Emergent Semantics Through Interaction in Image Databases

Simone Santini, Amarnath Gupta and Ramesh Jain*

To appear on *IEEE Transaction of Knowledge and Data Engineering*, circa summer 2001.

Abstract

In this paper we discuss briefly some aspects of image semantics and the role that it plays for the design of Image Databases. We argue that images don't have an intrinsic meaning, but that they are endowed with a meaning by placing them in the context of other images and by the user interaction. From this observation, we conclude that in an image database users should be allowed to manipulate not only the individual images, but also the relation between them. We present an interface model based on the manipulation of *configurations* of images.

1 Introduction

In this paper we propose some new ideas on image semantics, and study some of their consequences on the interaction with—and the organization of—image databases. Many current Content Based Image Retrieval (CBIR) systems follow a semantic model derived from traditional databases according to which the meaning of a record is a compositional function of its syntactic structure and of the meaning of its elementary constituents. We show that this definition and its obvious extensions are inadequate to capture the real meaning of an image. Even if we could do perfect object recognition (which we can't), this would still not be enough to define an image semantcs that can satisfy the user of a database. Images are designed to convey a certain message, but this message is concealed in the whole organization of the image, and is not always possible to divide it syntactically into smaller parts.

We propose that the meaning of an image is characterized by the following properties:

- It is *contextual*. The meaning of an image depends on the particular conditions under which a query is made, and the particular user that is querying the database.
- It is *differential*. The meaning of an image is made manifest by the differentiation between an image which possesses that meaning and images which don't, and by association among different images that share that meaning.

*Simone Santini and Ramesh Jain are with the Visual Computing Laboratory, University of California, San Diego, {ssantini,jain}@ece.ucsd.edu. Amarnath Gupta is with the San Diego Supercomputer Center, gupta@sdsc.edu

- It is *grounded in action*. The database can estabilish the meaning of an image based on the actions of the user when the image is presented. In a database situation, the only action allowed to the user is asking a query. Therefore, the meaning of an image will be revealed to the database by interpreting the sequence of queries posed by the user[1].

These ideas led us to the design of a differnt type of image database. In our system meaning is not an intrinsic property of the images that the database *filters* during the query, but an *emergent* property of the interaction between the user and the database. The interface between man and machine assumes a preponderant role in this new organization, and dictates new requirements for the design of the similarity engine. We will analyze these requirements and delineate the design solutions that they generated.

This paper is organized as follows. Section 2 discusses the inadequacy of traditional database semantics for the description of image meaning, and proposes an alternative definition in which semantics is an *emergent* property of the interaction between the user and the database. Section 3 is a high level functional description of an interface that support emergent semantics through exploration and reorganization of the image space. Section 4 is a more detailed description of the same interface as a set of operators that are defined between three spaces: the *feature space*, the *query space*, and the *display space*. This interface imposes certain generality requirements on the image representation and the similarity measure that go beyond the capabilities of many current similarity engines. Section 5 presents an image representation and a family of distance measures with the necessary representation power. Section 6 studies in detail the two most characteristic operators of the new interface model: *projection* and *query creation*. Section 7 presents examples of interaction between a user and a database using our prototype "El Niño." Conclusions and future directions of research are presented in Section 8.

2 Meaning

In most databases, the meaning[2] of a record is a simple function of the syntactic structure of the record and of the meanings of its components. In other words, the meaning of a record is compositional. The meaning of a stimulus in a given situation is related to the set of possible actions of an actor in that situation [10]. In a database situation, the only possible actions are asking queries and answering them. Then, if $\mathcal{Q}$ is the set of all possible queries, the meaning of a record, or a fragment of a record R, can be defined as a function

$$[R] : \mathcal{Q} \rightarrow \{\text{yes, no}\} \tag{1}$$

such that $[R](q) = \text{yes}$ if the record r satisfies the query q. Compositionality implies that, if a record is produced by a rule like

$$j : R \rightarrow \alpha_1 R_1 \alpha_2 \cdots \alpha_n R_n \alpha_{n+1} \tag{2}$$

[1] It has been recognized that this behavioristic interpretation impoverishes somehow the semiotic space of the image-sign [2], but even this impoverished semiotic space is sufficient for our purposes.

[2] In this paper, we will commit the slight imprecision of using the terms "meaning" and "semantics" interchangeably.

where α_i are terminal symbols of the record definition language, R_i are non terminal symbols, and j is the label of the production rule, then the meaning of R is:

$$[R] = f_j([R_1], [R_2], \ldots, [R_n]). \tag{3}$$

The meaning of the whole record depends on the production rule and on the meaning of the non terminals on the right side of the production, but not on the syntactic structure of the non terminals $R_1, \ldots, R_n$.

This property makes the analysis of the meaning of records in traditional databases conceptually easy but, unfortunately, it does not hold for images. As Umberto Eco puts it:

> The most naïve way of formulating the problem is: are there iconic sentences and phonemes? Such a formulation undoubtedly stems from a sort of verbocentric dogmatism, but in it ingenousness it conceals a serious problem.[2]

The problem is indeed important and the answer to the question, as Eco points out, is "no." Although some images contain syntactical units in the form of objects, underneath this level it is no longer possible to continue the subdivision, and the complete semantic characterization of images goes beyond the simple enumeration of the objects in it and their relations (see [20] for examples).

Semantic level beyond the objects are used very often in evocative scenarios, like art and advertising [1]. There is, for instance, a fairly complex theory of the semantics associated with color [5], and with artistic representational conventions [3].

The full meaning of an image depends not only on the image data, but on a complex of cultural and social conventions in use at the time and location of the query, as well as on other contigiencies of the context in which the user interacts with the database. This leads us to reject the somewhat Aristotelean view that the meaning of an image is an immanent property of the image data. Rather, the meaning arises from a process of interpretation and is the result of the interaction between the image data and the interpreter. The process of querying the database should not be seen as an operation during which images are filtered based on the illusory pre-existing meaning but, rather, as a process in which meaning is *created* through the interaction of the user and the images.

The cultural nature of the meaning and its dependence on the interpretation process show that meaning is not a simple function of the image data alone. The Saussurean relation between the *signifiant* and the *signifié* is always mediated. This relation, much like in linguistic signs, does not stand alone, but is only identifiable as part of a system of oppositions and differences. In other words, the referential relation between a sign (be it an icon or a symbol) can only stand when the sign is made part of a system. Consider the images of Fig. 1. The image at the center is a Modigliani portrait and, placed in the context of other 20th century paintings (some of which are portraits and some of not), suggests the notion of "painting." If we take the same image and place it in the context of Fig. 2, the context suggests the meaning "Face."

These observations rule out the possibility of extracting some intrinsically determined meaning from the images, storing it in a database, and using it for indexing. The meaning of the

Figure 1: A Modigliani portait placed in a context that suggests “Painting.”

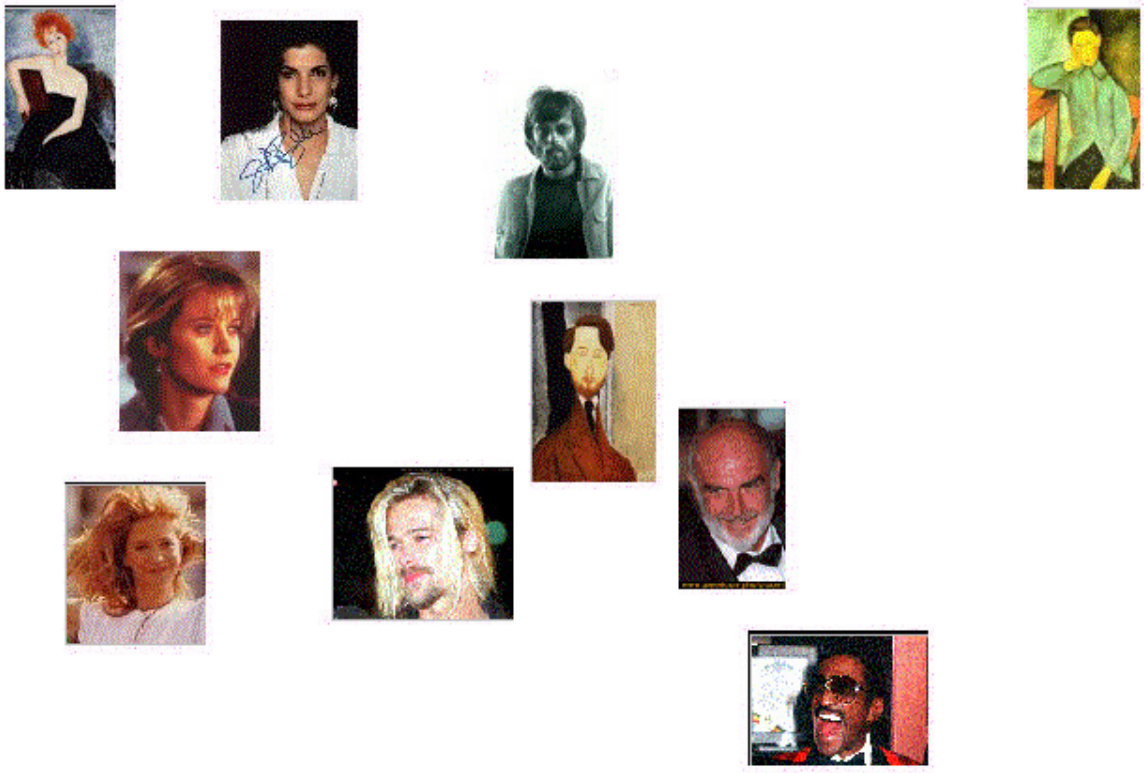

Figure 2: A Modigliani portrait placed in a context that suggests "Face."

