# Image Retrieval by Appearance

**S. Ravela**
Computer Vision Research Lab., Multimedia Indexing and Retrieval Group
Center for Intelligent Information Retrieval, University of Massachusetts at Amherst
ravela@cs.umass.edu
and
**R. Manmatha**
Multimedia Indexing and Retrieval Group
Center for Intelligent Information Retrieval, University of Massachusetts at Amherst
manmatha@cs.umass.edu

*Abstract*
A system to retrieve images using a syntactic description of appearance is presented. A multi-scale invariant vector representation is obtained by first filtering images in the database with Gaussian derivative filters at several scales and then computing low order differential invariants. The multi-scale representation is indexed for rapid retrieval. Queries are designed by the users from an example image by selecting appropriate regions. The invariant vectors corresponding to these regions are matched with those in the database both in feature space as well as in coordinate space and a match score is obtained for each image. The results are then displayed to the user sorted by the match score. From experiments conducted with over 1500 images it is shown that images similar in appearance and whose viewpoint is within 25 degrees of the query image can be retrieved with an average precision of 57.4%

## 1. INTRODUCTION

The goal of image retrieval systems is to operate on collections of images and in response to visual queries extract relevant images. The application potential for fast and effective image retrieval is enormous; ranging from database management in museums and medicine, architecture and interior design, image archiving, to constructing multi-media documents or presentations[5]. There are, however, several issues that must be understood before image retrieval is viable. Foremost among these is an understanding of what 'retrieval of relevant images' means. Relevance, for users of a retrieval system, is most likely associated with semantics. For example, a user might want all pictures of Christmas cards. A Christmas card has associated semantics that imply, Santa Claus, Christmas tree, reindeer, snow and so on. Encoding this semantic information into a general image retrieval system

This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623, in part by the United States Patent and Trademark Office and Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235 and in part by NSF Multimedia CDA-9502639 and NRaD Contract Number N66001-94-D-6054. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.

entails solving such problems as segmentation, recognition and automatic feature extraction. These are extremely hard problems that are as yet unsolved.

An alternative approach to the relevance problem comes from the observation that, in many cases attributes associated with an image when used together with some level of user input, correlate well with the kind of semantics that are desirable. For instance, one can provide an example of the Christmas tree and try to find other ones by texture. Or a user can provide a picture of Santa and find others by color, or shape or both. In simpler situations textual attributes have been associated with images. In particular, images are annotated with text and then retrieved using a text retrieval engine. This solution is limited because the variability and richness of images cannot be effectively captured by annotations within any reasonable effort. Recent work has focused directly on image content such as color [26; 25; 17], texture features[12; 4; 20; 13], shape[16; 1; 18; 24; 30] and combinations thereof[2; 7; 20].

In this paper images are retrieved using a characterization of the visual appearance of objects. Intuitively an object's visual appearance in an image is closely related to a description of the shape of its intensity surface. Appearance not only depends on the object's three-dimensional geometric shape, but also on its albedo, its surface texture, the view point from which it is imaged and a number of other factors. It is non-trivial to separate the different factors constituting an object's appearance and it is usually not possible to separate an object's three dimensional shape from the other factors. For example, the face of a person has a unique appearance that cannot just be characterized by the geometric shape of the 'component parts'. Similarly the shape of a car such as the one shown in Figure 1 is not just a matter of a geometric-shape outline of its 'individual parts'. In this paper we characterize the shape of the intensity surface of imaged objects and the term *appearance* will imply the phrase 'shape of the intensity surface'. The experiments conducted in this paper verify this association. That is, objects that appear to be visually similar can be retrieved by a characterization of the shape of the intensity surface.

Different representations of appearance have been used in object recognition [19; 23] and have been applied to specific types of retrieval such as face recognition [8; 29]. To the best of our knowledge the system presented here is the first attempt to characterize appearance to retrieve similar images and in this paper the development of Synapse (Syntactic Appearance Search Engine), an image database search engine, is documented. The approach taken here does not rely on image segmentation (manual or automatic) or binary feature extraction. Unlike some of the aforementioned methods, no training is required and objects can be embedded in different backgrounds. Using an *example image and user interaction to construct queries* Synapse retrieves similar images within small view and size variation in the order of their *similarity in syntactic appearance to a query.*

The claim is that, up to a certain order, the *local appearance* of the intensity surface (around some point) can be represented as responses to a set of scale parameterized Gaussian derivative filters. This set or vector of responses, called a feature vector, when computed with filters at a certain scale, and up to order N, completely and uniquely characterize the local jet[10] of order N, at that scale. Since the local jet generalizes to the Taylor series expansion of the underlying local intensity function, it is, therefore, argued that the local appearance of the intensity

function is represented to the order of expansion.

The proposed representation is also syntactic. This is because the filter responses are obtained solely from the signal content and without the use of "global context" or "symbolic interpretation". Further, the family of Gaussian filters are unique in their ability to describe the *scale-space* or *deep structure* [9; 11; 28; 3] of a function. Consequently, the change in appearance of an image due to a change in the viewing geometry can be computed by equivalently deforming the filter [14; 15]. In previous work it was demonstrated that feature vectors constructed using Gaussian derivative filters can be used to retrieve objects that are not only scaled versions of each other but also similar (in appearance) and within small view variations of one another [22].

In this paper an indexable strategy for image retrieval is developed using feature vectors that are constructed using combinations of the derivative filter outputs. These combinations yield a set of differential invariants [3] that are invariant to two-dimensional rigid transformations. Retrieval is achieved in two computational steps. During the off-line computation phase each image in the database is first filtered at sampled locations and then filter responses across the entire database are indexed. The run-time computation of the system begins with the user selecting an example image and marking a set of salient regions within the image. The responses corresponding to these regions are matched with those of the database and a measure of fitness per image in the database is computed in both feature space and coordinate space. Finally, images are displayed to the user in the order of fitness (or match score) to the query.

