framework in a specific domain, shape-based image retrieval, through various experiments. The approach has shown to be flexible and powerful in the search for optimal functions to combine descriptors.

The remainder of this paper is organized as follows. Section 2 describes the main AI techniques to understand the proposed framework. Section 3 presents a generic model for CBIR which includes the notion of a database descriptor and components. Section 4 presents a formal definition of the combination function discovery problem and describes our framework based on Genetic Programming. Section 5 presents several experiments, which validate our approach, while Sections 6 and 7 discuss the main achieved results and related works, respectively. In Section 8 we conclude the paper, explaining implications of this study and future research directions.

2 Background

2.1 Genetic Programming

Genetic algorithms (GAs) [11] and genetic programming (GP) [13] are a set of artificial intelligence problem-solving techniques based on the principles of biological inheritance and evolution. Each potential solution is called an individual (i.e., a chromosome) in a population. Both GA and GP work by iteratively applying genetic transformations, such as crossover and mutation, to a population of individuals to create more diverse and better performing individuals in subsequent generations. A fitness function is available to assign the fitness value for each individual.

The main difference between GA and GP relies on their internal representation - or data structure - of the individual. In general, GA applications represent each individual as a fixed-length bit string, like (1101110 . . .) or a fixed-length sequence of real numbers

(1.2, 2.4, 4, . . .). In GP, on the other hand, more complex data structures are used (e.g., trees, linked lists, or stacks [14]). Furthermore, GP data structure length is not fixed, although it may be constrained by implementation to be within a certain size limit. Because of the intrinsic parallel search mechanism and powerful global exploration capability in a high-dimensional space, both GA and GP have been used to solve a wide range of hard optimization problems that oftentimes have no known best solution.

2.2 GP Components

In order to apply GP to solve a given problem, several required key components of a GP system need to be defined. Table 1 lists these essential components along with their descriptions.

Components	Meaning
Terminals	Leaf nodes in the tree structure.
Functions	Non-leaf nodes used to combine the leaf nodes. Commonly numerical operations: +, -, *, /, log.
Fitness Function	The objective function GP aims to optimize.
Reproduction	A genetic operator that copies the individuals with the best fitness values directly into the population for the next generation without going through the crossover operation.
Crossover	A genetic operator that exchanges subtrees from two parents to form two new children. Its aim is to improve the diversity as well as the genetic fitness of the population.
Mutation	A genetic operator that replaces a selected individual's subtree, whose root is a picked mutation point, with a randomly generated subtree.

Table 1: Essential GP Components.

The entire combination discovery framework can be seen as an iterative process. Starting with a set of training images with known relevance judgments, GP first operates on a large population of random combination functions. These combination functions are then evaluated based on the relevance information from training images. If a stopping criterion is not met, it will go through the genetic transformation steps to create and evaluate the population of the next generation iteratively.

3 CBIR Model

In this section, we formalize how a CBIR system can be modeled.

Definition 1: An **image** $\hat{I}$ is a pair $(D_I, \vec{I})$, where:

- D_I is a finite set of *pixels* (points in $\mathbb{Z}^2$, that is, $D_I \subset \mathbb{Z}^2$), and

- $\vec{I} : D_I \rightarrow \mathsf{D}'$ is a function that assigns to each pixel p in D_I a vector $\vec{I}(p)$ of values in some arbitrary space D' (for example, $\mathsf{D}' = \mathbb{R}^3$ when a color in the RGB system is assigned to a pixel).

Definition 2: A **simple descriptor** (briefly, **descriptor**) D is defined as a pair (ϵ_D, δ_D), where:

- $\epsilon_D : \hat{I} \rightarrow \mathbb{R}^n$ is a function, which extracts a *feature vector* $\vec{v}_{\hat{I}}$ from an *image* $\hat{I}$.

- $\delta_D : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is a *similarity function* (e.g., based on a distance metric) that computes the similarity between two images as the inverse of the distance between their corresponding *feature vectors*.