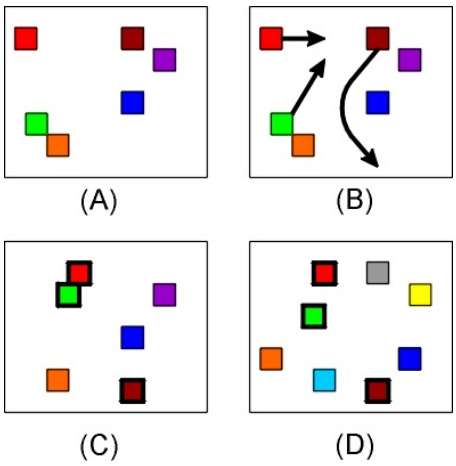

Figure 3: Schematic description of an interaction using a direct manipulation interface.

image data can only *emerge* from the interaction with the user. The user will provide the cultural background in which meaning can be grounded. We call this concept *emergent semantics.* This concept has important consequences for our query model and, ultimately, for the archtecure of image databases: the *filtering* approach, typical of symbolic databases, is no longer viable. Rather, interfaces should support a process of *guided exploration* of the database. The interface is no longer the place where questions are asked and answers are obtained, but it is the tool for active manipulation of the database as a whole.

3 Interfaces for Emergent Semantics

Based on the ideas in the previous sections, we replaced the query-answer model of interaction with a guided *exploration* metaphor. In our model, the database gives information about the status of the whole database, rather than just about a few images that satisfy the query. The user manipulates the image space directly by moving images around, rather than manipulating weights or some other quantity related to the current similarity measure[3] The manipulation of images in the display causes the creation of a similarity measure that satisfies the relations imposed by the user.

An user interaction using an exploratory interface is shown schematically in Fig. 3. In Fig. 3.A the database proposes a certain distribution of images (represented schematically as simple shapes) to the user. The distribution of the images reflects the current similarity interpretation of the database. For instance, the triangular star is considered very similar to the octagonal star, and the circle is considered similar to the hexagon. In Fig. 3.B the user moves some images around to reflect his own interpretation of the relevant similarities. The result is shown in Fig. 3.C. According to the user, the pentagonal and the triangular stars are

[3]The manipulation of the underlying similarity measure by explicitly setting a number of weights that characterize it is very common in current image databases but very counterintuitive.

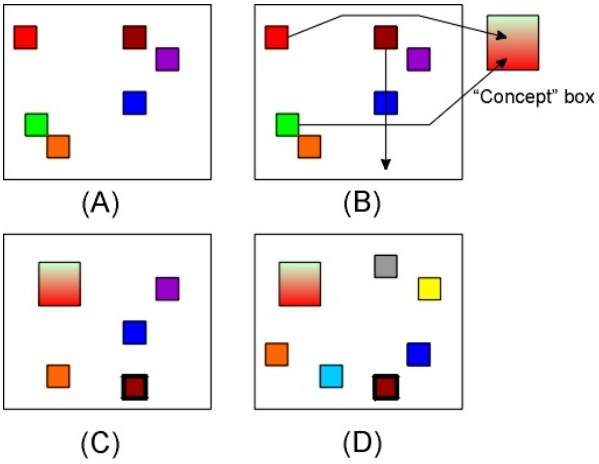

Figure 4: Interaction involving the creation of concepts.

are quite similar to each other, and the circle is quite different from both of them.

As a result of the user assessment, the database will create a new similarity measure, and re-order the images, yielding the configuration of Fig. 3.D. The pentagonal and the triangular stars are in this case considered quite similar (although they were moved from their intended position), and the circle quite different. Note that the result is not a simple rearrangement of the images in the interface. For practical reasons, an interface can't present more than a small fraction of the images in the database. Typically, we display the 100-300 images most relevant to the query. The reorganization consequent the user interaction involves the whole database. Some images will disappear from the display (the hexagon in Fig. 3.A), and some will appear (e.g. the black square in Fig. 3.D).

A slightly different operation on the same interface is the definition of *visual concepts*. In the context of our database, the term concept has a more restricted scope than in the common usage. A visual concept is simply a set of images that, for the purpose of a certain query, can be considered as equivalent or almost equivalent. Images forming a visual concept can be dragged into a "concept box" and, if necessary, associated with some text (the text can be used to retrieve the concept and the images similar to it). The visual concept can be then transformed into an icon and placed on the screen like every other image.

Fig. 4 is an example of interaction involving the creation of visual concepts. Fig. 4.A contains the answer of the database to a user query. The user considers the red and green images as two instances of a well defined linguistic concept. The user opens a *concept box* and drags the images inside the box. The box is then used as an icon to replace the images in the display space.

From the point of view of the interface, a concept is a group of images that occupy the same position in the display space. In addition, it is possible to attach metadata information to a concept. For instance, if a user is exploring an art database, he can create a concept called "medieval crucifixion." The words "medieval" and "crucifixion" can be used to replace

the actual images in a query. This mechanism gives a way of integrating visual and non visual queries. If an user looks for medieval paintings representing the crucifixion, she can simply type in the words. The corresponding visual concept will be retrieved from memory, placed in the visual display, and the images contained in the concept will be used as examples to start a visual query.

The interface, as it is presented here, is but a skeleton for interaction that can be extended along many directions. Its main limitation (and the relative opportunities for improvement) are the following:

- Visual concepts can't be nested: a visual concept can't be used as part of the definition of another visual concept (although the images that are part of it can).
- With the exception of visual concept, which can be stored, the system has no long term memory, that is, no context information can be transported from one user session to another.
- Contexts (distributions of images) can't be saved and restored, not even within a session.
- The system does not allow cooperation between users. There is no possibility of defining a "user profile" that can be used to retrieve contexts from other users and use them for some form of social filtering [].

All these limitations can be overcome without changing tha nature or the basic design of the interface, by adding appropriate funtions to the system. In this paper we will not consider them, and we will limit the discussion to the basic interface. Further study is necessary to determine which solutions are useful, useless, or (possibly) detrimental.

An exploration interface requires a different and more sophisticated organization of the database. In particular, the database must accommodate arbitrary (or almost arbitrary) similarity measures, and must automatically determine the similarity measure based on the user interface. In the following section we describe in greater details the principles behind the design of exploratory interfaces.

4 Exploration Operators

The exploration interface is composed of three spaces and a number of operators [4]. The operators can be transformations of a space onto itself or from one space to another. The three space on which the interface is based are:

- The *Feature space* $\mathcal{F}$. This is the space of the coefficients of a suitable representation of the image. The feature space topological, but not metric. There is in general no way to assign a "distance" to a pair of feature vectors.

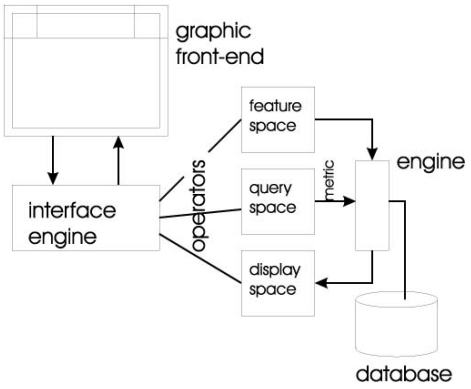

Figure 5: Operator algebra as a mediation between the graphical front-end and the database.

- The *Query space* $\mathcal{Q}$. When the feature space is endowed with a metric, the result is the query space. The metric of the query space is derived from the user query, so that the distance from the origin of the space to any image defines the "dissimilarity" of that image from the current query.

- The *Display space* $\mathcal{D}$ is a low dimensional space (0 to 3 dimensions) which is displayed to the user and with which the user interacts. The distribution of images in the display space is derived from that of the query space. We will mainly deal with two-dimensional display spaces (as implemented in a window on a computer screen.) For the sake of convenience, we also assume that every image in the visualization space has attached a number of *labels* λ_i drawn from a finite set. Examples of labels are the visual concepts to which an image belongs. The conventional label α is assigned to those images that have been selected and placed in their position (*anchored*)by the user.