The ability for the user to construct queries by selecting regions is an important distinction between the approach presented here and elsewhere. The user can use her considerable semantic knowledge about the world to construct a query. Such semantic information is difficult to incorporate in a system. An example of query construction is shown in Figure 1, where the user has decided to find cars similar to the one shown and decides that the most salient part is a 'wheel'. It is clear that providing such interaction removes the necessity for automatic determination of saliency. In the car example, the user provides the context to search the database by marking the wheel and retrieved images mostly contain wheels. The association of wheels to cars is not known to the system, rather it is one that the user decides is meaningful. We believe that this natural human ability in selecting salient regions must be exploited. Further, in a fast system, feedback can be quickly obtained by browsing through the results (see Figure 2). If the results are unsatisfactory a new query can be designed.

The remainder of this paper is organized as follows. In Section 2 the current literature is surveyed for related work. In Section 3 we develop the notion of appearance, the construction of differential invariant features and their final storage as indices. This sums up the off-line computation. In Section 4 the run-time component of the system is examined. Finally, experimental results containing examples, recall, precision and execution time are presented in Section 5.

## 2. RELATED WORK

Several authors have tried to characterize the appearance of an object via a description of the intensity surface. In the context of object recognition [19] represent

Fig. 1. *Allowing the user to construct queries by selecting the box shown*

the appearance of an object using a parametric eigen space description. This space is constructed by treating the image as a fixed length vector, and then computing the principal components across the entire database. The images therefore have to be size and intensity normalized, segmented and trained. Similarly, using principal component representations described in [8] face recognition is performed in [29]. In [27] the traditional eigen representation is augmented by using most discriminant features and is applied to image retrieval. The authors apply eigen representation to retrieval of several classes of objects. The issue however is that these classes are manually determined and training must be performed on each. The approach presented in this paper is different from all the above because eigen decompositions are not used at all to characterize appearance. Further the method presented uses no learning, does not depend on constant sized images and deals with embedded backgrounds and heterogeneous collections of images using local representations of appearance.

The use of Gaussian derivative filters to represent appearance is motivated by their use in describing the spatial structure [10] and its uniqueness in representing the scale-space of a function [11; 9; 31; 28] and the fact that the principal component of images are best described as Gaussians and their derivatives [6]. That is there is a natural decomposition of images into Gaussians and their derivatives. The use of invariant transformations of Gaussians is borrowed from descriptions provided by [3]. In [21] tracking is done by using a vector of Gaussian derivatives which are indexed. [23] use use indexed differential invariants for object recognition. We also use index on differential invariants but there are several differences between the approach presented here and theirs. First, in this work only the low two orders are used, which is more relevant to retrieving similar images (see section 3) while they use nine-invariants. Second, their indexing algorithm depends on interest point detection and is, therefore, limited by the stability of the interest operator. We on the other hand sample the image. Third, the authors do not incorporate multiple scales into a single vector whereas here three different scales are chosen. In addition

the index structure and spatial checking algorithms differ. Schmid and Mohr apply their algorithm primarily to the problem of object recognition, do not allow for the user to determine saliency and therefore have not applied their algorithm to retrieving similar images.

The earliest general image retrieval systems were designed by [2; 20]. In [2] the shape queries require prior manual segmentation of the database which is undesirable and not practical for most applications. There has been other work on shape using a description of polygons [16] and curves [1; 18; 24; 30]. Of particular interest is work by Mokhtarian et. al. where they use the curvature scale-space to represent shape. Texture based image retrieval is also related to the appearance based work presented in this paper. Using Wold modeling, in [12] the authors try to classify the entire Brodatz texture and in [4] attempt to classify scenes, such as city and country. Of particular interest is work by [13] who use Gabor filters to retrieve texture similar images, without user interaction to determine region saliency.

There have been attempts to combine different attributes. Shape, color and texture have been combined in [2; 20]. This combination is not transparent to the user; instead she must decide how to weight the different attributes. In [7] color and shape are combined by defining a composite metric over both.

## 3. SYNTACTIC REPRESENTATION OF APPEARANCE

This section begins by making explicit the notion of appearance and the uniqueness of Gaussian derivative filters therein. As a result, a representation, that is a multi-scale feature vector can be constructed by filtering an image with a set of Gaussian derivative filters. The multi-scale feature vector may b transformed so that the elements within this vector are invariant invariant to 2D rigid transformations. This transformed feature vector is called the multi-scale invariant vector. Then a scheme for indexing multi-scale invariant vectors computed over the entire image database is presented. This completes all the steps of the off-line computation described in the introduction.

### 3.1 Characterization of Appearance

A function can be locally characterized by it's Taylor series expansion provided the derivatives at the point of expansion are well conditioned. The intensity function of the image , on the other hand, need not exhibit continuity at the point of expansion. However, it is well known that the derivative of a, possibly discontinuous, function can be made well posed if it is convolved with the derivative of a smooth test function. Consider the normalized Gaussian as a choice for the smooth test function which in two dimensions is defined as,

$$G\left(\mathbf{x},\sigma\right)=\frac{1}{2\pi\sigma^{2}}e^{\frac{-\mathbf{x}^{2}}{2\sigma^{2}}} \tag{1}$$

where $\mathbf{x}\in\Re^{2}$, $\sigma$ is the scale of the Gaussian. Then the derivatives of the image $I_{\sigma}\left(\mathbf{x}\right)=\left(I\star g\right)\left(\mathbf{x},\sigma\right),\mathbf{x}\in\Re^{2}$,are well conditioned for some value of $\sigma$. This is written as

$$I_{i_{1}\ldots i_{n},\sigma}\left(\mathbf{x}\right)=\left(I\star G_{i_{1}\ldots i_{n}}\right)\left(\mathbf{x},\sigma\right)$$

$$G_{i_{1}\ldots i_{n}}=\frac{\delta^{n}}{\delta_{i_{1}}\ldots\delta_{i_{n}}}G$$

and $i_k = x_1 \dots x_D, k = 1 \dots n$.

