M.I.T. Media Laboratory Perceptual Computing Technical Report No. 255, Nov. 1993

Appeared: SPIE Storage and Retrieval Image and Video Databases II, No. 2185, Feb 6-10, 1994, San Jose

To Appear: International Journal of Computer Vision, Fall 1995

Photobook: Content-Based Manipulation of Image Databases

A. Pentland, R. W. Picard, S. Sclaroff
Perceptual Computing Section, The Media Laboratory
Massachusetts Institute of Technology
{sandy,picard,stan}@media.mit.edu

June 12, 1995

Abstract

We describe the Photobook system, which is a set of interactive tools for browsing and searching images and image sequences. These query tools differ from those used in standard image databases in that they make direct use of the image content rather than relying on text annotations. Direct search on image content is made possible by use of *semantics-preserving image compression*, which reduces images to a small set of perceptually-significant coefficients. We discuss three types of Photobook descriptions in detail: one that allows search based on appearance, one that uses 2-D shape, and a third that allows search based on textural properties. These image content descriptions can be combined with each other and with text-based descriptions to provide a sophisticated browsing and search capability. In this paper we demonstrate Photobook on databases containing images of people, video keyframes, hand tools, fish, texture swatches, and 3-D medical data.

1 Introduction: The Problem

Digital imagery, whether single frames or extended sequences, is becoming an important component of computer and telecommunication usage. However the increasing use of imagery is causing severe problems, because the technology for organizing and searching images based on their content is still in its infancy. This is especially clear in the development of multimedia applications, where the difficulty of searching and editing image data is often the single largest cost factor.

Currently the standard approach to searching image and video is to create text annotations that describe the content of the image, and then enter these textual annotations into a standard database. The images themselves are not really part of the database; they are only referenced by text strings or pointers.

The problem with this approach is that the old saying "a picture is worth 1000 words" is an understatement. In most images there are literally hundreds of objects that could be referenced, and each imaged object has a long list of attributes. Even worse, spatial relationships are important in understanding image content, so that complete annotation of an image with n objects each with m attributes requires $O(n^2m^2)$ database entries. And if we must also consider relations among images, then the problem quickly becomes intractable.

In today's image database systems these annotations must be entered by hand with great tedium and prohibitive cost. The result is that users enter only the minimum number of annotations required to accomplish their current task. Consequently, the resulting labelings are not rich enough or consistent enough for different sorts of queries, so that image databases are typically re-annotated for each problem.

1.1 Semantic Indexing of Image Content

The problem is that to make a user- and purpose-independent image database we must annotate everything in the images and all the relations between them. Text databases avoid this problem by using strings of characters (*e.g.*, words) that are a consistent encoding of the database's semantic content. Thus questions about the database's semantic content can be answered by simply comparing sets of text strings. Because this search is efficient, users can search for their answers at query time rather than having to pre-annotate everything.

To accomplish the same thing for image databases, we must be able to efficiently compare the images themselves, to see if they have the same (or more generally, similar) semantic content. There is, of course, a tradeoff between how much work you do at input time and how much you do at query time. For instance, one could try to precompute the answers to all possible queries, so that no search would be required. Alternatively, one could search the raw images themselves, repeating all of the low-level image processing tasks for each query.

For image databases there is a compelling argument for employing a pre-purposive "iconic" level of representation. It does not make sense to try to precompute a "general purpose," completely symbolic representation of image content, because the number of possibly-interesting geometric relations is combinatorially explosive. Consequently, the output of our precomputation must be image-like data structures where the geometric relationships remain implicit. On the other hand, it *does* make sense to precompute as much as is possible, because low-level image operations are so expensive.

These precomputed image primitives must play a role similar to that of letters and words in a database query sentence. The user can use them to describe "interesting" or "significant" visual events, and then let the computer search for instances of similar events. For instance, the user should be able to select a video clip of a lush waterfall, and be able to ask for other video sequences in which more of the same "stuff" occurs. The computer would then examine the pre-computed decomposition of the waterfall sequence, and characterize it in terms of texture-like primitives such as spatial and temporal energy. It could then search the pre-computed decomposition of other video clips to find places where there is a similar distribution of primitives.

Alternatively, the user might circle a "thing" like a person's face, and ask the computer to track that person within the video clip, or ask the computer to find other images where the same person appears. In this case the computer would characterize the person's 2-D image appearance in terms of primitives such as edge geometry and the distribution of normalized intensity, and then either track this configuration of features over time or search other images for similarly-arranged conjunctions of the same features.

These two types of semantic indexing — using texture-like descriptions of "stuff" and using object-like descriptions of "things" — constitute the two basic types of image search operation in our system. These two types of description seem to be fundamentally different in human vision [1], and correspond roughly to the distinction between mass nouns and count nouns in language. Note that both types of image query can operate on the same image primitives (*e.g.*, the energy in different band-pass filters) but they differ in how they group these primitives for comparison. The "stuff" comparison method pools the primitives without regard to detailed local geometry, while the "things" method preserves local geometry.