Definition 3: A **feature vector** $\vec{v}_{\hat{I}}$ of an image $\hat{I}$ is a point in $\mathbb{R}^n$ space: $\vec{v}_{\hat{I}} = (v_1, v_2, ..., v_n)$, where n is the dimension of the vector. Examples of possible feature vectors are a color histogram [26], a multiscale fractal curve [6], a set of Fourier coefficients [20]. They essentially encode image properties, such as color, shape, and texture. Note that different types of feature vectors may require different similarity functions.

Figure 1 illustrates the use of a simple descriptor D to compute the similarity between two images $\hat{I}_A$ and $\hat{I}_B$. First, the extraction algorithm ϵ_D is used to compute the feature vectors $\vec{v}_{\hat{I}_A}$ and $\vec{v}_{\hat{I}_B}$ associated with the images. Next, the similarity function δ_D is used to determine the similarity value d between the images.

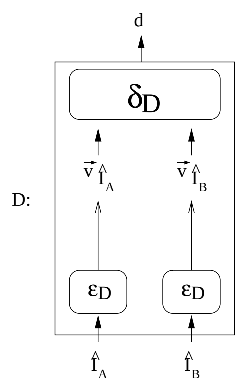


Figure 1: The use of a simple descriptor D for computing the similarity between images.

Definition 4: A **composite descriptor** $\hat{D}$ is a pair $(\mathcal{D}, \delta_{\mathcal{D}})$ (see Figure 2), where:

- $\mathcal{D} = \{D_1, D_2, \ldots, D_k\}$ is a set of k pre-defined simple descriptors.

- $\delta_{\mathcal{D}}$ is a similarity function which combines the similarity values obtained from each descriptor $D_i \in \mathcal{D}$, $i = 1, 2, \ldots, k$.

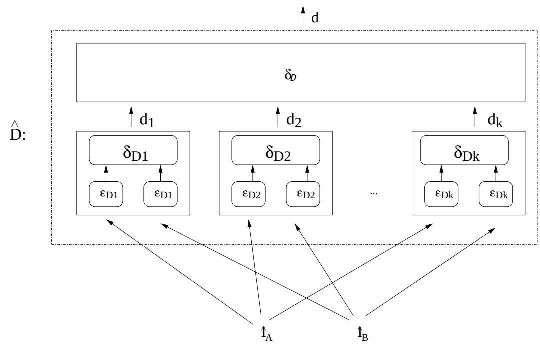


Figure 2: Composite descriptor.

4 GP Framework for CBIR

The present framework uses GP to combine simple descriptors. This decision stemmed from three reasons: (i) the large size of the search space for combination functions; (ii) previous success of using GP in information retrieval and image processing; and (iii) little prior work on applying GP to image retrieval.

The corresponding CBIR system can be characterized as follows. For a given large image database and a given user-defined query pattern (e.g., a query image), the system retrieves a list of images from the database which are most "similar" to the query pattern, according to a set of image properties. These properties may take into account shape, color, and/or texture of the image objects, and are represented by simple descriptors. These simple descriptors are combined using a composite descriptor $\mathcal{D}_{GP}$, where $\delta_{\mathcal{D}_{GP}}$ is a mathematical expression uniquely represented as an expression tree, whose non-leaf nodes are numerical operators (see Table 1) and the leaf node set is composed of the similarity values d_i, $i = 1, 2, \ldots, k$. Figure 3 shows a possible combination (obtained through the GP framework) of the similarity values d_1, d_2, and d_3 of three simple descriptors.

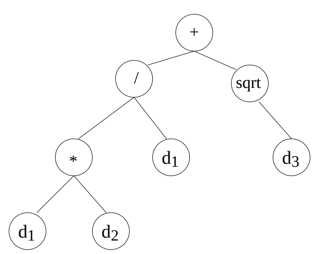


Figure 3: Example of a GP-based similarity function represented in a tree.

The overall retrieval framework is as follows:

Algorithm 1:

1. Generate an initial population of random "similarity trees".

2. Perform the following sub-steps on training images for N_{gen} generations:
   2.1. Calculate the fitness of each similarity tree.
   2.2. Record the top N_{top} similarity trees.
   2.3. Create a new population by:
      2.3.1. Reproduction
      2.3.2. Crossover
      2.3.3. Mutation
3. Apply the "best similarity tree" (i.e., the first tree of the last generation) on a set of testing (query) images.