The local N-jet of $I(\mathbf{x})$ at scale $\sigma$ and order $N$ is defined as the set [10](cite Koenderink):

$$J^N[I](\mathbf{x},\sigma) = \{I_{i_1 \dots i_n,\sigma} | n = 0 \dots N\} \tag{2}$$

It can be observed that $\lim_{N\to\infty} J^N[I](\mathbf{x},\sigma)$, bundles all the derivatives required to fully specify the Taylor expansion of $I_\sigma$ up to derivatives of order N. Thus, for any order $N$, the local N-jet at scale $\sigma$ locally contains all the information required to reconstruct $I$ at the scale of observation $\sigma$ up to order $N$. This is the primary observation that is used to characterize appearance. That is, up to any order the derivatives locally characterize the shape of the intensity surface, i.e. appearance, to that order. From the experiments shown in this paper it is also observed that this representation can be used to retrieve images that appear visually similar.

As a practical example consider the local 2-jet of an image $I(\mathbf{p}), p = \langle x, y\rangle \in \Re^2$, at scale $\sigma$.

$$J^2[I](\mathbf{p},\sigma) = \{I_\sigma(\mathbf{p}), I_{x,\sigma}(\mathbf{p}), I_{y,\sigma}(\mathbf{p}), I_{xx,\sigma}(\mathbf{p}), I_{xy,\sigma}(\mathbf{p}), I_{yy,\sigma}(\mathbf{p})\}^{1}$$

That is image $I$ is filtered with the first two Gaussian derivatives (and the Gaussian itself) in both $x$ and $y$ directions. Point $p$ is, therefore, associated with a *feature vector* of responses at scale $\sigma$.

The choice of the Gaussian as the smooth test function, as opposed to others, is motivated by the fact that it is unique in describing the scale-space or deep structure of an arbitrary function. A full review of scale-space is beyond the scope of this paper and the reader is referred to [31; 9; 3; 11; 28] for a study. Here some of the important consequences of incorporating scale space are considered. For increasing values of $\sigma$ the Gaussian filter admits a narrowing band of frequencies and $I$ will appear smoother. The scale-space of $I$ is simply $I_\sigma$, where $\sigma$ is the free variable. Similarly, the scale space of the derivatives of $I$ is the range of $I_{i_1 \dots i_n,\sigma}$ where $\sigma$ is the free variable. Scale-space has important physical interpretations. For example, as an object moves away from a camera (in depth) its image appears less structured and finer contrasts get blurred. The scale-space of the image of the object models this observation. By filtering an image with Gaussian derivative filters (up to some order) at several scales, the appearance of that image from different depths (from a camera) can be represented. An argument is therefore made for a *multi-scale feature vector* which describes the intensity surface locally at several scales. Multi-scale vectors represent appearance better than a single-scale vector. From a practical standpoint this means that mis-matches due to an accidental similarity can be reduced.

As a consequence of the scale-space description using Gaussian derivatives, image patches that are scaled versions of each other can also be compared in a straightforward manner. Consider two images $I_0$ and $I_1$ that are scaled versions of each other (but otherwise identical). Without loss of generality assume that the scaling is centered at the origin. That is $I_0(\mathbf{p}) = I_1(s\mathbf{p})$ Then the following relations hold

[1] $I_{yx} = I_{xy}$ and is therefore dropped

[14; 15]

$$\begin{array}{ll} I_0(\mathbf{p}) \star g(\cdot,\sigma) & = I_1(s\mathbf{p}) \star g(\cdot, s\sigma) \\ I_0(\mathbf{p}) \star g^{(k)}(\cdot,\sigma) & = I_1(s\mathbf{p}) \star g^{(k)}(\cdot, s\sigma) \\ \text{where, } g^{(k)}(\cdot,t) & = t^k g_{i_1 \ldots i_k}(\cdot,t) \end{array} \tag{3}$$

These equations state that if the image $I_s$ is a scaled version of $I_0$ by a factor $s$ then in order to compare any two corresponding points in these images the filters must also be stretched (i.e. scaled) by the same factor. For example, if a point $p_0$ is being compared with a point $p_1$ in images $I_0$ and $I_1$ where $I_1$ is twice the size of $I_0$, then the filter used to compute the response at $p_1$ must be twice that of $p_0$ for the responses to be equal.

The multi-scale approach is, therefore, a robust representation of appearance which may be used to directly compare images that are scaled versions of each other. From an implementation stand point a *multi-scale feature vector* at a point $p$ in an image $I$ is simply the elements of the vector:

$$\left\{ J^N[I](\mathbf{p},\sigma_1), J^N[I](\mathbf{p},\sigma_2) \ldots J^N[I](\mathbf{p},\sigma_k) \right\} \tag{4}$$

for some order $N$ and a set of scales $\sigma_1 \ldots \sigma_k$. In practice the zeroth order terms are dropped to achieve invariance to constant intensity changes.