2 Semantics-Preserving Image Compression

The ability to search at query-time for instances of the same (or similar) image events depends on two conditions:

- There must be a similarity metric for comparing objects or image properties (*e.g.,* shape, texture, color, object relationships, *etc.*) that matches human judgments of similarity. This is *not* to say that the computation must somehow mimic the human visual system; but rather that computer and human judgments of similarity must be generally correlated. Without this, the images that the computer finds will not be those desired by the human user.

- The search must be efficient enough to be interactive. A search that requires minutes per image is simply not useful in a database with millions of images. Furthermore, interactive search speed makes it possible for users to recursively refine a search by selecting examples from the currently retrieved images and using these to initiate a new select-sort-display cycle. Thus users can iterate a search to quickly "zero in on" what they are looking for.

Consequently, we believe that the key to solving the image database problem is *semantics-preserving image compression*: compact representations that preserve essential image similarities. This concept is related to some of the "semantic bandwidth compression" ideas put forth in the context of image compression [30] [31] [46] [40]. Image coding has utilized semantics primarily through efforts to compute a compact image representation by exploiting knowledge about the *content* of the image. A simple example of semantic bandwidth compression is coding the people in a scene using a model specialized for people, and then using a different model to code the background.

In the image database application, compression is no longer the singular goal. Instead, it is important that the coding representation 1) be "perceptually complete" and 2) be "semantically meaningful." The first criterion will typically require a measure of perceptual similarity. Measures of similarity on the coefficients of the coded representation should correlate with human judgments of similarity on the original images.

The definition of "semantically meaningful" is that the representation gives the user direct access to the parts of the image content that are important for their application. That is, it should be easy to map the coefficients that represent the image to "control knobs" that the user finds important. For instance, if the user wishes to search among faces, it should be easy to provide control knobs that allow selection of facial expressions or selection of features such as moustaches or glasses. If the user wishes to search among textures, then it should be easy to select features such as periodicity, orientation, or roughness.

Having a semantics-preserving image compression method allows you to quickly search through a large number of images because the representations are compact. It also allows you to find those images that have perceptually similar content by simply comparing the coefficients of the compressed image code. Thus in our view the image database problem requires development of semantics-preserving image compression methods.

2.1 Comparison with Other Approaches

In recent years there has been a growing interest in the image database problem [2, 25]. The first proposed solutions were intended for engineering drawings, and typically assumed that hand preprocessing had fully "predigested" them into meaningful parts and functional features [8, 9, 28, 29]. We feel that this requirement is acceptable for things like CAD drawings, but not for general imagery.

More recently, researchers have proposed a variety of *image indexing* methods, based on shape [10, 23, 24, 26, 27, 33], color [4, 50, 22], or combinations of such indices [35, 13]. The general approach is to calculate some approximately invariant statistic, like a color histogram or invariants of shape moments, and use that to stratify or partition the image database. Such partitioning allows users to limit the search space when looking for a particular image, and has proven to be quite useful for small image databases [35, 13].

The difference between these methods and ours is that they emphasize computing a discriminant that can reject many false matches, whereas ours can encode the image data to the accuracy required to retain "all" of its perceptually salient aspects. Generally speaking, the coefficients these earlier efforts have produced are not sufficiently meaningful to reconstruct the perceptually salient features of the image. For instance, one cannot reconstruct an image region from its moment invariants or its color histogram. In contrast, the models we present use coefficients which allow reconstruction. Figure 1 shows three reconstructions using appearance, shape, and texture descriptions of image content.

In our view the problem with using invariants or discriminants is that significant semantic information is irretrievably lost. For instance, do we really want our database to think that apples, Ferrarris, and tongues are "the same" just because they have the same color histogram? Discriminants give a way to limit search space, but do not answer "looks like" questions except within constrained data sets. In contrast, when the coefficients provide a perceptually complete representation of the image information, then things the database thinks are "the same" actually *look the same*.

Another important consequence of representational completeness is that we can ask a wide range of questions about the image, rather than being limited to only a few predefined questions. For in-

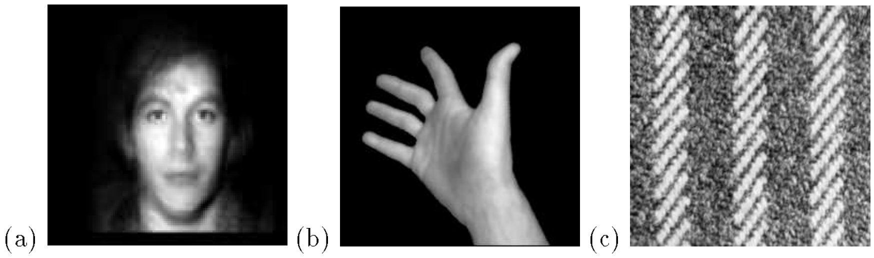


Fig. 1. Images reconstructed from coefficients used for database search: (a) 30 appearance coefficients, (b) 100 shape coefficients, (c) 60 texture coefficients

stance, it requires only a few matrix multiplies per image to calculate indices such as color histograms or moment invariants from our coefficients. The point is that if you start with a relatively complete representation, then you are not limited in the types of questions you can ask; whereas if you start by calculating discriminants, then you are limited to queries about those particular measures *only*.