The GP framework for the image retrieval problem is considered "global", as it tries to find the best descriptor combination (represented as just one tree), which optimizes the number of relevant images returned. "Local" strategies, which are suitable to determine the best descriptor combination for a given class, would be useful in classification problems (e.g., [27]). Ongoing work addresses this research topic.

5 Experiments

Even though the proposed framework has been specified in a generic way, allowing the combination of descriptors that encode different properties (i.e., color, texture, etc), the experiments described below were carried out for shape-based descriptors.

5.1 Shape Descriptors

This section presents a brief overview of the shape descriptors used in our experiments. This list includes widely used descriptors for comparison purposes [10] and recently proposed ones [2, 6]. Here, the GP framework is used to combine them in a suitable way, taking advantage of the fact that they encode different shape properties (frequency and spatial features, local and global information, etc.).

Many versions of these methods have been presented, but this work considers their conventional implementations.

Beam Angle Statistics (BAS): The BAS [2] is a novel shape descriptor which has been compared with several others. In [2], it is shown that BAS functions (with 40 and 60 samples) outperform all of them. The experiments of the present paper used the BAS descriptor with both 40 and 60 samples. Basically, the BAS descriptor is based on the *beams* originated from a contour pixel. A beam is defined as the set of lines connecting a contour pixel to the rest of the pixels along the contour. At each contour pixel, the angle between a pair of lines is calculated, and then the shape descriptor is defined by using the third-order statistics of all the beam angles in a set of neighborhood systems. The similarity between two BAS moment functions is measured by an optimal correspondent subsequence (OCS) algorithm as shown in [2].

Moment Invariants: For Moment Invariants, each object has been represented by a 14 dimensional feature vector, including two sets of normalized Moment Invariants [10], one from the object boundary and another from a solid silhouette. The Euclidean distance was used as similarity measure.

Fourier Descriptors: We have implemented the method described in [10, 20] to represent a shape with Fourier Descriptors applied to a contour. Each original object and its transformed versions have been represented by the most significant 126 components. Again, the Euclidean distance was used as similarity function.

Contour Multiscale Fractal Dimension or shortly, **MS Fractal Dimension**: We have implemented the method described in [6] (with degree of the multiscale fractal polynomial equal to 25 and generating a 25-bin representation) to extract multiscale fractal values for a contour. Again, the Euclidean distance has been used to measure the similarity between two multiscale fractal dimension representations.

5.2 GP System

The following is a detailed description of our implementation of the above framework.

List of terminals: As pointed out in Section 4, our terminals are composed of the similarity functions defined by each descriptor presented in Section 5.1.

Functions: The following functions were used in our implementation: $+, \times, /, sqrt$. Subtraction is not used, to avoid handling negative results. This function set is widely used in common GP experiments and is suitable to validate our ideas. We plan to use more complex functions in future experiments.

Initial Population Generation: The initial set of trees, constrained to have a maximum depth of 4 levels, were generated by the *ramped half-and-half method* [13]. This method stipulates that half of the randomly-generated trees *must* be generated by a random process which ensures all branches of the maximum initial depth. The remaining randomly generated trees require branches whose lengths do not exceed this depth. These constraints have been found to generate a good initial sample of trees [13].

Fitness Functions: The fitness function plays a very important role in guiding GA/P to obtain the best solutions within a large search space. By considering our problems, a fitness function measures how effective a combination function represented by an individual tree is for ranking images. Good fitness functions will help GA/P to explore the search space more effectively and efficiently. Bad fitness functions, on the other hand, can easily make GA/P get trapped in a local optimum solution and lose the discovery power.

The next paragraphs present a formal definition of the chosen fitness functions:

FFP1 [7]: $F_{FFP1} = \sum_{i=1}^{|D|} r(d_i) \times k_1 \times ln^{-1}(i + k_2)$, where $r(d) \in \{0, 1\}$ is the relevance score assigned to an image, it being 1 if the image is relevant and 0 otherwise. $|D|$ is the total number of retrieved images. k_1, k_2 are scaling factors. After exploratory analysis we set $k_1 = 6$ and $k_2 = 1.2$ in our experiments.