A measure of similarity between two multi-scale vectors can be obtained by correlating them or computing the distance between the vectors. In earlier work [22] it was shown that multi-scale vectors can be used to retrieve images that are not only scaled versions of each other but also ones that are similar to the query. This was achieved by correlating the derivative feature vectors across scales using the scale shifting theorem presented above. An important observation from that work is that as images become more dissimilar (due to several reasons) their response vectors become less correlated, starting at the higher order. Thus, similar images, can be expected to be more correlated in their lower order than higher ones. As a consequence only the first two order derivatives were used. Likewise, in this paper the lower order derivatives are used. Similar arguments can be made for scales. As images get dissimilar, they can be expected to retain strong correlation only at large scales (lower spatial frequency). Further the range of scales over which they correlate well gets smaller. As a consequence, in this paper the multi-scale vector is computed at three different scales placed half an octave apart. This is discussed in the next sub-section. In general these parameters can be computed a priori for a large range of values (at the expense of disk space) and then left to the user to pick appropriate ones on at run-time. Or they can be conditioned in advance for a specific task at the expense of generality.

### 3.2 Multi-Scale Invariant Vectors

The limitation of using the derivatives directly in a feature vector is that it limits the viewing range (both out-of-plane and in-plane rotations of the image). This issue is partially addressed by transforming the multi-scale feature vector so that it is invariant to 2D rigid transformations.

Given the derivatives of an image $I$ *irreducible differential invariants*, that are invariant under the group of displacements can be computed in a systematic manner [3]. The term irreducible is used because other invariants can be reduced to a

combination of the irreducible set. The value of these entities independent of the choice of coordinate frame (up to rotations) for the low orders (two here) terms are enumerated.

The irreducible set of invariants up to order two of an image $I$ are:

$$\begin{array}{lll} d_0 & = I & \text{Intensity} \\ d_1 & = I_x^2 + I_y^2 & \text{Magnitude} \\ d_2 & = I_{xx} + I_{yy} & \text{Laplacian} \\ d_3 & = I_{xx}I_xI_x + 2I_{xy}I_xI_y + I_{yy}I_yI_y & \\ d_4 & = I_{xx}^2 + 2I_{xy}^2 + I_{yy}^2 & \end{array}$$

In experiments conducted in this paper, the vector, $\Delta_\sigma = \langle d_1, \ldots d_4 \rangle_\sigma$ is computed at three different scales. The element $d_0$ is not used since it is sensitive to gray-level shifts. The resulting multi-scale invariant vector has at most twelve elements. Computationally, each image in the database is filtered with the first five partial derivatives of the Gaussian (i.e. to order 2) at three different scales at uniformly sampled locations. Then the multi-scale invariant vector $D = \langle \Delta_{\sigma_1}, \Delta_{\sigma_2}, \Delta_{\sigma_3} \rangle$ is computed at those locations. The list of multi-scale vectors across the entire database is then indexed for rapid retrieval and is described next.

### 3.3 Indexing Invariant Vectors

The multi-scale invariant vector $D$ can be viewed as a fixed length record. A location across the entire database can be identified by the *generalized coordinates*, defined as, $c = (i, x, y)$ where $i$ is the image number and $(x, y)$ a coordinate within this image. The computation described in the previous sub-section generates an association between generalized coordinates and invariant vectors. This association can be viewed as a table $M : (i, x, y, D)$. The number of columns in this table are $3 + k$, where, $k$ is the number of fields in an invariant vector. Each row is simply the invariant vector corresponding to a generalized coordinate and the number of rows is the total number of invariant vectors across the entire database.

To find an invariant vector by coordinate, that is input a generalized coordinate and obtain the corresponding invariant vector, a simple look up in $M$ can be performed which is a constant time operation (if the dimensions of each image is known). However, to retrieve images, a 'find by value' functionality is needed, wherein, a query invariant vector can be found within $M$ and the corresponding generalized coordinate is returned. But this entails a linear search in $M$ which is extremely time consuming. The solution is to generate inverted files (or tables) for $M$, based on each field of the invariant vector and index them. Then the operation of 'find-by-value' can be performed in log time and is described below. To index the database by fields of the invariant vector, the table $M$ is split into $k$ smaller tables $M_1^{'} \ldots M_k^{'}$, one for each of the $k$ fields of the invariant vector. Each of the smaller tables $M_p^{'}, p = 1 \cdots k$ contains the four columns $(D(p), i, x, y)$. At this stage any given row across all the smaller tables contains the same generalized coordinate entries, as that in $M$. Then, each $M_p^{'}$ is sorted by it's first column and a binary tree structure on this column is generated. As a result, the entire database is indexed.

The following steps are needed to perform a find-by-value operation on a query invariant vector. Each field of the query vector is used to traverse through the corresponding index tree. Once a match is found, the generalized coordinate is