2.2 Semantics-preserving image compression

How can we design "semantics-preserving image compression" algorithms? Our general idea is to first transform portions of the image into a canonical coordinate system that preserves perceptual similarities, and then to use a lossy compression method to extract and code the most important parts of that representation. By careful choice of transform and coding methods this approach can produce an optimally-compact, semantics-preserving code suitable for image database operations.

Note that because different parts of the image have different characteristics, we must use a variety of representations, each tuned for a specific type of image content. This is the same requirement as for semantic bandwidth compression. In the examples below we will describe representations for faces, textures, hand tools, fish, video keyframes and human brain ventricles.

The necessity for multiple content-specific representations means that we must also have an efficient, automatic method for developing "basis functions" specific to object or texture classes. For representing object classes, which require preservation of detailed geometric relations, we use an approach derived from the Karhunen-Loève transform. The Karhunen-Loève transform is known to provide an optimally-compact linear basis (with respect to RMS error) for a given class of signal. For characterization of texture classes, we use an approach based on the Wold decomposition. This transform separates "structured" and "random" texture components, allowing extremely efficient encoding of textured regions while preserving their perceptual qualities.

2.3 Finding instances of models

To employ the strategy of semantics-preserving image compression for image database search, we must be able to determine which image data belongs to each of our different content-classes as we are preprocessing the data for entry into the database. While this remains a difficult problem in general, and must often be solved using heuristic methods, we have developed two useful solutions that appear to be fairly general-purpose.

The first solution is to use motion and color to pull out foreground objects. We have found that this sort of figure-ground segmentation can be done reliably and efficiently by use of clustering in conjunction with optical flow [11, 55] and/or color difference information [12]. This provides us with good "cut-outs" of foreground objects, as is illustrated in Figure 2. We can then analyze the shape, appearance, motion, and texture of these foreground objects, inserting their descriptions into our database. Similarly, we can analyze the appearance, motion, and texture of the background, and insert this information into our database.

The computation of foreground/background motion can also be used to provide a qualitative characterization of camera and object motion within a video clip, e.g, pan left, zoom in, or move stage right. This allows us to select *keyframes* from video clips. Keyframes are images that are "characteristic" or "typical" of the video clip's content. For instance, good keyframes typically occur at the beginning and end of clips, in the middle of no-motion segments, or in the middle of segments where the camera is tracking a foreground object. That is, good keyframes can be found at zero-crossings and extrema of camera and object motion.

By using camera and foreground/background motion to automatically select keyframes, we can reduce the problem of searching video data to the much less costly processing of a few individual images. Editors and artists have long known that the semantic content of video can be accurately summarized by a series of appropriately-selected keyframes that have been assembled into a *storyboard*. Keyframe extraction, therefore, is an important example of semantics-preserving video compression.

Our second method for finding instances of models is to recast the problem as one of *detection* rather than segmentation. The basic idea is to represent specific classes of interest by using prototype(s) and a small set of parametric variations or deformations. Such a representation can be made to be narrowly "tuned" for its target; it can very efficiently describe the signals it was trained for, but will be very bad at describing other signals. Thus if a particular content-specific representation accurately describes some portion of an image, then it is very likely to be an *appropriate* representation of that image data.

This allows us to detect instances of models by asking how well they can describe each part of the image. Although not a real-time process on current workstations, this computation is sufficiently efficient to be incorporated in the image preprocessing step. We first used this approach for finding faces [53], and have now applied it to finding a wide variety of "things" (including eyes, cars, roads, *etc.* [34]). Multiple texture models can also be used to find "stuff" such as sky, trees, buildings, *etc.* [44].

Finally, it should be remarked that this framework for searching images is based on 2-D matching of appearance, rather than matching of 3-D properties. There are two reasons for adopting this approach. The first is that a 2-D matching approach can be trained directly from image data; it does not require a 3-D model. The second reason is that the 2-D approach has *lower* computational complexity than 3-D methods. Breuel [6], for instance, has proven that only $O(\delta^{-2})$ 2-D aspects are needed to cover the entire 3-D viewing sphere with a 2-D matching error bounded by δ radians ($0 < \delta < 1$). For instance, a 2-D, template-based object recognition algorithm may require only thirty templates to cover all possible viewing directions. This lower computational complexity is an important consideration for image database applications.