FFP2 [7]: $F_{FFP2} = \sum_{i=1}^{|D|} r(d_i) \times k_3 \times \log_{10}(1000/i)$. k_3 is a scaling factor. We set $k_3 = 2$ in our experiments.

FFP3 [7]: $F_{FFP3} = \sum_{i=1}^{|D|} r(d_i) \times k_4^{-1} \times (e^{-k_5 \times \ln(i) + k6} - k7)$. k_4, k_5, k_6, k_7 are scaling factors that are set to 3.65, 0.1, 4, and 27.32, respectively.

FFP4 [7]: $F_{FFP4} = \sum_{i=1}^{|D|} r(d_i) \times k_8 \times k_9^i$. Two scaling factors, k_8 and k_9, are set to 7 and 0.982, respectively.

PAVG@10 [3]: $F_{PAVG@10} = \frac{\sum_{i=1}^{10} \left(r(d_i) \times \left(\frac{\sum_{j=1}^{i} r(d_j)}{i} \right) \right)}{TRel}$, where $r(d) \in \{0, 1\}$ is the relevance score assigned to an image, being 1 if the image is relevant and 0 otherwise. $TRel$ is the total number of relevant images in a collection.

CHK [17]: $F_{CHK} = \frac{1}{|D|} \sum_{i=1}^{|D|} \left(r(d_i) \times \sum_{j=i}^{|D|} \frac{1}{j} \right)$

LGM [17]: $F_{LGM} = \left(\sum_{i=1}^{|D|} \left(r_B(d_i) \times \frac{1}{A} \left(\frac{A-1}{A} \right)^{i-1} \right) \right) \times \frac{\sum_{i=1}^{|D|} r(d_i)}{|D|}$, where $r_B(d) \in \{1, -1\}$ is a function returning the relevance of image d, being +1 if d is relevant, -1 otherwise. A is a user-defined parameter. We set A to 2.

The fitness functions defined above were evaluated under the GP framework. PAVG@10, or average precision after 10 images are returned, is a common measure used in information retrieval evaluations [3]. Functions FFP1, FFP2, FFP3, FFP4, CHK, and LGM were used since they follow the principles of utility theory [7, 8]. According to utility theory, there exists a utility function (a user's preference function) that assigns a utility value (the gained value from a user's perspective) for each item. These values vary from item to item. The item can be a book, a product, or an image, as in our case. In general, we assume *the utility of a relevant image decreases with its ranking order*. More formally, given a utility function $U(x)$, and two ranks x_1, x_2, with $x_1 < x_2$, according to this assumption, we expect the following condition to hold: $U(x_1) > U(x_2)$. The question is how to define the utility function. There are many possible functions that can be used to model this utility function satisfying the order-preserving condition given above. We decided to use FFP1, FFP2, FFP3, FFP4, CHK, and LGM, since most of them presented a good result in previous work on using GP for the ranking discovery problem [7].

The GP Operators:

Reproduction. Reproduction copies the top $rate\_r \times P$ trees in the current generation to the next, directly without undergoing any genetic transformation. The reproduction rate, $rate\_r$, is generally 0.1 or less, and P is the population size. In our case, $rate\_r = 0.05$.

Crossover. Crossover ensures variety by creating trees that differ from their parents. For crossover, a method called *tournament selection* [13] is used. Tournament selection works by first selecting, with replacement, k (we use 6) trees at random from the current generation. The two trees with the highest fitness are paired and exchange subtrees.

Mutation. In this case, an individual is selected, and a mutation point picked (a subtree of the individual). The subtree of the mutation point is deleted and replaced by a randomly generated subtree. Our experiments considered 0.25 as the percentage of individuals selected for mutation (the mutation rate).

Stopping Criterion: We stop the GP discovery process after 50 generations. First, the simulation is highly computationally intensive. Second, our pilot experiments with sample queries indicated that 50 generations was a sufficient period to generate high-performing trees.

5.3 Image Databases

Two different databases have been used to compare the proposed GP-based shape descriptors.

Fish Shape Database: This shape database contains one thousand images created by using one hundred fish contours chosen randomly from the data set available from [23]. Since there is no semantic definition of classes for the fish contours in this database, we defined a class as consisting of 10 different manifestations of each contour by rotation and scaling. Then, the problem consists of 100 classes with 10 shapes each. In this case, each original image is considered as query image, and its manifestations are taken as relevant images.