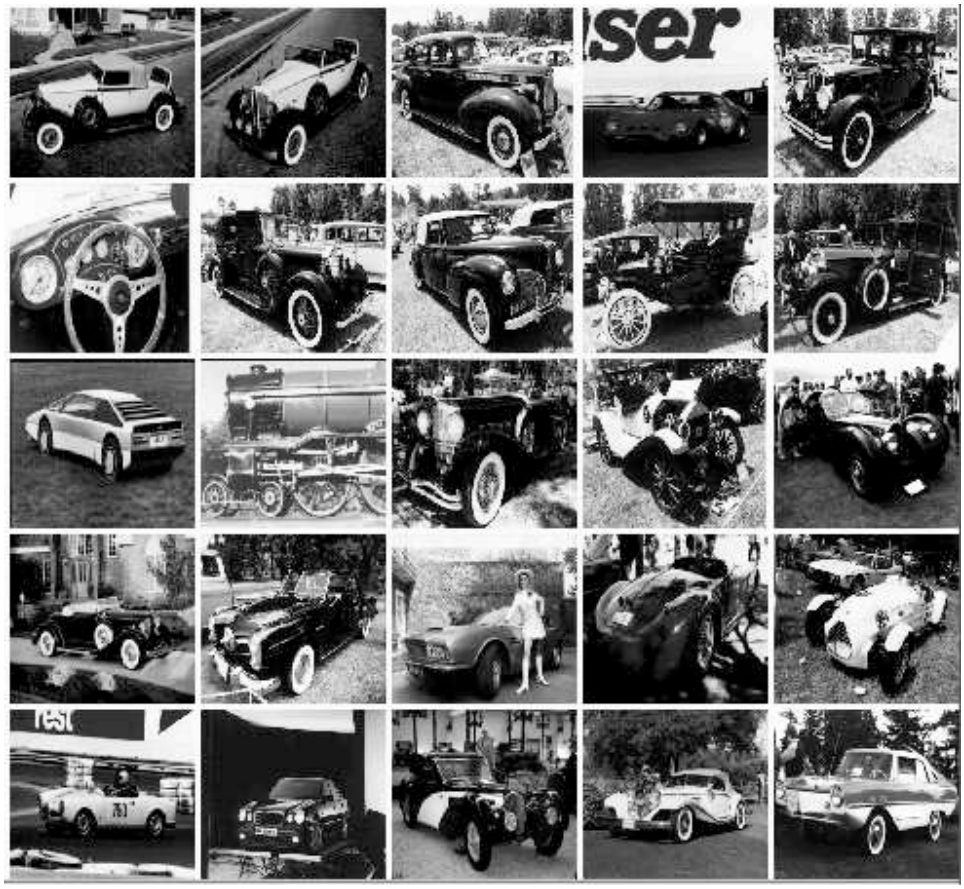

Fig. 2. *The results of the car query shown in Figure 1*

extracted. After all the $k$ fields complete their search successfully, $k$ generalized coordinates would have been extracted. If all of these are exactly the same then the find-by-value routine has succeeded.

The entire process of Off-line computation can be summarized in the following steps.

(1) Filter each image at uniformly sampled locations with Gaussian derivatives at several scales up to order two.
(2) Generate the multi-scale invariants at these points and hence the table $M$.
(3) Compute the inverted file $M_k^{'}$ for each key of the record across the entire database.
(4) Sort the inverted file by key value and create a binary index.

## 4. MATCHING INVARIANT VECTORS

Run-time computation begins with the user marking selecting regions in an example image. At sampled locations within these regions, invariant vectors are

computed and submitted as a query. The search for matching images is performed in two stages. In the first stage each query invariant is supplied to the 'find-by-value' algorithm and a list of matching generalized coordinates is obtained. In the second stage a spatial check is performed on a per image basis, so as to verify that the matched locations in an image are in spatial coherence with the corresponding query points. In this section the 'find-by-value' and spatial checking components are discussed.

### 4.1 Finding by Invariant Value

The multi-scale invariant vectors at sampled locations within regions of a query image can be treated as a list. The $n^{th}$ element in this list contains the information $Q_n = (D_n, x_n, y_n)$, that is, the invariant vector and the corresponding coordinates. In order to find by invariant value, for any query entry $Q_n$, the database must contain vectors that are within a threshold $t = (t_1 \dots t_k) > 0$. The coordinates of these matching vectors are then returned. This can be represented as follows. Let $p$ be any invariant vector stored in the database. Then $p$ matches the query invariant entry $D_n$ only if $D_n - t < p < D_n + t$. This can be rewritten as

$$\&_{j=1}^{k} \left[ D_n(j) - t(j) < p(i) < D_n(j) - t(j) \right]$$

where & is the logical and operator and $k$ is the number of fields in the invariant vector. To implement the comparison operation two searches can be performed on each field. The first is a search for the lower bound, that is the largest entry smaller than $D_n(j) - t(j)$ and then a search for the upper-bound i.e. the smallest entry larger than $D_n(j) + t(j)$. The block of entries between these two bounds are those that match the field $j$. In the inverted file the generalized coordinates are stored along with the individual field values and the block of matching generalized coordinates are copied from disk. To implement the logical-and part, an intersection of all the returned block of generalized coordinates is performed. The generalized coordinates common to all the $k$ fields are the ones that match query entry $Q_n$. The find by value routine is executed for each $Q_n$ and as a result each query entry is associated with a list of generalized coordinates that it matches to.

In practice, the fields over which the intersection operation is performed is a matter of experimentation. For example, for several queries and those listed here, the last two fields of the invariant vector are used. This corresponds to six field searches or twelve traversals through the index trees.