3 Photobook

Photobook is a computer system that allows the user to browse large image databases quickly and efficiently, using both text annotation information in an AI database and by having the computer search the images directly based on their content [38, 16, 42]. This allows people to search in a flexible and intuitive manner, using semantic categories and analogies, *e.g.*, "show me images with text annotations similar to those of this image but shot in Boston," or visual similarities, *e.g.*, "show me images that have the same general appearance as this one."

Interactive image browsing is accomplished using a Motif interface. This interface allows the user to first select the category of images they wish to examine; *e.g.,* pictures of white males over 40 years of age, or images of mechanic's tools, or cloth samples for curtains. This subset selection is accomplished by searching text annotations using an object-oriented, memory-based AI database called Framer [18, 19]. Photobook then presents the user with the first screenful of these images (see Figure 3); the rest of the images can be viewed by "paging" through them one screen at a time.

Users most frequently employ Photobook by selecting one (or several) of the currently-displayed images, and asking Photobook to sort the entire set of images in terms of their similarity to the selected image (or set of images). [1] Photobook then re-presents the images to the user, now sorted by similarity to the selected images. The select-sort-redisplay cycle typically takes less than one second. When searching for a particular item, users quickly scan the newly-displayed images, and initiate a new select-sort-redisplay cycle every two or three seconds.

Photobook can have many different types of im-

[1] By selecting several example images the user is providing information about the distribution of visual parameters that constitute the class of interest. Photobook uses multiple examples to make an improved estimate of the parameter's probability distribution function (PDF). We have experimented with allowing the user to provide both positive and negative examples, and with characterization of arbitrary PDFs [34, 44], although the current interface only supports updating the parameter's mean from multiple positive examples.

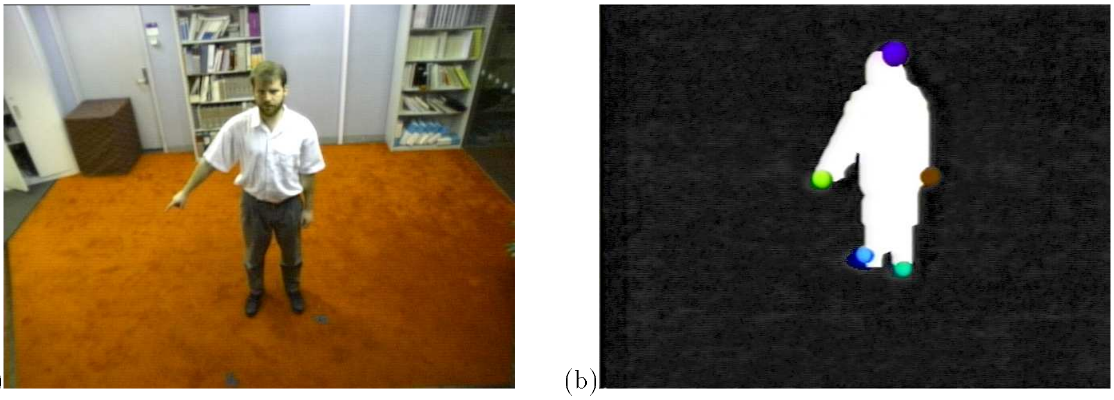
(a) (b)


Fig. 2. Using motion and color information, we can separate foreground objects from background. This figure shows a system that extracts the outlines of people in view; a geometric analysis of the outline is then used to label position of head, hands, and feet. This system runs at 20 frames/second without special hardware, and has been tested on more than 2,000 people [12].

age descriptions available to it. In this paper we will discuss appearance-specific descriptions ("Appearance Photobook") applied to face and keyframe databases, texture descriptions ("Texture Photobook") applied to texture-swatch and keyframe databases, and shape descriptions ("Shape Photobook") applied to hand-tool and fish databases. Each of these descriptions can be made rotation and scale invariant, although for many applications this is not desirable.

Photobook can also handle combinations of these descriptors, *e.g.*, shape and appearance, which we will illustrate using 3-D data of human brain ventricles. It can also handle complex functions of text annotations, via functionality of the Framer knowledge representation language [18, 19].

Obvious applications for "Appearance Photobook" as applied to face databases include customs, security, and criminal investigation. A different application would be a dating service where individuals could browse a database of prospective partners based on their looks as well as biographical data.

Applications of "Shape Photobook" include searching catalogs of consumer goods such as hand tools. Another economically important application is searching inventories of mechanical parts, or botanical and biological catalogs.

Similarly, a natural application of "Texture Photobook" as applied to texture patches is in the design and decorating industries, where the buyer/designer can browse a large database of fabrics, tiles, wallcoverings, and other textiles, while incorporating factors such as material composition and manufacturing costs in the search.