Experiments using this database will assess the invariance of the GP-based descriptor regarding rotation and scaling transformations.

MPEG-7 Part B: This is the main part of the Core Experiment CE-Shape-1 [15]. The total number of images in the database is 1400: 70 classes of various shapes, each class with 20 images.

We follow a two data-sets design in our experiments. We randomly split the data into training and test parts. The training set used a random 50% sample for each class. Furthemore, we considered two different samples for each data set in order to show that our approach is sample invariant.

6 Results

As mentioned earlier, the objective of an image retrieval system is to match images to a user's query and place them in descending order of their predicted relevance to the user's information requirement.

6.1 Comparison Criterion and Baselines

We used Precision after 10 images are returned as our comparison criterion.

Table 2 shows the average precision for each similarity evidence (shape descriptor). Note that the BAS60 shape descriptor presents the best result in both the MPEG-7 and Fish Shapes collections.

We also compare the effectiveness of our GP approach with a GA-based composite descriptor. The GA-based descriptor uses fixed-length sequence of real numbers (weights) to indicate the importance of each

descriptor. In this case, given a set of similarity functions δ_i of pre-defined descriptors, a GA-based similarity function is defined as $\delta_{GA}(\delta_1, \delta_2, \ldots, \delta_k) = w_1\delta_1 + w_2\delta_2 + \ldots w_k\delta_k$, where

w_i are weights defined by the GA framework. In our GA implementation, we considered a population of 100 individuals and 30 generations.

Descriptor	MPEG-7 Precision@10		Fish Shapes Precision @10	
	Sample 1	Sample 2	Sample 1	Sample 2
BAS40	65.35	64.84	83.35	81.10
BAS60	**66.27**	**65.37**	**93.25**	**92.30**
Contour Multiscale Fractal Dimension	40.71	40.05	71.35	68.85
Fourier Descriptors	20.25	20.44	24.20	23.75
Moment Invariants	34.68	35.02	63.20	61.45

Table 2: Average Precision after 10 images are returned, considering the evidences in isolation.

6.2 GP Results

Table 3 presents the average precision of the GP-based shape descriptors, using different fitness functions.

With regard to the MPEG-7 collection, GP-based descriptors outperform the BAS60 baseline. For the first sample, FFP1 was the best fitness function, while LGM was the best for the second sample. Note also that GP presents a better result when compared to the GA-based descriptor, except for the CHK fitness function when applied to sample 2.

For the Fish Shapes collection, the BAS60 shape descriptor yields a high precision value, since the relevant image set is composed of similar images obtained by affine transformations (rotation and scaling). However, despite the high effectiveness of the baseline, the results based on the GP approach are better. For this collection, the best results were obtained for the FFP2 fitness function with regard to both samples (Sample 1 and Sample 2).

Figure 4 presents the best tree obtained by the GP framework, considering the FFP2 fitness function on Sample 1 of the MPEG-7 collection. Note that the BAS60 descriptor appears in several nodes. This is an expected result since this is the best descriptor in isolation (see Table 2). Note also that this tree includes Moment Invariants and MS Fractal Dimension descriptors and does not consider the Fourier Descriptor (the worst one in isolation – see Table 2).

Figure 5 presents the precision versus recall curves of the best GP-based descriptor, the GA-based descriptor, and the best evidence by taking into account the two samples of the MPEG-7 and fish shapes collections. Note that the GP-based descriptor has the best curve in all cases, except for Sample 1 of the fish shapes data set. In this case, the GA-based descriptor outperforms the GP one for recall values lower than 0.47.