### 4.2 Spatial-Fitting

The association between a Query entry $Q_n$ and the list of $f$ generalized coordinates that match it by value can be written as

$$A_n = \langle x_n, y_n, c_{n_1}, c_{n_2} \dots c_{n_f} \rangle = \langle x_n, y_n, (i_{n_1}, x_{n_1}, y_{n_2}) \dots (i_{n_f}, x_{n_f}, y_{n_f}) \rangle$$

. Here $x_n, y_n$ are the coordinates of the query entry $Q_n$ and $c_{n_1} \dots c_{n_f}$ are the $f$ matching generalized coordinates. The notation $c_{n_f}$ implies that the generalized coordinate $c$ matches $n$ and is the $f^{th}$ entry in the list. Once these associations are available, a spatial fit on a per image basis can be performed. In order to describe the fitness measure, two definitions are needed. First, define the distance between

two query entries $m$ and $n$ as $\delta_{m,n} = (x_m - x_n)^2 + (y_m - y_n)^2$. Second, define the distance between any two generalized coordinates $c_{m_j}$ and $c_{n_j}$ that are associated with two query entries $m, n$ by:

$$\delta_{c_{m_j},c_{n_k}} = \left(x_{m_j} - x_{n_k}\right)^2 + \left(y_{m_j} - y_{n_k}\right)^2$$

Any image $u$ that contains two points (locations) which match some query entry $m$ and $n$ respectively are coherent with the query entries $m$ and $n$ only if the distance between these two points is the same as the distance between the query entries that they match. Using this as a basis, a binary fitness measure can be defined as

$$\mathcal{F}_{m,n}(u) = \begin{cases} 1 & \text{if } \exists j \exists k \mid \left|\delta_{m,n} - \delta_{c_{m_j},c_{n_k}}\right| \leq T, i_{m_j} = i_{n_k} = u, m \neq n \\ 0 & \text{otherwise} \end{cases}$$

That is, if the distance between two matched points in an image is close to the distance between the query points that they are associated with, then these points are spatially coherent (with the query). Using this fitness measure a match score for each image can be determined. This match score is simply the maximum number of points that together are spatially coherent (with the query). Define the match score by:

$$score(u) \equiv \overset{max}{m} S_m(u) \tag{5}$$

where, $S_m(u) = \sum_{n=1}^{f} \mathcal{F}(u)_{m,n}$. The computation of $score(u)$ is at worst quadratic in the total number of query points. The array of scores for all images is sorted and the images are displayed in the order of their score. $T$ used in $\mathcal{F}$ is a threshold and is typically 25% of $\delta_{m,n}$. Note that this measure not only will admit points that are rotated but will also tolerate other deformations as permitted by the threshold. It is placed to reflect the rationale that similar images will have similar responses but not necessarily under a rigid deformation of the query points.

### 4.3 Query Construction

The success of a retrieval in part depends on well designed queries. That implies that the user should be provided with a facility to design queries. Several other approaches in the literature take the entire feature set or some global representation over the entire image. While this may be reasonable for certain types of retrieval, it cannot necessarily be used for general purpose retrieval.

More importantly, letting the user design queries eliminates the need for detecting the salient portions on an object, and the retrieval can be customized so as to remove unwanted portions of the image. Based on the feedback provided by the results of a query, the user can quickly adapt and modify the query to improve performance.

## 5. EXPERIMENTS

The choice of images used in the experiments is based on a number of considerations. First it is general in that it doesn't reflect a bias towards any particular method, such as texture alone or shape alone. Second, it is expected that when

very dissimilar images are used the system should have little difficulty in ranking the images. For example, if a car query is used with a database containing cars and apes, then it is expected that cars would be ranked ahead of apes. This is borne out by the experiments done to date. Much poorer discrimination is expected if the images are much more 'similar'. For example, different species of apes should be harder to discriminate.

The database used in this paper has digitized images of cars, steam locomotives, diesel locomotives, apes, faces, people embedded in different background(s) and a small number of other miscellaneous objects such as houses. 1561 images were obtained from the Internet and the Corel photo-cd collection to construct this database.These photographs, were taken with several different cameras of unknown parameters, and, under varying but uncontrolled lighting and viewing geometry. Also, the objects of interest are embedded in natural scenes such as car shows, railroad stations, country-sides and so on.

Prior to describing the experiments, it is important to clarify what a correct retrieval means. A retrieval system is expected to answer questions such as 'find all cars similar in view and shape to this car' or 'find all faces similar in appearance to this one'. To that end one needs to evaluate if a query can be designed such that it captures the appearance of a generic steam engine or perhaps that of a generic car. Also, one needs to evaluate the performance of matching under a specified query. In the examples presented here the following method of evaluation is applied. First, the objective of the query is stated and then retrieval instances are gauged against the stated objective. In general, objectives of the form 'extract images similar in appearance to the query' will be posed to the retrieval algorithm.

In this section we start out by demonstrating two retrieval examples and then go on to discuss the performance of the system in terms of recall and precision. Finally the typical computation times for running a query are presented.

A measure of the performance of the retrieval engine can be obtained by examining the recall/precision table for several queries. Briefly, recall is the proportion of the relevant material actually retrieved and precision is the proportion of retrieved material that is relevant. Five queries were submitted to the database to compute the recall/precision shown in Table 1. These queries are enumerated below. For lack of space pictorial results are shown only for the first two.

(1) Using the white wheel as the salient region find all cars with white wheels. This query is depicted in Figure 1. The top twenty five results of this query are shown in Figure 2 read in a text-book manner. Although, as it is clear from the results picture, several valid cars were found within reasonable viewpoint the user is only interested in white wheels and the average precision for this query is 48.6%.

(2) This query is depicted in Figure 3. The user seeks to find similar dark textured apes including monkeys and points to the texture on this ape's coat. The average precision is 57.5% and the top 25 are shown in Figure 4. Note that although the $20^{th}$ image is a monkey (patas monkey), it is not a valid match in as far as the user is concerned because it is not a dark textured ape or monkey. Hence, it is not counted.

(3) The third query is that of the face of a human and the user expects all human

Fig. 3. *Ape query: The user decides that the texture on the coat is important*

faces in the database. The average precision is 74.7%

(4) The fourth query is that of the same human face but this time the user expects to obtain all the pictures of this particular person in the database. The average precision is 61.7%

(5) The fifth query is that of the face of a patas monkey and the user expects to retrieve all patas monkeys whose faces are clearly visible. The average precision is 44.5%.