The feature space is a relatively fixed entities, and is a property of the database. The query space, on the other hand, is created anew with a different metric after every interaction with the user.

The interaction is defined by an algebra of operators between these three spaces. The operators play more or less the same role that the query algebra plays in traditional databases (although, as we havementioned in the previous section, the term "query" is in this case inappropriate). In all practical instances of the system, the user does not deal with these operators directly, but through a suitable graphic front-end (see Fig. 5), whose characteristics vary depending on the particular application. Later in this paper we will consider the graphical front-end of our database system El Niño ([18]). In this section we will be concerned with the operators that mediate the exploration of the database.

4.1 Operators in the Feature Space

A feature is an attribute obtained applying some image analysis algorithms to the image data. Features are often numeric, collected in a feature vector, and immersed in a suitable vector

space, although this is not always the case.

We make a distinction between the raw, unprocessed vector space and spaces that are adapted from it for reasons of convenience. This distinction is not fundamental (all feature spaces are the result of some processing) but it will be useful to describe the operators. The *raw feature space* is the set of complete feature vectors, as they come out of the image analysis algorithms. In many cases, we need to adjust these vectors for the convenience of the database operations. A very common example is dimensionality reduction [12]. In this case, we will say that we obtain a (reduced-dimensional) *stored view* of the feature space. The operators defined on the feature space are used for this purpose. The most common are:

Projection. The feature vector is projected on a low dimensional subspace of the raw feature space, obtaining a low dimensional view. Operators like Singular Value Decomposition, projection of Zernike and statistical moments belong to this class.

Quantization. These operators are used in non-vector feature spaces like the set of coefficients used in [18]. In this case, we reduce the dimensionality of the feature space by representing an image with a limited number of coefficients (e.g. 50 or 100). This is done by vector quantization of the image coefficients (this kind of operation will be considered more in detail in section 5).

Apply Function. Applies the function F to all the elements of a set of numbers to create another set of numbers of the same dimension. Filtering operations applied to color histograms belong to this class.

These operators "prepare" the feature space for the database operations. They are not properly part of the interaction that goes on in the interface, since they are applied off-line before the user starts interacting. We have mentioned them for the sake of completeness.

4.2 The Query Space

The feature space, endowed with a similarity measure derived from a query, becomes the query space. The "score" of an image is determined by its distance from the origin of this space according to a metric dependent on the query. We assume that every image is represented as a set of n number (which may or may not identify a vector in an n-dimensional vector space) and that the query space is endowed with a distance function that depends on m parameters.

The feature sets corresponding to images x and y are represented by x^i and y^i, $i = 1, \ldots n$, and the parameters by ξ^μ, $\mu = 1, \ldots, m$. Also, to indicate a particular image in the database we will use either different Latin letters, as in x^i, y^i or an uppercase Latin index. So, x_I is the I-th image in the database ($1 \leq I \leq N$), and $x_I^j, j = 1, \ldots, n$ is the corresponding feature vector.

The parameters ξ^μ are a representation of the current query, and are the values that determine the distance function. They can also be seen as encoding the current database approximation to the user's semantic space.

Given the parameters ξ^μ, the distance function in the query space can be written as

$$f : \mathbf{R}^n \times \mathbf{R}^n \times \mathbf{R}^m \to \mathbf{R}^+ : (x^i, y^i, \xi^\mu) \mapsto f(x^i, y^i; \xi^\mu) \tag{4}$$

with $f \in L^2(\mathbf{R}^n \times \mathbf{R}^n \times \mathbf{R}^m, \mathbf{R}^+)$. Depending on the situation, we will write $f_\xi(x^i, y^i)$ in lieu of $f(x^i, y^i; \xi^\mu)$.

As stated in the previous section, the feature space per se is topological but not metric. Rather, its intrinsic properties are characterized by the functional

$$L : \mathbf{R}^m \to L^2(\mathbf{R}^n \times \mathbf{R}^n, \mathbf{R}^+) \tag{5}$$

which associates to each query ξ^μ a distance function:

$$L(\xi^\mu) = f(\cdot, \cdot; \xi^\mu). \tag{6}$$

A query, characterized by a vector of parameters ξ^μ, can also be seen as an operator q which transforms the feature space into the query space. If L is the characteristic functional of the feature space, then $qL = L(\xi)$ is the metric of the query space. This is a very important operator for our database model and will be discussed in section 6.

Once the feature space $\mathcal{F}$ has been transformed into the metric query space Q, other operations are possible [4, 20], like:

Distance from Origin Given a feature set x^i, return its distance from the query:

$$D_0(x^i) = f(0, x^i; \xi^\mu) \tag{7}$$

Distance Given a two feature sets x^i, y^i, return the distance between the two

$$D(x^i, y^i) = f(0, x^i; \xi^\mu) \tag{8}$$

Select by Distance. Return all feature sets that are closer to the query than a given distance:

$$S(d) = \left\{ x^i : D(x^i) \leq d \right\} \tag{9}$$

k-Nearest Neighbors. Return the k images closest to the query

$$N(k) = \left\{ x^i : \left| \left\{ y^i : D(y^i) < D(x^i) \right\} \right| < k \right\} \tag{10}$$

It is necessary to stress again that these operations are not defined in the feature space $\mathcal{F}$ since that space is not endowed with a metric. Only when a query is defined does a metric exist.

4.3 The Display Space

The display operator ϕ projects image x^i on the screen position X^Ψ, $\Psi = 1, l$[4] in such a way that

$$d(X^\Psi, Y^\Psi) \approx f(x^i, y^i; \xi^\mu) \tag{11}$$

We use a simple elastic model to determine the position of images in the display space, as discussed in section 6. The result is an operator that we write:

$$\phi(x_I^k; f_\xi) = (X_I^\Psi, \emptyset). \tag{12}$$

The parameter f_ξ reminds us that the projection that we see on the screen depends on the distribution of images in the query space which, in turn, depends on the query parameters ξ^μ. The notation $(X_I^\Psi, \emptyset)$ means that the image x_I is placed at the coordinates x_I^Ψ in the display space, and that there are no labels attached to it (viz. the image is not anchored at any particular location of the screen, and does not belong to any particular visual concept).

A configuration of the display space is obtained by applying the display operator to the entire query space:

$$\phi(\mathcal{Q}) = \phi(\mathcal{F}; f_\xi) = \left\{(X_I^\Psi, \lambda_I)\right\} \tag{13}$$

where λ_I is the set of labels associated to image I. As we said before, it is impractical to display the whole database. More often, we display only a limited number P of image. Formally, this can be done by applying the P-nearest neighbors operator to the space $\mathcal{Q}$:

$$\phi(N(P)(\mathcal{Q})) = \phi(N(P)(\mathcal{F}; f_\xi)) = \left\{(X_I^\Psi, \lambda_I), i = 1, \ldots, P\right\} \tag{14}$$

The display space $\mathcal{D}$ is the space of the resulting configurations. With these definitions, we can describe the operators that manipulate the display space.

The Place Operator The place operator moves an image from one position of the display space to another, and attaches a label α to the images to anchor it to its new position. The operator that places the I-th image in the display is $\zeta_I : \mathcal{Q} \to \mathcal{Q}$ with:

$$\zeta_I \left\{(X_J^\Psi, \lambda_J)\right\} = \left(\left\{(X_J^\Psi, \lambda_J)\right\} - \left\{(X_I^\Psi, \lambda_I)\right\}\right) \cup \left\{(\tilde{X}_I^\Psi, \lambda_I \cap \alpha)\right\} \tag{15}$$

where $\tilde{X}$ is the position given to the image by the user.

Visual Concept Creation A visual concept is is a set of images that, conceptually, occupy the same position in the display space and are characterized by a set of labels. Formally, we will include in the set of labels the keywords associated to the concept as well as the identifiers

[4]Capital Greek indices will span $1, l$. The most frequent case, in our experience, is the simple two-dimensional display, that is, $l = 2$. The arguments presented here, however, hold for any l-dimensional display. Conventionally, $l = 0$ refers to a browser in wihch only ordinal properties are displayed.

of the images that are included in the concept. So, if the concept contains images $I_1, \ldots, I_k$, the set of labels is

$$\chi = (W, \{I_1, \ldots, I_k\}) \tag{16}$$

where W is the set of keywords. We call X the set of concept, and we will use the letter χ to represent a concept.