It is worth mentioning that the training step took 30 minutes, on average, for the the Fish Shapes data set (considering the two samples), running on a 3.2GHz Pentium 4 with 2G RAM. For the MPEG-7 data set, training took 40 minutes, on average.

```
(+ (+ (+ ContourMSFractal 0_0) 1)
   (+ (+ (+ ContourMSFractal 0_0)
         (+ (sqrt 0_5)
            (sqrt (* BAS60 0_5))))
      (* (* BAS60 BAS60)
         (* (* BAS60 BAS60)
            (* (* (* BAS60 BAS60)
                  (* (+ (/ BAS60
                           (/ (+ ContourMSFractal MomentInvariants)
                              (sqrt (+ 0_0 BAS60))))
                        (sqrt (+ BAS40 0_0))) BAS40))
               (+ (+ (sqrt (* (* BAS60 BAS60)
                              (* (* (* BAS60 BAS60)
                                    (+ (* BAS60 BAS60)
                                       (* 1 BAS60))) BAS40)))
                     (sqrt 0_0)) BAS60))))))
```

Figure 4: Best GP tree using the FFP2 fitness function on Sample 1 of the MPEG-7 collection.

Descriptor	MPEG-7 Precision @10		Fish Shapes Precision @10	
	Sample 1	Sample 2	Sample 1	Sample 2
BAS60	66.27	64.84	93.25	92.30
GP with PAVG@10	70.56 (6.47%)	69.21 (6.74%)	93.75 (0.54%)	92.75 (0.49%)
GP with FFP1	**70.92 (7.02%)**	69.59 (7.33%)	94.20 (1.02%)	93.30 (1.08%)
GP with FFP2	70.79 (6.82%)	69.76 (7.59%)	**94.30 (1.13%)**	**93.35 (1.14%)**
GP with FFP3	70.75 (6.76%)	69.44 (7.09%)	94.05 (0.86%)	93.30 (1.08%)
GP with FFP4	70.40 (6.23%)	68.97 (6.37%)	94.05 (0.86%)	93.30 (1.08%)
GP with CHK	70.73 (6.73%)	66.78 (2.99%)	94.20 (1.02%)	93.30 (1.08%)
GP with LGM	70.86 (6.93%)	**70.90 (9.35%)**	94.15 (0.97%)	93.20 (0.98%)
GA	69.37 (4.68%)	68.30 (5.38%)	93.40 (0.16%)	92.55 (0.27%)

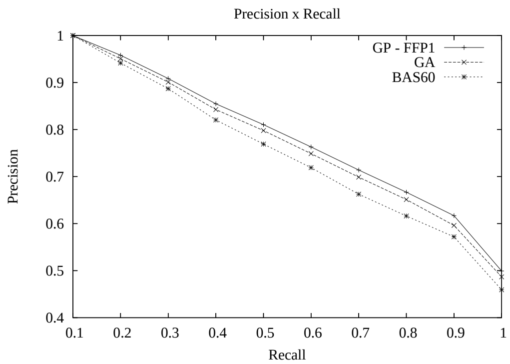
Table 3: Average Precision after 10 images are returned, considering the GP-based descriptors.

7 Related Work

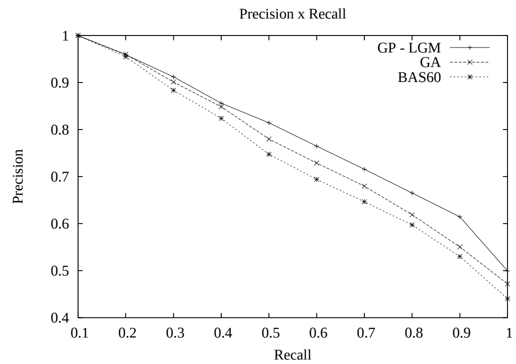
7.1 Descriptors Combination

In general, approaches for descriptors combination rely on assigning weights to indicate the importance of a descriptor [16, 21]. Basically, the higher the weight the more important a descriptor is assumed to be.

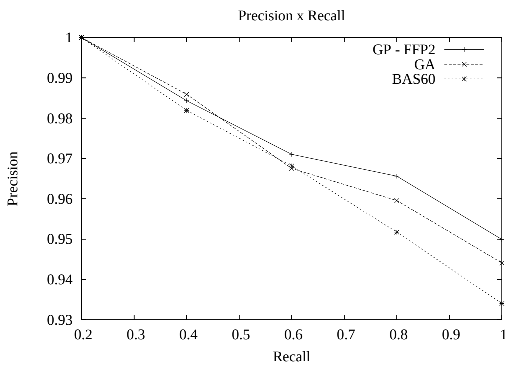
The main drawback of these approaches is the fact that it is not easy to define good weight values for a given application, or even for a given user in advance. Therefore, several techniques (such as [18] and [22]) based on user feedback have been proposed to assist weight assignment for descriptors in retrieving images by content. In general, these methods are based on user judgments with regard to the relevance of previously returned images.