4 Appearance Photobook

To efficiently measure similarity in appearance within an object class we must first determine which features are most effective at describing the images of those objects. The standard linear method for extracting such information about a set of images is known as the Karhunen-Loève transform (KLT). This transform uses the eigenvectors of the covariance matrix of the set of image features, *i.e.*, it uses the principal components of the distribution of image features. These eigenvectors can be thought of as a set of *parametric variations* from the mean or *prototypical* appearance. These eigenvectors together characterize all of the variations between images of the object and the object's prototypical appearance. Normally only a few eigenvectors with the largest eigenvalues are employed, as these will account for the vast majority of the variance between object images.

In this paper we will illustrate this technique using databases of face images and video keyframes. We will also illustrate how the technique can be combined with shape descriptions to search and sort 3-D medical data.

4.1 Eigenimage representations

The general approach taken to produce an appearance description is as follows. Input images are first preprocessed to normalize them for position, scale, orientation and similar nonlinear effects. Eigenvectors of the normalized image covariance are

(a)

(b)

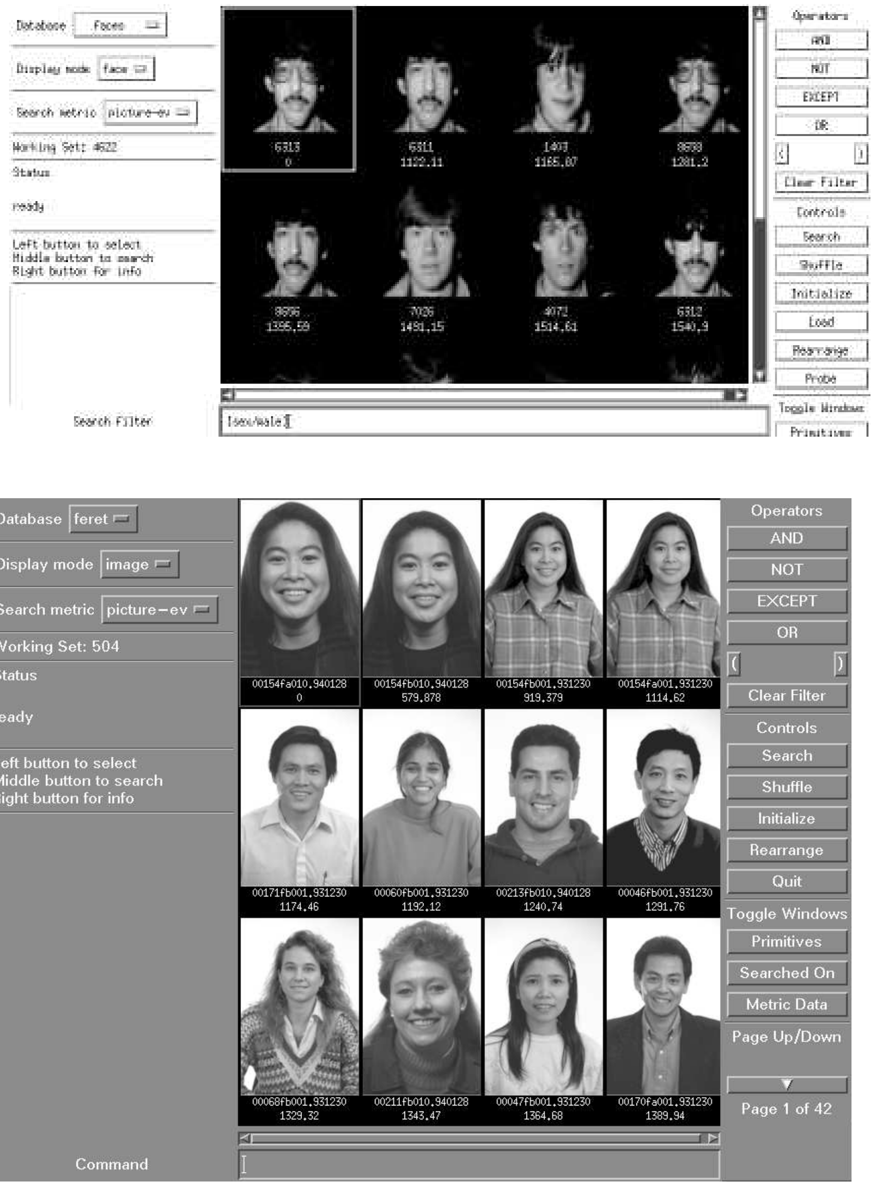


Fig. 3. In both (a) and (b) the face at the upper left was selected randomly; the remainder of the faces are the most-similar faces from among the entire database. Similarity decreases left to right, top to bottom. Note the ability to match people despite wide variations in expression, size, hairstyle, etc.

then calculated for a set of training images and subregions of the training images, resulting in eigenimage representations both for the whole object and its subfeatures (*e.g.*, the whole face as well as eyes, nose, and mouth).

Note that the input data may be grey-level or color images (as in the following examples), or they may be images of extracted edges or extracted texture measurements. Voxel and 1-D data have also been used. Regardless of the type and dimensionality of the input data, Appearance Photobook represents the input data in terms of its principal variations from the mean or prototypical appearance of the input class.