The creation of a concept is an operator $\kappa : \mathcal{D} \to X$ defined as:

$$\kappa\left\{(X_J^\Psi, \lambda_J)\right\} = \left(\cdot, \bigcup_J \lambda_J \cup W \cup \{I_1, \ldots, I_k\}\right\} = \{\cdot, \chi \bigcup_J \lambda_J\} \tag{17}$$

This concept takes a set of images, in positions X_J^Ψ and attached labels λ_J, and transforms them in an entity that formally is an image with unspecified position, and a set of labels composed of the union of all the labels of the images ($\bigcup_J \lambda_J$), the set of keywords associated to the concept, and the list of the images that belong to the concept.

Visual Concept placement The insertion of a concept in a position Z^Ψ of the display space is defined as the action of the operator $\eta : X \times \mathbf{R}^2 \times \mathcal{D} \to \mathcal{D}$ defined as:

$$\eta\left(\chi, Z^\Psi, \left\{(X_J^\Psi, \lambda_J)\right\}\right) = \left\{(X_J^\Psi, \lambda_J)\right\} \cup \left\{Z^\Psi, \alpha \cup \chi\right\} \tag{18}$$

Metadata Queries Visual concepts can be used to create visual queries based on semantic categories. Suppose a user enters a set of words A. It is possible to define the distance from the keyword set A to the textual portion of a visual concept label χ using normal information retrieval techniques [13]. Let $d(A, \chi)$ be such a distance. Similarly, it is possible to determine the distance between two concepts $d(\chi_1, \chi_2)$. Then the textual query A can be transformed in a configuration of the display space

$$\left\{X_I^\Psi, \alpha \cup \chi_I\right\} \tag{19}$$

where

$$\left[\sum_\Psi \left(X_I^\Psi\right)^2\right]^{\frac{1}{2}} \approx d(A, \chi_I) \tag{20}$$

and

$$\left[\sum_\Psi \left(X_I^\Psi - X_J^\Psi\right)^2\right]^{\frac{1}{2}} \approx d(\chi_I, \chi_J) \tag{21}$$

The resulting metadata query operator μ takes a set of keywords A and returns a configuration of images in the display space that satisfies the relation (21):

$$\mu(A) = \{(x_I^j, \alpha \cup \chi_I)\} \tag{22}$$

In other words, we can use the distance between the concepts and the query, as well as the distances between pairs of concepts, to place the corresponding images in the display space, thus transforming the textual query in a visual query.

The description of the interaction as the action of a set of operators provides a useful algebraic language for the description of the query process. Operators provide the necessary grounding for the study of problems of query organization and optimization in the exploratory framework. Operators, however, still constitute a functional description of the exploration components. They don't (nor do they purport to) offer us an *algorithmic* description of the database. The implementation of the operators requires considering a suitable image representation in a space with the required metric characteristics. We will study this problem in the next section.

5 Image Representation

A good image representation for image database applications should permit a flexible definition of similarity measures in the resulting feature space. There are two ways in which this requirement can be interpreted. First, can require that some predefined distance (usually the Euclidean distance or some other Minkowski distance) give us sound results when applied to the feature space within a certain (usually quite narrow) range of semantic significance of these features. Extracting features like color histograms or Gabor descriptors for textures [9] may satisfy this requirement for certain classes of images.

A second sense in which the requirement can be understood is that of *coverage.* In this case we require that the feature space support many different similarity criteria, and that it be possible to switch from one criterion to another simply by changing the definition of the distance in the feature space.

This possibility is a property both of the distance and of the feature space. On one hand, we must define a class of distances rich enough to represent most of the similarity criteria that we are likely to encounter in practice. On the other hand, this effort would be fruitless if the feature space (i.e. the image representation) were not "rich in information," that is, if it didn't represent enough aspects of the images to support the definition of these similarity criteria.

To make a simple example, consider a feature space consisting of a simple histogram of the image. No matter what similarity measure we define, in this space it is impossible to formulate a query based on structure or spatial organization of the image.

In this section we introduce a general and efficient image representation that constitutes a feature space with wide coverage for image database applications.

A color image is a function $f : S \subset R^2 \to \mathcal{C} \subset R^3$. $\mathcal{C}$ is the color space, which we assume endowed with a metric, whose exact characteristics depend on the specific color system adopted. In order to measure distances, we need to endow the space $\mathcal{F}$ of image functions with a suitable metric structure.

It is a well known fact that groups of transformations can be used to generate continuous wavelet transforms of images [7]. If the transformation group from which we start is endowed with a metric, this will naturally induce a metric in the space of transformed images.

Consider the space $L^2(X, m)$, where m is a measure on X, and a Lie group G which acts freely and homogeneously on X[14]. We will assume that G, as a manifold, is endowed with a

Riemann metric. If $g \in G$, then a representation of G in $L^2(X)$ is a homomorphism

$$\pi : G \to L^2(X) \tag{23}$$

The irreducible unitary representation of G on $L^2(X, Y)$ (where Y is a given smooth manifold) is defined as

$$\pi(g)(f) : x \mapsto \sqrt{\frac{dm(g^{-1}x)}{dm(x)}} f\left(g^{-1}x\right) \tag{24}$$

where m is a measure on X and f is a function $f : X \to Y$. This representation can be used to generate a wavelet transform. Starting from a *mother wavelet* $\psi \in L^2(X)$, we define $\psi_g = \pi(g)(\psi)$. If, for the moment, we assume $Y = \mathbb{R}$ (as would be the case, for instance, for a grayscale image), the transform of f is:

$$T_f(g) = \langle f, \psi_g \rangle \tag{25}$$

Note that the wavelet transform of f is defined on G. In the case of images, we have $X = \mathbb{R}^2$. In case of color images, the assumption $Y = \mathbb{R}$ does not hold, and it not clear how to define the inner product $\langle f, \psi_g \rangle$. A possibility, often used in literature, is to treat the color image as three separate grayscale images, and to process them independently [6]. Another possibility, introduced in [16], is to define an algebra of colors. In this case the values of $T_f(g)$ in (30) are also colors.

It is possible to extend the same definition to discrete groups [15] by consider a discrete subgroup $G_D \subseteq G$. If G is a Lie group of dimension n, consider a set of indices $I \subset Z^n$. We consider the set

$$G_D = \{g_j \in G : j \in I\}, \tag{26}$$

under the usual closeness conditions that characterize subgroups.

Given a representation of G, $\pi : G \to L^2(X)$, the restriction of π to G_D is obviously a representation of G_D. In particular, the canonical unitary representation of G_D is

$$\left[\pi_D(g_j)f\right](x) = \sqrt{\frac{dm(g_j^{-1} \cdot x)}{dm(x)}} f\left(g_j^{-1} \cdot x\right) \tag{27}$$

and the G_D-frame transform of a function f with mother wavelet ψ is:

$$T_f : G_D \to \mathcal{C} : f \mapsto \langle f, \pi_D(g_j)\psi \rangle. \tag{28}$$

The system of indices j plays an important role in implementation: the discrete transform is usually stored in a multidimensional array and accessed through the indices of the array.

We can endow the set of indices I with a group structure such that there is a homomorphism η of I into G_D. This homomorphism induces a distance function in the group I by

$$d_I(i, j) = d_G(\eta(i), \eta(j)). \tag{29}$$

We can now define the discrete transform

$$\Phi_f : I \to \mathcal{C} : j \mapsto \langle f, \pi_D(g_j)\psi \rangle. \tag{30}$$

In the case of color images, if the image is divided in three independent channels, we will have three seprate multidimensional arrays of coefficients, while in the case of a color algebra we will have a single multidimensional array whose entries are colors. In the following, we will consider the latter case, since the former can easily be reduced to it.

Thanks to the distance d_I, this transform has a metric structure just like the continuous transform. Note that the functions $\psi_j = \pi_D(g_j)\psi$ will not in general form an orthogonal basis of $L^2(\mathbf{R}^2)$, but an overcomplete frame [8].

An image can be completely represented by the set of coefficients of its transform. As we will see in the following section, this representation is very convenient for defining metrics in the image space, but it is too costly in terms of storage space for applications in image databases. For instance, applying the affine group to an image of size 128×128 will result in about $21,000$ coefficients, each of which is a color and therefore is represented by three numbers.

Each coefficient in the image representation is defined by the element $g \in G$ (its "position" in the group) and its color $c \in \mathcal{C}$. Therefore, a coefficient in the representation can be seen as a point $\kappa = (g, c) \in G \times \mathcal{C}$. Since both G and $\mathcal{C}$ are metric spaces, so is $G \times \mathcal{C}$. Therefore, given two coefficients κ_1 and κ_2 it is possible to define their distance $d(\kappa_1, \kappa_2)$. With this distance function, it is possible to apply vector quantization to the set of coefficients and obtain a more compact representation [15]. In our tests, we apply Vector Quantization to represent the image with a number of coefficients between 100 and 300.