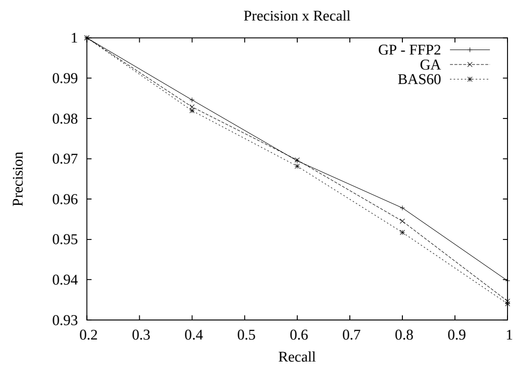
(a) MPEG-7 - Sample 1



(b) MPEG-7 - Sample 2



(c) Fish shapes - Sample 1



(d) Fish shapes - Sample 2

Figure 5: Precision versus recall curves of the best GP descriptor, GA-based descriptor, and the best evidence.

7.2 AI Techniques in Image Processing

AI techniques, such as GA and GP, have been successfully used in several image processing applications: object recognition [12, 25], object detection [4, 5], image classification [1], etc.

Howard *et al.* [12] investigated the use of GP to support automatic ship detectors in SAR (synthetic aperture radar) imagery. They use pixel statistics associated with pixel windows as terminals. Unfortunately, they do not compare their method with any other approach. Bhanu and Lin [5] applied GP to combine image processing operations for object detection. In their framework, composite operators are represented by binary trees where internal nodes represent the pre-specified primitive operators and the leaf nodes represent the original image or primitive (pre-defined) image features. They also worked on selecting appropriate features for target detection using GA [4]. A similar approach based on GA was used by Sun *et al.* [25] to select features for object detection. In image classification, Agnelli *et al.* [1] used the a GP-based framework to find out the best combination of image scalar features. They used a small database (102 images) for validation and did not compare their GP-based method with any other evolutionary approach.

8 Conclusions

We considered the problem of combining simple descriptors for content-based image retrieval. Our solution uses Genetic Programming to discover an optimal combination function. The proposed framework was validated for shape-based image retrieval, through several experiments involving two image databases, and many simple descriptors and fitness functions.

We conclude that the new framework is flexible and powerful to the design of optimal combination functions. The effectiveness results demonstrate that the GP framework can find better similarity functions than the ones obtained from the individual descriptors. Our experiments also show better results with GP than using a GA approach. In fact, even compared to outstanding baselines (BAS60 on Fish shapes data set), GP was able to find out a better result.

We also evaluated a set of fitness functions based on utility theory to find the best combination function for the image search problem. The experiments showed that several of the used fitness functions are very effective in guiding the GP search. Among the various fitness functions we tested, FFP1, FFP2, and LGM are the ones we recommend for the the image retrieval problem.

Future work will focus on evaluating the use of *validation sets* to select combination functions that generalize well for unseen images, and thus avoiding the effect of over-training [27]. We also plan to devise an automatic mechanism to incorporate the GP-based descriptors in searching engines.

Acknowledgements

The authors thank the Microsoft Escience Project, FAPESP (Proc. 03/14096-8), and CNPq (Proc.302427/04-0).

This work was also supported by AOL (through a fellowship to Marcos Andre Goncalves) and by NSF grants DUE-0121679, DUE-0333531, IIS-0307867, and IIS-0325579. Marcos Andre Goncalves is currently supported by CNPq.

References

[1] D. Agnelli, A. Bollini, and L. Lombardi. Image Classification: an Evolutionary Approach. *Pattern Recognition Letters*, 26(1-3):303–309, January 2002.

[2] N. Arica and F. T. Y. Vural. BAS: A Perceptual Shape Descriptor Based on the Beam Angle Statistics. *Pattern Recognition Letters*, 24(9-10):1627–1639, June 2003.