In the case where we do not know the class of the imaged object (e.g., is it a forward view of a face, a side view of face, or a car?), we can automatically determine which appearance model is most appropriate for a new image by measuring how well each model describes the image data. This is accomplished by determining which set of eigenimages provides the best encoding of the image; the same approach is also used to detect occurrences of these models in the image. The details of this procedure are described in references [39, 34] and discussed in Section 7.2.

Note that because this approach is view-based, we must have separate models if we want to describe appearance from different points of view. For instance, to represent facial appearance as a function of out-of-plane rotation, we separately train eigenimage representations at rotations of ± 90, ± 45 and 0 degrees.

4.1.1 Building Eigenrepresentations

Let an image region $I(x,y)$ be a two-dimensional N by N array of intensity values, or a vector of dimension N^2. An ensemble of such regions, then, maps to a collection of points in a space of size N^2. Images of compact objects and features (*e.g.*, faces, cars, eyes) for a given viewing geometry will not be randomly distributed in this huge image space and thus can be described by a relatively low-dimensional subspace. This subspace can be approximated by use of the Karhunen-Loève expansion, *e.g.*, the eigenvectors of the autocorrelation matrix. For face imagery we refer to this subspace as "face space" and the eigenvectors as "eigenfaces" or "eigenfeatures" [53, 39].

Let the training set of images be $\Gamma_1, \Gamma_2, \Gamma_3, \ldots \Gamma_M$. The average of the set is defined by $\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n$. Each training image differs from the average by the vector $\Phi_i = \Gamma_i - \Psi$. This set of large vectors is then subject to the Karhunen-Loève expansion, to produce the unique set of M orthonormal vectors $\mathbf{u}_n$ and their associated eigenvalues λ_k that optimally describe the distribution of the data in an RMS error sense. The vectors $\mathbf{u}_k$ and scalars λ_k are the eigenvectors and eigenvalues, respectively, of the covariance matrix

$$\begin{aligned} C &= \frac{1}{M}\sum_{n=1}^{M}\Phi_n\Phi_n^T \\ &= \frac{1}{M}AA^T \end{aligned} \tag{1}$$

where the matrix $A = [\ \Phi_1\ \Phi_2\ \ldots\ \Phi_M\ ]$. The mean and first few eigenvectors for human faces are shown in Figure 4; linear combinations of these eigenimages span the space of human face images at coarse resolution and with fixed position, orientation, and scale. Note that the first three eigenvectors primarily describe variations due to illumination and surface albedo.

Note that the matrix C is N^2 by N^2, so directly determining the N^2 eigenvectors and eigenvalues is difficult for typical image sizes. We need a computationally feasible method to find these eigenvectors. Fortunately we can determine the eigenvectors by first solving a much smaller M by M matrix problem, and taking linear combinations of the resulting vectors [46, 53].

Code for this calculation, together with technical reports providing additional detail, is available by anonymous FTP from whitechapel.media.mit.edu.

A new image region (Γ) is transformed into its eigenimage representation (*e.g.*, projected into "face space") by a simple operation, $\omega_k = \mathbf{u}_k^T(\Gamma - \Psi)$ for $k = 1, \ldots, M' < M$. The vector $\Omega^T = [\omega_1\, \omega_2\, \ldots\, \omega_{M'}]$ describes the input image in terms of the orthogonal eigenfeature basis set; thus, the vector Ω^T is an encoding of the image in terms of the eigenimage basis. An example encoding of a face is shown in Figure 1(a). The similarity between two images i and j is computed by comparing their within-eigenimage-subspace distance $\epsilon_{ij}^2 = \|(\Omega_i - \Omega_j)\|^2$.

4.2 Database experiments

Most image database applications require comparison with a large number of possible images. This is particularly true for face images; for instance, dating services, casting agencies, and police