5.1 Distance Functions

The representation introduced in the previous section gives us the necessary general feature space. In order to obtain a complete foundation for an image search engine, it is now necessary to define a suitably comprehensive class of distances into this space. For the sake of simplicity, we will begin by considering representations obtained from the continuous group G. In this case, the representation of an image is a function $f : G \to \mathcal{C}$, and the *graph* of the representation if a subset of $G \times \mathcal{C}$. Both G and $\mathcal{C}$ are metric spaces. If g_{ij} is the metric tensor of G interpreted as a Riemann manifold, and c_{uv} is the metric tensor of $\mathcal{C}$, then the metric of $H = G \times \mathcal{C}$ is

$$h = \begin{bmatrix} g & 0 \\ 0 & c \end{bmatrix} \tag{31}$$

and the distance between two points of H is

$$d(p, q) = \int_p^q \left(\sum_{rs} h_{rs} \dot{u}^r \dot{u}^s \right)^{\frac{1}{2}} dt \tag{32}$$

where the integral is computed on a geodesic $u(t)$ between p and q. In the discrete case $d(p, q)$ is the distance between two coefficients of the transform.

The distance between a point $x \in H$ and a set $A \subseteq H$ is defined as:

$$d(x, A) = \inf\{d(x, a) : a \in A\} \tag{33}$$

and the distance between two sets $A, B \subseteq H$ as

$$d(B, A) = \frac{1}{\|B\|} \int_B w(x) d(x, A) dx \tag{34}$$

where $w(x) > 0$ and $\int w(x)dx = 1$ is a weighting function that we will use in the following to manipulate the distance function. Note that with this definition the distance is not symmetric. For a justification of this, see [19, 23]. This distance can be used to measure dissimilarities between images.

An important property of this class of distances is given by the following theorem (proved in [15]):

Theorem 5.1 *Let H be a subgroup of G, and assume that the metric tensor in G is such that the corresponding distance function is*

$$d : G/H \times G/H \to \mathbf{R}^+ \tag{35}$$

If $f_1, f_2 : G \to \mathcal{C}$ are two functions such that

$$\exists h \in H : \forall g \in G f_1(g) = f_2(h \cdot g) \tag{36}$$

then $d(f_1, f_2) = 0$.

In other words, we can select a subgroup of transformations such that two images obtained one from the other through a transformation in the subgroup have distance 0.

By acting on the tensors g and c and the function w it is possible to define a number of different similarity criteria. Consider the following:

- If we are interested in the global color of an image, we will define the distance in G to be invariant to translation, and the metric tensor to be non-null only at low frequencies.
- If we are interested in the structure of an image, we can define the tensor g to be non-null only at high frequencies, and the tensor c to depend only on the brightness of the image.
- To search for a certain object, we define the tensor to be nonzero only in the region of the object in the query image, and enforce translation invariance.
- It is possible to enforce "quasi invariance:" the measure is not exactly invariant to a given transformation, but the distance increases very slowly when the images change. Quasi-invariance gives often more robust results than complete invariance.

More importantly, it is possible to parameterize the tensors g and c and the function w and define distances based on user interaction [15].

It is possible to extend the distance computation to the approximate representation. Let us start with an idealized case. In vector quantization, an image is represented by a set of Voronoi polygons that cover all G. Assume that the Voronoi polygons of the two images coincide on the subspace G, and that they differ only in color.

We have the following:

Definition 5.1 *Let $V = \{S_1, \ldots, S_V\}$ the set of Voronoi polygons of an image, $S_k \in V$ and $a \in S_k$. The point a is at distance d from the border of S_k if*

$$d = \inf \{d_I(a, b) : b \notin S_k\}. \tag{37}$$

We will use the notation $\delta(a) = d$

Definition 5.2 *Let S_A and S_B two overlapping Voronoi polygons, and let the distance between the colors of these polygons be Δ. A point $a \in S_A$ is internal if $\delta(a) > \Delta$, and is a boundary point otherwise. The interior of S_A, $\underline{S}_A$, is the set of internal points of S_A, and the border of S_A, $\overline{S}_A$, is the set of boundary points.*

Lemma 5.1 *Let S_A and S_B be two overlapped Voronoi polygons with color distance Δ. If a is an internal point of S_A, then*

$$d(a, S_B) = \Delta \tag{38}$$

The distance between S_A and S_B is:

$$\begin{aligned} d(S_A, S_B) &= \frac{1}{\|S_A\|} \sum_{a \in S_A} d(a, S_B) \qquad (39) \\ &= \frac{1}{\|S_A\|} \left[\sum_{a \in \underline{S}_A} d(a, S_B) + \sum_{a \in \overline{S}_A} d(a, S_B) \right] \end{aligned}$$

If $\|\overline{S}_A\| \ll \|S_A\|$, then

$$d(S_A, S_B) \approx \Delta \tag{40}$$

(see [15]).

The assumption that the Voronoi polygons are overlapping is obviously false. Given a Voronoi polygon V_A belonging to the image A, however, it is possible to compute its distance from image B by doing some approximation:

- Assume that only a fixed number N of Voronoi polygons of image B overlap V_A, and that the area by which $V_{B,i}$ overlaps V_A depends only on the distance between the respective centroids.

- Find the N polygons of B whose centroids are closer to that of V_A, and compute the distance d_i between their centroids and that of V_A. Let $D = \sum_i d_i$
- Let Δ_i be the distance between the color of V_A and the color of $V_{B,i}$.
- Compute $d(V_A, B) \approx \frac{1}{D} \sum_i d_i \Delta_i$

In this way, we can compute (approximately) the distance between one of the Voronoi polygons of image A and image B. If $\mathcal{V}_A$ is the set of all Voronoi polygons that define A, then the approximate distance betwen A and B is given by:

$$d(A, B) = \frac{1}{|\mathcal{V}_A|} \sum_{V \in \mathcal{V}_A} w_V d(V, B) \tag{41}$$

with $w_V > 0$ and $\sum_{V \in \mathcal{V}_A} w_V = 1$.

6 Query and Projection Operators

The two operators that most characterize the exploratory approach are the projection operator ϕ, and the query operator q. The first operator must project the images from the query space to the image space in a way that represent as well as possible the current database similarity (which, as we have argue, is the origin root of database semantics). The second operator must generate a metric for the query space that reflects as well as possible the similarity relation that the user communicated by placing images on the interface.

6.1 Projection Operator

The projection operator in its entirety is written as $\phi(x_I^k; f_\xi) = (X_I^\Psi, \emptyset)$. For the present purposes, we can ignore the set of labels, and we can remove the explicit reference to the metric f_ξ, so we can write $\phi(x_I) = X_I$. The distance between images x_I and x_J in the query space is $f(x_I, x_J; \xi)$, while for the display space we assume that the perception of "closeness" is well represented by the Euclidean distance: $d(X_I, X_J) = E(X_I, X_J) = \left[\sum_\Psi (X_I^\Psi - X_J^\Psi)^2\right]^{1/2}$.

As mentioned in the previous sections, we will not display the whole database, but only the P images closest to the query (with, usually, $50 \leq P \leq 300$). Let $\mathcal{I}$ be the set of indices of such images, with $|\mathcal{I}| = P$. The operator needs not be concerned with all the images in the database, but only with those that will be actually displayed.

First, we query the database so as to determine the P images closer to the query. We imagine to attach a spring of length $E(X_I, Y_I) = E(\phi(x_I), \phi(x_J))$ between images I and J. The spring is attached only if the distance between I and J in the feature space is less than a threshold τ. If the distance is greater than τ, the two images are left disconnected. That is, their distance in the display space is left unconstrained. The use of the threshold τ can be shown to create a more "structured" display space []. The operator ϕ is the solution of the

following optimization problem

$$\phi = \min_{\psi:\mathcal{F}\to\mathcal{D}} \arg \sum_{\substack{I,J \in C \\ E(\phi(x_I),\phi(x_J)) < \tau}} (f(x_I, x_J; \xi) - E(\psi(x_I), \psi(x_J)))^2 \tag{42}$$

In practice, of course, the operator ϕ is only defined by the positions in which the images are actually projected, so that the minimization problem would rewrite as

$$\min_{X_I : I \in \mathcal{I}} \sum_{\substack{I,J \in C \\ E(X_I,X_J) < \tau}} (f(x_I, x_J; \xi) - E(X_I, X_J))^2 \tag{43}$$

Standard optimization techniques can be used to solve this problem. In our prototype we use the simplex method described in [11].