The recall/precision curve for all these queries together is shown is Table 1. The average precision over all the queries is a 57.4% This compares well with text retrieval where some of the best systems have an average precision of 50%[2].

Table 1. Precision at standard recall points for Five Queries

| Recall | 0 | 10 | 20 | 30 | 40 | 50 | 60 | 70 | 80 | 90 | 100 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Precision % | 100 | 94.1 | 90.6 | 76.4 | 61.8 | 55.3 | 44.1 | 39.5 | 35.8 | 20.0 | 14.1 |

| average | 57.4% |
|---|---|

[2]Based on personal communication with Bruce Croft

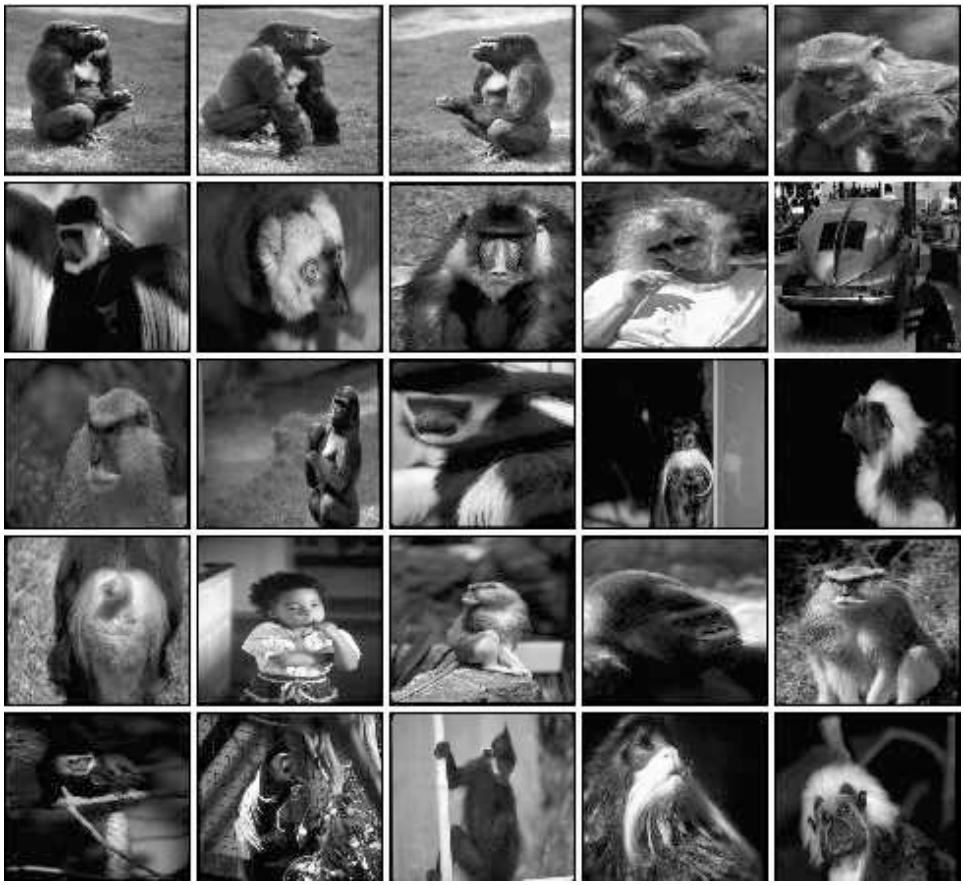

Fig. 4. *'Monkey' Query Results*

Unsatisfactory retrieval occurs for several reasons. First it is possible that the query is poorly designed. In this case the user can design a new query and re-submit. Also Synapse allows users to drop any of the displayed results in to a query box and re-submit. Therefore, the user can not only redesign queries on the original image, but also can use any of the result pictures to refine the search. A second source of error is in matching generalized coordinates by value. The choice of scales in the experiments carried out in this case are $\frac{3}{\sqrt{2}}, 3, \frac{3}{\sqrt{2}}$ with the top two invariant vectors i.e. $\langle d_3, d_4 \rangle$. It is possible that locally the intensity surface may have a very close value, so as to lie within the chosen threshold and thus introduce an incorrect point. By adding more scales or derivatives such errors can be reduced, but at the cost of increased discrimination. Many of these 'false matches' are eliminated in the spatial checking phase. Errors can also occur in the spatial checking phase because it admits much more than a rotational transformation of points with respect to the query configuration. Overall the performance to date has been very satisfactory and we believe that by experimentally evaluating each phase the system can be further improved.

The time it takes to retrieve images is dependent linearly on the number of query points. On a Pentium Pro-200 Mhz Linux machine, typical queries execute in between one and six minutes.

## 6. CONCLUSIONS, LIMITATIONS AND FUTURE WORK

Within small view variations, images that are similar to a query are retrieved. These images are also observed to be visually similar and we posit that this method has good potential for image retrieval.

While a discussion of matching objects across different sizes was presented and has been implemented elsewhere [22], in this paper, the multi-scale invariant vector was used only to robustly characterize appearance. The next immediate step is to explicitly incorporate matching across size variations.

A second important question is, what types of invariants should constitute a feature vector ? This is a question of open research and is subject to extensive verification.

Finally, although the current system is some what slow, it is yet a remarkable improvement over our previous work. We believe that by examining the the spatial checking and sampling components a further increase in speed is possible.