[3] R. A. Baeza-Yates, R. Baeza-Yates, and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc, Boston, MA, USA, 1999.

[4] B. Bhanu and T. Lin. Genetic Algorithm based Feature Selection in SAR Images. *Image and Vision Computing*, 21(7):591–608, 2003.

[5] B. Bhanu and Y. Lin. Object Detection in Multi-Modal Images Using Genetic Programming. *Applied Soft Computing*, 4(2):175–201, May 2004.

[6] R. da S. Torres, A. X. Falcão, and L. da F. Costa. A Graph-based Approach for Multiscale Shape Analysis. *Pattern Recognition*, 37(6):1163–1174, June 2004.

[7] W. Fan, E. A. Fox, P. Pathak, and H. Wu. The Effects of Fitness Functions on Genetic Programming-Based Ranking Discovery for Web Search. *Journal of the American Society for Information Science and Technology*, 55(7):628–636, 2004.

[8] P. C. Fishburn. *Non-Linear Preference and Utility Theory*. Johns Hopkins University Press, Baltimore, 1988.

[9] M. Flickner, H. Sawhney, W. Niblack, Q. H. J. Ashley, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by Image and Video Content: the QBIC System. *IEEE Computer*, 28(9):23–32, Sep 1995.

[10] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley, Reading, MA, USA, 1992.

[11] J. H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, 1992.

[12] D. Howard, S. C. Roberts, and R. Brankin. Target Detection in SAR Imagery by Genetic Programming. *Advances in Engineering Software*, 30(5):303–311, May 1999.

[13] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[14] W. B. Langdon. *Data Structures and Genetic Programming: Genetic Programming + Data Structures = Automatic Programming!* Kluwer, 1998.

[15] L. J. Latecki and R. Lakamper. Shape Similarity Measure Based on Correspondence of Visual Parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1185–1190, 2000.

[16] M. S. Lew, editor. *Principles of Visual Information Retrieval – Advances in Pattern Recognition*. Springer-Verlag, London Berlin Heidelberg, 2001.

[17] C. Lopez-Pujalte, V. P. G. Bote, and F. de Moya Anegon. Order-based fitness functions for genetic algorithms applied to relevance feedback. *Journal of the American Society for Information Science and Technology*, 54(2):152–160, 2003.

[18] S. D. MacArthur, C. E. Brodley, A. C. Kak, and L. S. Broderick. Interactive Content-Based Image Retrieval Using Relevance Feedback. *Computer Vision and Image Understanding*, 88(2):55–75, 2002.

[19] V. E. Ogle and M. Stonebraker. Chabot: Retrieval from Relational Database of Images. IEEE *Computer*, 28(9):40–48, Sep 1995.

[20] E. Persoon and K. Fu. Shape Discrimination Using Fourier Descriptors. *IEEE Transanctions on Systems, Man, and Cybernetics*, 7(3):170– 178, 1977.

[21] K. Porkaew, S. Mehrotra, M. Ortega, and K. Chakrabarti. Similarity Search Using Multiple Examples in MARS. In *Visual Information and Information Systems, LNCS*, volume 1614, pages 68–75, Amsterdam, 1999.

[22] Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. A Power Tool in Interactive Content-Based Image Retrieval. *IEEE Tran on Circuits and Systems for Video Technology*, 8(5):644–655, 1998.

[23] ShapeDB. www.ee.surrey.ac.uk/research/vssp/imagedb/ demo.html, May 2005.

[24] A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-Based Image Retrieval at the End of the Years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, December 2000.

[25] Z. Sun, G. Bebis, and R. Miller. Object detection using feature subset selection. *Pattern Recognition*, 37(11):2165–2176, 2004.

[26] M. Swain and D. Ballard. Color Indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[27] B. Zhang, M. A. Gonçalves, W. Fan, Y. Chen, E. A. Fox, P. Calado, and M. Cristo. Combining structural and citation-based evidence for text classification. In *Proceedings of the 13th ACM Conference on Information and Knowledge Management*, pages 162–163, 2004.

[28] D. Zhang and G. Lu. Review of Shape Representation and Description. *Pattern Recognition*, 37(1):1–19, Jan 2004.