6.2 Query Operator

The query operator solves the dual problem of deriving the metric of the query space from the examples that the user gives in the display space. Let us assume that the user has selected, as part of the interaction, a set D of images that are deemed relevant. The relative position of the images in the display space (as determined by the user) gives us the values $q_{IJ} = E(X_I, X_J)$, $i, j \in D$. The determination of the query space metric is then reduced to the solution of the optimization problem

$$\mathcal{E} = \min_{\xi} \sum_{I,J \in D} (f(x_I, x_J; \xi) - q_{IJ})^2 \tag{44}$$

Note that this is the same optimization problem as in (43), with the exception of the set of images that are considered relevant (in this case it is the set D of selected images, rather than the set of displayed images), and the optimization is done over ξ rather than over the projection.

The latter difference is particularly relevant, since in general it makes the problem severely underconstrained. In most cases, the vector ξ can contain a number of parameters of the order of 100, while the user will select possibly only a few images. In order to regularize the problem, we use the concept of *natural distance*.

We define one natural distance for every feature space, the intuition being that the natural distance is a uniform and isotropic measure in a given feature space. The details of the determination of the natural distance depend on the feature space. For the remainder of this section, let N be the natural distance function in a given feature space $\mathcal{F}$, and let $\Delta_{\mathcal{F}}$ be a distance functional defined on the space of distance functions defined on $\mathcal{F}$. In other words, let $\Sigma_{\mathcal{F}}(f)$ be the logical statement

$$\Sigma_{\mathcal{F}}(f) = (\forall x, y, z \in \mathcal{F}\,(d(x,y) \geq 0 \wedge d(x,x) = 0 \wedge d(x,y) = d(y,x) \wedge d(x,y) \leq d(x,z) + d(z,y))) \tag{45}$$

(i.e. $\Sigma_{\mathcal{F}}(f)$ is true if f is a distance function in the space $\mathcal{F}$). Let

$$D^2(\mathcal{F}) = \{f : \mathcal{F} \times \mathcal{F} \to \mathbf{R} : \Sigma_{\mathcal{F}}(f)\} \tag{46}$$

The set of distance functionals in $D^2(\mathcal{F})$ is then

$$\left\{\Delta_{\mathcal{F}} : D^2(\mathcal{F}) \times D^2(\mathcal{F}) \to [0,1)\right\} \tag{47}$$

Note that for the functions $\Delta_{\mathcal{F}}$ we do not require, at the present time, the satisfaction of the distance axioms. That is, we do not require $\Sigma_{D^2(\mathcal{F})}(\Delta_{\mathcal{F}})$. Whether this assumption can be made without any negative effect is still an open problem.

If we have identified a function $\Delta_{\mathcal{F}}$ (we will return to this problem shortly), then the metric optimization problem can be regularized (in the sense of Tihonov [21]) as

$$\mathcal{E} = \min_{\xi} \sum_{I,J \in D} \left(f(x_I, x_J; \xi) - q_{IJ}\right)^2 + w\Delta_{\mathcal{F}}(f(\cdot,\cdot;\xi), N) \tag{48}$$

that is, we try to find a solution that departs as little as possible from the natural distance in the feature space.

For the definition of the function $\Delta_{\mathcal{F}}$, an obvious candidate is the square of the L_2 metric in the Hilbert space of distance functions:

$$\Delta_{\mathcal{F}}(f) = \int_{x,y} \left(f(x,y;\xi) - N(x,y)\right)^2 dx, dy \tag{49}$$

In practice, the integral can be reduced to a summation on a sample taken from the database, or to the set of images displayed by the database.

An alternative solution can be determined if the natural distance N belongs to the family of distance functions $f(\cdot,\cdot;\xi)$ that is, if there is a parameter vector ξ_0 such that, for all x, y, $f(x,y;\xi_0) = N(x,y)$, and if the function $r(\xi) = f(x,y;\xi)$ is regular in ξ_0 for all x and y. Note that, since the only thing that changes in the family of distances f is the parameters vector ξ, we can think of $\Delta_{\mathcal{F}}(f)$ as a function of the parameters: $\Delta_{\mathcal{F}}(\xi)$. Because of the regularity hypothesis on f, $\Delta_{\mathcal{F}}(\xi)$ is also regular in ξ_0. In this case, we can write

$$\begin{aligned}
\Delta_{\mathcal{F}}(\xi) &= \Delta_{\mathcal{F}}(\xi_0) + \sum_{\mu} \left.\frac{\partial \Delta_{\mathcal{F}}}{\partial \xi^\mu}\right|_{\xi_0^\mu} (\xi^\mu - \xi_0^\mu) + \frac{1}{2} \sum_{\mu,\nu} \left.\frac{\partial^2 \Delta_{\mathcal{F}}}{\partial \xi^\mu \partial \xi^\nu}\right|_{\xi_0^\mu} (\xi^\mu - \xi_0^\mu)(\xi^\nu - \xi_0^\nu) + o(\|\xi - \xi_0\|^2) \\
&= \Delta_{\mathcal{F}}(\xi_0) + \nabla(\Delta_{\mathcal{F}})(\xi_0) \cdot (\xi - \xi_0) + (\xi - \xi_0)' H(\Delta_{\mathcal{F}})(\xi_0) \cdot (\xi - \xi_0) + + o(\|\xi - \xi_0\|^2)
\end{aligned} \tag{50}$$

where H is the Hessian of the function $\Delta_{\mathcal{F}}$. Since $\Delta_{\mathcal{F}}$ is the square of a regular function, we have $\nabla(\Delta_{\mathcal{F}})(\xi_0) = 0$. The Hessian contains elements of the form

$$\begin{aligned}
\frac{1}{2} \left.\frac{\partial^2 \Delta_{\mathcal{F}}}{\partial \xi^\mu \partial \xi^\nu}\right|_{\xi_0} &= \frac{1}{2} \int_{x,y} \left.\frac{\partial^2 (f(x,y;\xi) - f(x,y;\xi_0))w^2}{\partial \xi^\mu \partial \xi^\nu}\right|_{\xi_0} dxdy \\
&= \int_{x,y} \left[\frac{\partial f(x,y;\xi)}{\partial \xi^\mu} \frac{\partial f(x,y;\xi)}{\partial \xi^\nu} + \frac{\partial f(x,y;\xi)}{\partial \xi^\mu} \frac{\partial^2 f(x,y;\xi)}{\partial \xi^\mu \partial \xi^\nu}\right]_{\xi_0} dxdy \\
&= \int_{x,y} \left.\frac{\partial f(x,y;\xi)}{\partial \xi^\mu} \frac{\partial f(x,y;\xi)}{\partial \xi^\nu}\right|_{\xi_0} dxdy
\end{aligned} \tag{51}$$

If we define the matrix

$$\Phi_{\mu\nu} = \int_{x,y} \frac{\partial f(x,y;\xi)}{\partial \xi^\mu} \frac{\partial f(x,y;\xi)}{\partial \xi^\nu}\bigg|_{\xi_0} dxdy \tag{52}$$

we have a regularization problem of the type

$$\mathcal{E} = \min_{\xi} \sum_{I,J \in D} \left(f(x_I, x_J; \xi) - q_{IJ}\right)^2 + w(\xi - \xi_0)' \Phi (\xi - \xi_0). \tag{53}$$

This form is particularly convenient since the matrix Φ depends only on the natural distance of the feature space, and therefore can be precomputed.

7 The Interface at Work

We have used the principles and the techniques introduced in he previous section for the design of our database system El Niño[18, 20].

The feature space is generated with a discrete wavelet transform of the image. Depending on the transformation group that generates the transform, the space can be embedded in different manifolds. If the transformation is generated by the two dimensional affine group, then the space has dimensions x, y, and scale, in addition to the three color dimensions R, G, B. In this case the feature space is diffeomorphic to $\mathbb{R}^6$.

In other applications, we generate the transform using the Weyl-Heisenberg group [15, 22], obtaining transformation kernels which are a generalization of the Gabor filters [7]. In this case, in addition to the six dimensions above we have the direction θ of the filters, and the feature space is diffeomorphic to $\mathbb{R}^6 \times S^1$. After Vector Quantization, an image is represented by a number of coefficients between 50 and 200 (depending on the particular implementation of El Niño).

El Niño uses a class of metrics known as Fuzzy Feature Contrast (FFC) model [19], and derived from Tversky's similarity measure [23]. Consider an n-dimensional feature space (in our case $n = 6$ for the affine group, and $n = 7$ for the Weyl-Heisenberg group). We define two types of *membership functions*, the first of which is used for linear quantities, the second for angular quantities, like the angle of the Weyl-Heisenberg group.