### Acknowledgements

The authors wish to thank Adam Jenkins and Morris Hirsch for programming support and Prof. Bruce Croft and CIIR for continued support of this work.

### REFERENCES

[1] Bimbo, A. D., and Pala, P. Image-indexing using shape-based visual features. In *Proc. IEEE Int. Conf. Patt. Recog.* (1996), vol. 3, pp. 351–355.

[2] Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Lee, D., Petkovix, D., Steele, D., and Yanker, P. Query by image and video content: The qbic system. *IEEE Computer Magazine 28*, 9 (September 1995), 23–30.

[3] Florack, L. M. J. *The Syntactic Structure of Scalar Images.* PhD thesis, University of Utrecht, 1993.

[4] Gorkani, M. M., and Picard, R. W. Texture orientation for sorting photos 'at a glance'. In *Proc. 12th Int. Conf. on Pattern Recognition* (October 1994), pp. A459–A464.

[5] Gudivada, V. N., and Raghavan, V. V. Content-based image retrieval systems. *IEEE Computer Magazine 28*, 9 (September 1995), 18–21.

[6] Hancock, P. J. B., Bradley, R. J., and Smith, L. S. The principal components of natural images. *Network 3* (1992), 61–70.

[7] Jain, A. K., and Vailaya, A. Image retrieval using color and shape. *Pattern Recognition 29* (1996), 1233–1244.

[8] Kirby, M., and Sirovich, L. Application of the kruhnen-loeve procedure for the characterization of human faces. *IEEE Trans. Patt. Anal. and Mach. Intel. 12*, 1 (January 1990), 103–108.

[9] Koenderink, J. J. The structure of images. *Biological Cybernetics 50* (1984), 363–396.

[10] Koenderink, J. J., and van Doorn, A. J. Representation of local geometry in the visual system. *Biological Cybernetics 55* (1987), 367–375.

[11] Lindeberg, T. *Scale-Space Theroy in Computer Vision*. Kluwer Academic Publishers, 1994.

[12] Liu, F., and Picard, R. W. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE Trans. PAMI 18*, 7 (July 1996), 722–733.

[13] Ma, W. Y., and Manjunath, B. S. Texture-based pattern retrieval from image databases. *Multimedia Tools and Applications 2*, 1 (January 1996), 35–51.

[14] Manmatha, R. Measuring affine transformations using gaussian filters. In *Proc. European Conference on Computer Vision* (1994), vol. 2, pp. 159–164.

[15] Manmatha, R., and Oliensis, J. Measuring affine transform - i, scale and rotation. In *Proc. DARPA Image Understanding Workshop* (Washington D.C., 1993), pp. 449–458.

[16] Mehrotra, R., and Gary, J. E. Similar-shape retrieval in shape data management. *IEEE Computer 28*, 9 (September 1995), 57–62.

[17] Mehtre, B. M., Kankanhalli, M. S., Narasimhalu, A. D., and Man, G. C. Color matching for image retrieval. *Pattern Recognition Letters 16*, 3 (March 1995), 325–331.

[18] Mokhtarian, F., Abbasi, S., and Kittler, J. Efficient and robust retrieval by shape content through curvature scale-space. In *First International Workshop on Image Databases and Multi-media Search* (August 1996).

[19] Nayar, S. K., Murase, H., and Nene, S. A. Parametric appearance representation. In *Early Visual Learning*. Oxford University Press, February 1996.

[20] Pentland, A., Picard, R. W., and Sclaroff, S. Photobook: Tools for content-based manipulation of databases. In *Proc. Storafe and Retrieval of Image and Video Databases II* (1994), vol. 2, SPIE, pp. 34–47.

[21] Rao, R., and Ballard, D. Object indexing using an iconic sparse distributed memory. In *Proc. International Conference on Computer Vision* (1995), IEEE, pp. 24–31.

[22] Ravela, S., Manmatha, R., and Riseman, E. M. Image retrieval using scale-space matching. In *Computer Vision - ECCV '96* (Cambridge, U.K., April 1996), B. Buxton and R. Cipolla, Eds., vol. 1 of *Lecture Notes in Computer Science*, 4th European Conf. Computer Vision, Springer.

[23] Schmid, C., and Mohr, R. Combining greyvalue invariants with local constraints for object recognition. In *Proc. Computer Vision and Pattern Recognition* (June 1996), IEEE.

[24] Sclaroff, S. Encoding deformable shape categories for efficient content-based search. In *Proc. First International Workshop on Image Databases and Multi-Media Search* (August 1996).

[25] Strickler, M., and Orengo, M. Similarity of color images. In *Storage and Retrieval for Image and Video Databases III* (1995), vol. 2420 of *SPIE Proceedings Series*, pp. 318–192.

[26] Swain, M., and Ballard, D. Color indexing. *Int. Jrnl. Comput. Vision 7*, 1 (1991), 11–32.

[27] Swets, D. L., and Weng, J. Using discriminant eigen features for retrieval. *IEEE Trans. Patt. Anal. and Mach. Intel. 18* (August 1996), 831–836.

[28] ter Har Romeny, B. M. *Geometry Driven Diffusion in Computer Vision*. Kluwer Academic Publishers, 1994.

[29] Turk, M., and Pentland, A. Eigen faces for recognition. *Jrnl. Cognitive Neuroscience 3* (1991), 71–86.

[30] Vailaya, A., Zhong, Y., and Jain, A. K. A hierarchical system for efficient image retrieval. In *Proc. Int. Conf. on Patt. Recog.* (August 1996).

[31] Witkin, A. P. Scale-space filtering. In *Proc. Intl. Joint Conf. Art. Intell.* (1983), pp. 1019–1023.