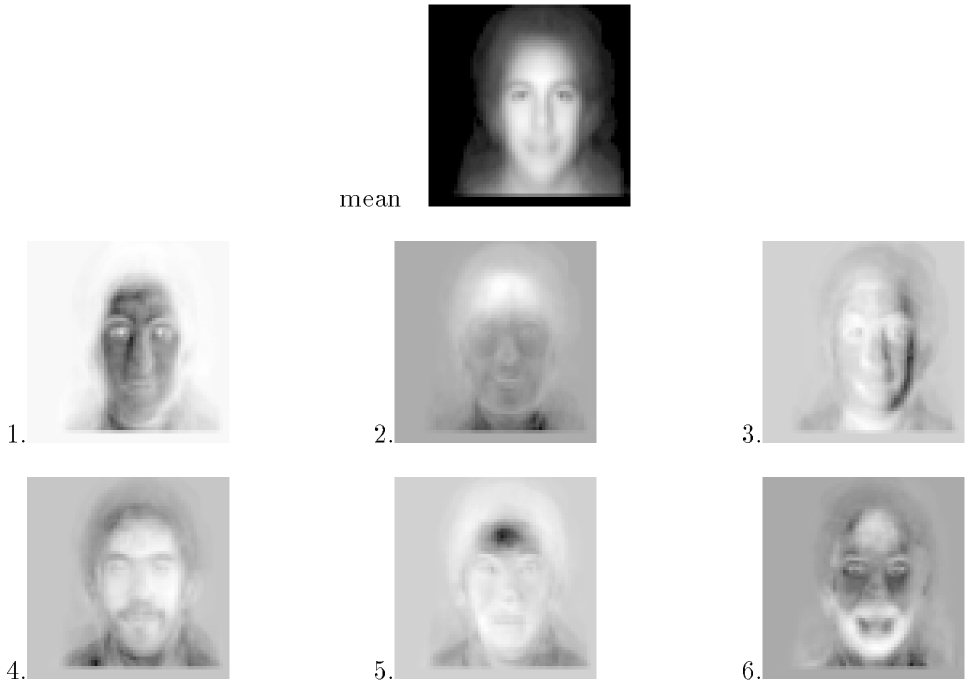


Fig. 4. The mean and first few eigenvectors computed from a large database of faces of men, women, and children of all races. Note that illumination effects appear primarily in the subspace spanned by eigenvectors one through three.

stations all commonly have collections of more than 1,000 images.

Our first test of Appearance Photobook, therefore, was on the Media Laboratory database of 7,562 images of approximately 3,000 people. The images were collected in a small booth at a Boston photography show, and include men, women, and children ranging between (approximately) 4 to 75 years of age. A wide range of ethnic and racial types were included in a proportion similar to that of the general Boston area population. Head position was controlled by asking people to take their own picture when they were lined up with the camera. Two LEDs placed at the bottom of holes adjacent to the camera allowed them to judge their alignment; when they could see both LEDs then they were correctly aligned. Each image was then annotated (by hand) as to sex, race, approximate age, facial expression, and other salient features. Whether or not two images were of the same person was also annotated by hand. Almost every person has at least two images in the database; several people have many images with varying expression, headwear, facial hair, *etc.*

Figure 3(a) shows a typical result of a similarity search on this database. The face at the upper left was selected by the user; the remainder of the faces are the next most-similar faces from among the entire 7,562 Media Laboratory database. Similarity decreases left to right, top to bottom. As can be seen, the image most similar to the selected image is another image of the same person. Note that at the lower right is still another image of this same person...but wearing sunglasses. Photobook's performance on this database was evaluated on a random sample of 200 images, and recognition accuracy was found to be 95%, while verification accuracy was above 99% [39].

Figure 3(b) illustrates Photobook's performance on a second face database, assembled by the Army Research Laboratory at Ft. Belvoir, which contains substantial variations in scale, position, and head orientation. The face at the upper left was selected by the user; the remainder of the faces are the most-similar faces from the 575 frontal views in this database. Note that the first four images (in the top row) are all of the same person. On this database Photobook achieved a recognition accuracy of 99.4%, and a verification accuracy of 100%. Section 7.2 describes in more detail how the problems of scale, position, and orientation were addressed.

In both cases the entire searching and sorting operation takes less than one second on a standard Sun Sparcstation, because each face is described using only a very small number of eigenvector coefficients. Of particular interest is Appearance Photobook's ability to find the same person despite wide variations in expression, hairstyle, image size, and eyewear.

5 Shape Photobook

To compare the shape similarities between two objects, we must be able to describe the deformations (differences) that relate them. Sometimes differences between objects of the same type are due to changes in viewing geometry, *e.g.*, foreshortening or distance change. Other times they are due to physical deformation: one object is a [stretched, bent, tapered, dented, ...] version of the other. For instance, most biological objects are flexible and articulated.

To describe these deformations, therefore, it is reasonable to qualitatively model the physics by which real objects deform, and then to use that information to guide the matching process. So rather than using image correlations as the basis for a semantics-preserving code, we model the physical "interconnectededness" of the shape. In other words, we build a shape model made of a virtual material that fills the space between nearby features, *e.g.,* edges, corners, or high-curvature points. In engineering, this interconnectededness is standardly computed by use of the finite element method (FEM). This method produces a positive definite symmetric matrix, called the *stiffness matrix*, which describes how each point on the object is connected to every other point. This stiffness matrix plays the same role in Shape Photobook that the covariance matrix did in Appearance Photobook.

Consequently, we derive our semantics-preserving code for shape in a manner similar to that used for appearance: we calculate the eigenvectors of the stiffness matrix, and use these to encode deformations relative to some base or average shape. Once the eigenvector shape description has been computed, we can compare shapes simply by looking at the amplitudes of the eigenvectors, as was done in the Appearance Photobook example described above. Perhaps the major difference in how the shape and appearance codes are used in Photobook is the preprocessing to align the shapes. This preprocessing is developed in detail in references [47, 48] and discussed in Section 7.2.