Let x^i and y^i be two elements in this feature space. In other words, x^i and y^i are the components of one of the coefficients that define images X and Y. Define the following functions:

$$\mu_{\alpha,\beta}(x) = \frac{1}{1 + \exp(-\alpha(x - \beta))} \tag{54}$$

and

$$\eta_{\alpha,\beta}(x) = |\cos(2(x - \beta))|^\alpha, \tag{55}$$

and the quantities

$$p^i = \begin{cases} \mu_{\alpha^i,\beta^i}(x^i) - \mu_{\alpha^i,\beta^i}(y^i) & \text{if } x^i, y^i \text{ are linear features} \\ \eta_{\alpha^i,\beta^i}(x^i - y^i) & \text{if } x^i, y^i \text{ are angular features} \end{cases} \tag{56}$$

$$q^i = \begin{cases} \min\{\mu_{\alpha^i,\beta^i}(x^i), \mu_{\alpha^i,\beta^i}(y^i)\} & \text{if } x^i, y^i \text{ are linear features} \\ \eta_{\alpha^i,\beta^i}(x^i - y^i) & \text{if } x^i, y^i \text{ are angular features} \end{cases} \tag{57}$$

The distance between the two coefficients is defined as

$$T(x,y) = n - \sum_i \left(q^i - \xi|p^i|\right) \tag{58}$$

where $\zeta > 0$ is a weighting coefficient.

The determination of the distance between two coefficients depends on 13 parameters for the six-dimensional feature space and on 15 parameters for the seven-dimensional feature space (viz. the α^i's, the β^i's, and ζ). These distances must be inserted in the distance function (41). For an image represented by 100 coefficients, this would bring the total number of parameters to up to 1,500. This large number of parameters makes the regularization problem unmanageable. In order to reduce this number, we have collected the coefficients in 10 bins, depending on their location in the coefficient space, and enforce equality of parameters within a bin. This brings the number of independent parameters to less than 150. These parameters are set by the query operator based on the user interaction by solving the optimization problem (48).

In El Niño we use two types of image selections, represented as a rectangle around the image and as a cross, respectively. The difference between the two is in the type of parameters that they contributer to determine. The parameters β^i individuate a point in the feature space around which the query is centered. All the other parameters determine the shape of the space centered around this point. We noticed early on in our tests that users would like to select images that are not relevant for their query in order to give the database a distance reference between the query images and the rest of the database. Therefore, we introduced the "cross-selected" images that contribute to the adaptation of all the parameters except the β^i.

The interface of El Niño is shown in Fig. 6 with a configuration of random images displayed. In a typical experiment, subjects were asked to look for a certain group of car images, that look like the image in Fig. 7 (not necessarily that particular one). The car was showed to the subjects on a piece of paper before the experiment started, and they had no possibility of seeing it again, or to use it as an example for the database. This procedure was intended to mimick in a controlled way the vague idea of the desired result that users have before starting a query. In the display of Fig. 6 there are no cars like the one we are looking for, but there are a couple of cars. The user selected one (the subjectively most similar to the target) and marked another one with a cross. The second car image will be used to determine the similarity measure of the database, but it will not be considered a query example (the database will not try to return images similar to that car).

After a few interaction, the situation is that of Fig. 8 (for the sake of clarity, from now on we only show the display space rather than the whole interface). Two images of the type that we are looking for have appeared. They are selected and placed very close to one another. A third car image is placed at a certain distance but excluded from the search process. Note that the selection process is being refined as we proceed. During some of the previous interactions, the car image that is now excluded was selected as a positive example because, compared to

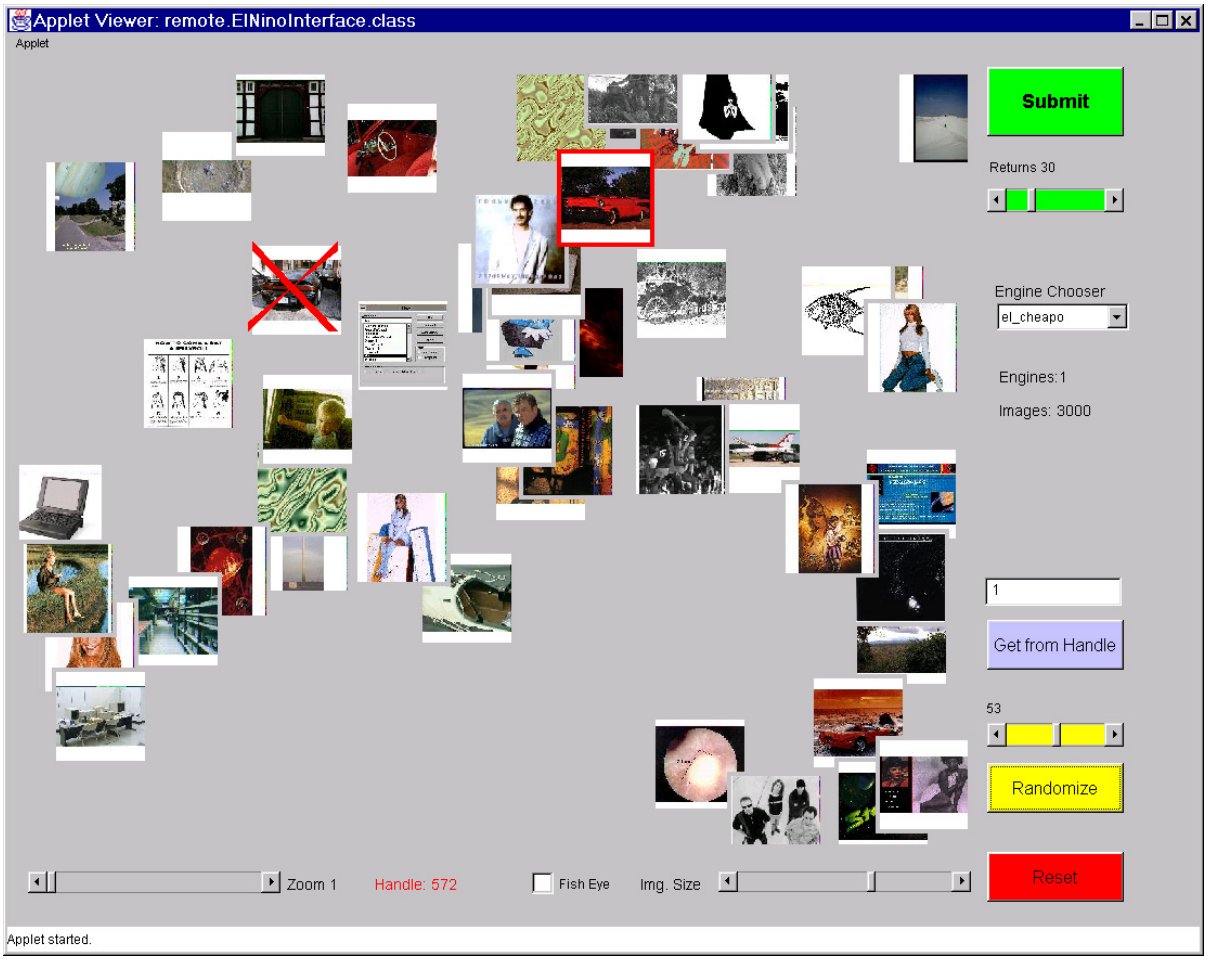

Figure 6: The interface of El Niño with the beginning of a search process.

what it was presented at the time, it was relatively similar to our target. Now that we are "zeroing in" to the images that we are actually interested in, the red car is no longer similar to what we need.

At the next iteration, the situation is that of Fig. 9. At this time we have a number of examples of the images we are looking for. Further iterations (e.g. with the selection represented in the figure) can be used to obtain more examples of that class of images. Note that the "negative" examples are placed much farther away from the positive examples than in the previous case. This will lead to a more discriminating distance measure which, in effect, will try to zoom in the class of images we are lookig for.

During this interaction, the subjects' idea of what would be an answer to the query changed continuously as they learned what the database had to offer, and redefined their goals based on what they saw. For instance, some of the positive examples that the subjects used at the beginning of the query—when their ideas were more generic and confused—are not valid later

Figure 7: One of our target images.

on, when they have a more precise idea of what they are looking for.

This simple experiment is intended only to give a flavor of what interaction with a database entails, and what kind of results we can expect. Evaluation of a highly interactive system like El Niño is a complex task, which could be the subject of a paper of its own. The interested reader can find some general ideas on the subject in [17].

8 Conclusions

In this paper we have defined a new model of interface for image databases. The motivation for the introduction of this model comes from an analysis of the semantics of images in the context of an image database. In traditional databases, the meaning of a record is a function from the set of queries to a set of truth values. The meaning of an image, on the other hand, is a function from the cartesian product of the feature space times the set of queries to the positive real values. This definition embodies the observation that the meaning of an image can only be revealed by the contraposition of an image with other images in the feature space.

These observations led us to define a new paradigm for database interfaces in which the role of the user is not just asking queries and receiving answer, but a more active *exploration* of the image space. The meaning of an image is *emergent*, in the sense that it is a product of the dual activities of the user and the database mediated by the interface.

We have proposed a model of interface for active exploration of image spaces. In our interface, the role of the database is ot *focus* the attention of the user on certain relation that, given the current database interpretation of image meanings, are relevant. The role of the user is exactly the same: by moving images around, the user focuses the attention of the database on certain relations that, given the user interpretation of meaning, are deemed important.

Our interface is defined formally as a set of operators that operate on three spaces: the *feature space*, the *query space*, and the *display space*. We have formalized the work of an interface as the action of an operators algebra on these spaces, and we have introduced a possible implementation of the most peculiar operators of this approach.

Finally, a word on performance. Evaluating the performance of an interactive system like this is a rather complex task, since the results depend on a number of factors of difficult

characterization. For instance, a system with a more efficient indexing (faster retrieval) but a weaker presentation can require more iterations—and therefore be less efficient—than a slower system with a better interface. A complete characterization of the performance of the system goes beyond the scope of this paper, but we can offer a few observations. An iteration in a system with our interface entails two steps: a k-nearest neighbors query to determine the k images that will be presented to the user, and the solution of an optimization problem to place them in the interface. The optimization problem is $O(k^2)$, but its complexity is independent of the number of images in the database (it depends only on the number of images that will be shown). Therefore, we don't expect it to be a great performance burden, not to present scalability problems. Early evaluations seem to support these conclusions.

References

[1] M. Caliani, P. Pala, and A. Del Bimbo. Computer analysis of TV spots: The semiotics perspective. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Austin, TX*, 1998.

[2] Umberto Eco. *A Theory of Semiotics*. Indiana University Press, Bloomington, 197.

[3] E. H. Gombrich. *Art and Illusion. A study in the pshychology of pictorial representation.* Pantheon Books, 1965.

[4] Amarnath Gupta, Simone Santini, and Ramesh Jain. In search of information in visual media. *Communications of the ACM*, 40(12):35–42, December 1997.

[5] Johannes Itten. *The art of Color*. Reinhold Pub. Corp., New York, 1961.

[6] Charles E. Jacobs, Adam Finkelstein, and Savid H. Salesin. Fast multiresolution image querying. In *Proceedings of SIGGRAPH 95, Los Angeles, CA*. ACM SIGGRAPH, New York, 1995.

[7] C. Kalisa and B. Torrésani. N-dimensional affine Weyl-Heisenberg wavelets. *Annales de L'Instut Henri Poincarè, Physique thèorique*, 59(2):201–236, 1993.

[8] Shidong Li and Jr Healy, D.M. A parametric class of discrete gabor expansions. *IEEE Transactions on Signal Processing*, 44(2):201–211, 1996.

[9] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, 1996.

[10] Roger F. Gibson, Jr. *The Philosophy of W. V. Quine*. University Presses of Florida, Tampa St. Petersburg Sarasota, 1982.

[11] William H. Press, Brian P. Flannery, Saul A. Teulolsky, and William T. Vetterling. *Numerical Recipes, The Art of Scientific Computing*. Cambridge University Press, 1986.

[12] K. Ravi Kanth, Divyakant Agrawal, and Ambuj Singh. Dimensionality reduction for similarity searching in dynamic databases. In *Proceedings of the 1998 ACM SIGMOD conference*, pages 166–175, 1998.

[13] E. Riloff and L Hollaar. Text databases and information retrieval. *ACM Computing Surveys*, 28(1):133–135, 1996.

[14] Artur Sagle and Ralph Walde. *Introduction to lie Groups and Lie Algebras*. Academic Press, New York and London, 1973.

[15] Simone Santini. *Explorations in Image Databases*. PhD thesis, Univerity of California, San Diego, January 1998.

[16] Simone Santini. Distance preservation in color image transforms. In *Proceedings of SPIE Vol. 3972, Storage and Retrieval for Image and Video Databases VIII*, 2000.

[17] Simone Santini. Evaluation vademecum for visual information systems. In *Proceedings of SPIE Vol. 3972, Storage and Retrieval for Image and Video Databases VIII*, 2000.

[18] Simone Santini and Ramesh Jain. The "El Niño" image database system. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Florence, Italy*, June 1999.

[19] Simone Santini and Ramesh Jain. Similerity measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(9), 1999.

[20] Simone Santini and Ramesh Jain. User interfaces for emergent semantics in image databases. In *Proceedings of the 8th IFIP Working Conference on Database Semantics (DS-8), Rotorua (New Zealand)*, January 1999.

[21] A. N. Tihonov. Regularization of incorrectly posed problems. *Soviet Mathematical Doklady*, 4:1624–1627, 1963.

[22] B. Torrésani. Wavelets associated with representations of the affine weyl-heisenberg group. *Journal of Mathematical Physics*, 32(5):1273–1279, 1991.

[23] Amos Tversky. Features of similarity. *Psychological review*, 84(4):327–352, July 1977.

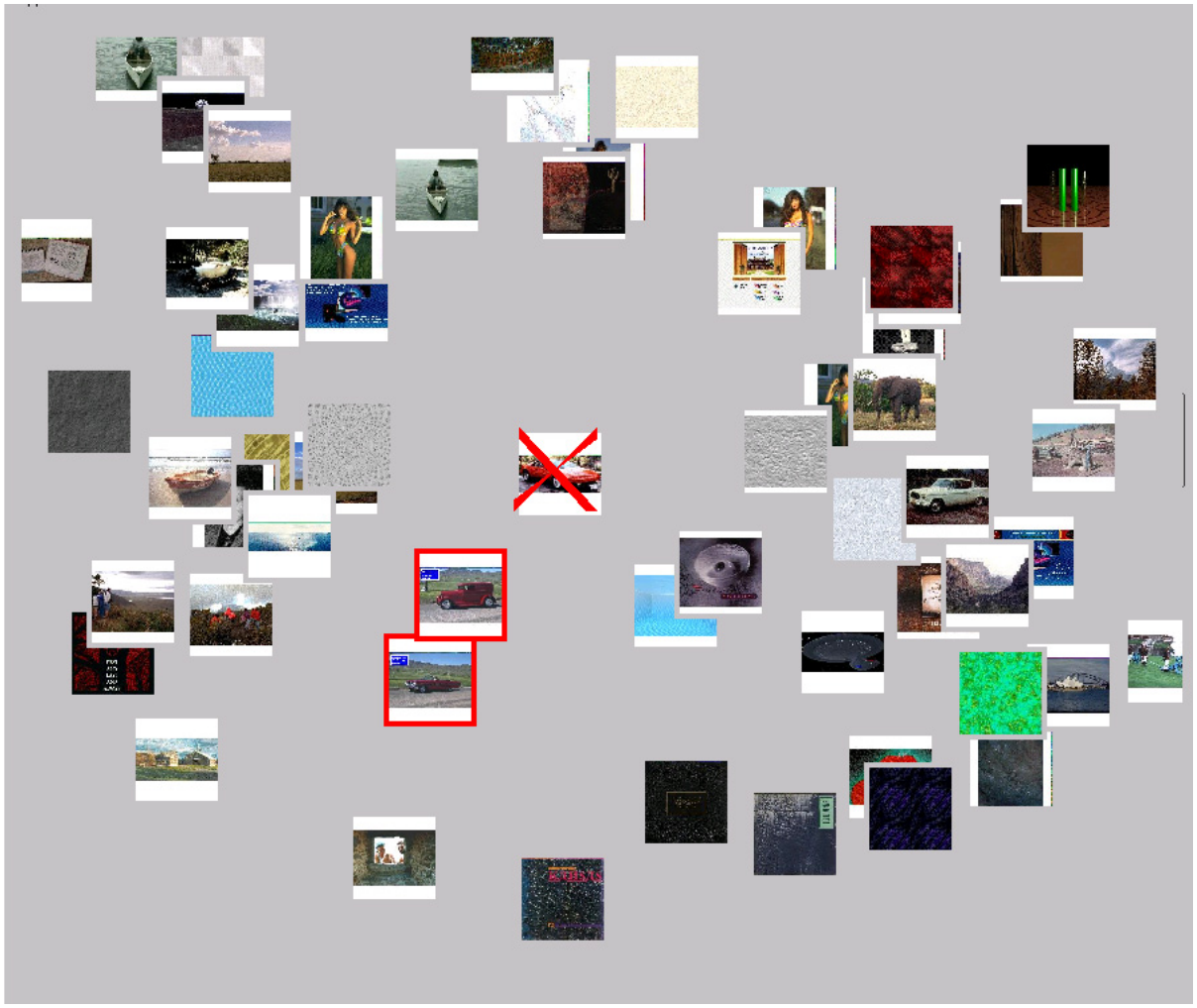

Figure 8: The interface of El Niño during the search process.

Figure 9: The interface of El Niño during the search process.