5.1 Eigenmode Representations

In Shape Photobook an object's shape representation is based on the eigenvectors of its physical model. In physical systems these eigenvectors are called the *modes* of the system; they describe the intrinsic symmetries of the object in a unique and canonical manner.

Before obtaining these eigenvectors, we first build a physical model for the shape using the finite element method. Interpolation functions are developed that allow continuous material properties, such as mass and stiffness, to be integrated across the region of interest. In [47] we introduced a new finite element formulation that uses Gaussian basis functions as FEM interpolants; this allows us to use the data itself to define the deformable object, by building stiffness and mass matrices that use the positions of image feature points as the finite element nodes. For an in-depth description of this formulation, readers are directed to [37, 47, 48].

To compare two FEM shape representations, we deform one elastic shape model to align it with the other. This requires solving the *dynamic equilibrium equation*:

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{R}, \tag{2}$$

where $\mathbf{U}$ is a vector describing the object's deformation, $\mathbf{R}$ is a load vector whose entries are the spring forces pulling the first shape into alignment with the second, and where $\mathbf{M}$ and $\mathbf{K}$ are the element mass and stiffness matrices, respectively.

This system of equations can be decoupled by posing the equations in a basis defined by the $\mathbf{M}$-orthogonalized eigenvectors of $\mathbf{K}$. These eigenvectors and values are the solution (ϕ_i, ω_i^2) to the following generalized eigenvalue problem:

$$\mathbf{K}\phi_i = \omega_i^2 \mathbf{M}\phi_i. \tag{3}$$

The vector ϕ_i is called the i^{th} *eigenmode shape vector* and ω_i is the corresponding frequency of vibration. The i^{th} shape vector describes how each node is displaced by the i^{th} eigenmode.

The shape vectors ϕ_i are $\mathbf{M}$-orthonormal, this means that

$$\mathbf{\Phi}^T \mathbf{K} \mathbf{\Phi} = \mathbf{\Omega}^2 \quad \text{and} \quad \mathbf{\Phi}^T \mathbf{M} \mathbf{\Phi} = \mathbf{I}. \tag{4}$$

where the ϕ_i are columns in the transform $\mathbf{\Phi}$, and ω_i^2 are the elements of the diagonal matrix $\mathbf{\Omega}^2$.

The generalized coordinate transform $\mathbf{\Phi}$ is then used to transform between nodal point displacements $\mathbf{U}$ and decoupled eigenmode displacements $\tilde{\mathbf{U}}$, where $\mathbf{U} = \mathbf{\Phi}\tilde{\mathbf{U}}$. We can now rewrite Equation 2 in terms of these generalized or eigenmode displacements, obtaining a decoupled system of equations:

$$\ddot{\tilde{\mathbf{U}}} + \mathbf{\Omega}^2\tilde{\mathbf{U}} = \mathbf{\Phi}^T\mathbf{R}, \tag{5}$$

allowing for closed-form solution to equilibrium problems such as shape fitting [37].

Code for these operations (for the case of simple 3-D objects only), together with technical reports providing additional detail, is available by anonymous FTP from whitechapel.media.mit.edu.

Whenever a new object is entered into Shape Photobook, the first step is to compute its $\mathbf{M}$, $\mathbf{K}$, and $\mathbf{\Phi}$ matrices. To obtain an eigenmode description of an object relative to some base or average object, we must determine correspondence between the features of the two objects. Normally this is done once when a new object is entered into Shape Photobook, and the correspondences stored. This process is called *modal matching*, and is discussed in references [47, 48]. Given these correspondences, we can then recover the eigenmode deformations $\tilde{\mathbf{U}}$ that deform the matched points on one object to their corresponding positions on a prototype object.

This is done by noting that the nodal displacements $\mathbf{U}$ that align corresponding features on both shapes can be written:

$$\mathbf{u}_i = \mathbf{x}_{1,i} - \mathbf{x}_{2,i}, \tag{6}$$

where $\mathbf{x}_{1,i}$ is the i^{th} node on the first shape and $\mathbf{x}_{2,i}$ is its matching node on the second shape. These nodal displacements can then be transformed into eigenmode amplitudes by the relation $\tilde{\mathbf{U}} = \mathbf{\Phi}^T\mathbf{U}$.

Such a set of eigenmode amplitudes can be used directly for object recognition and comparison [37], exactly as the eigenimage amplitudes were used in Appearance Photobook. As in that case we need use only a few coefficients to obtain an accurate encoding of the shape; discarding high-frequency eigenmodes also tends to make our comparisons robust to noise and local shape variations.

Alternatively — since the underlying model is a physical one — we can compute and compare the amount of energy needed to align an object, and use this as a similarity measure instead. If the strain energy required to align two feature sets is relatively small, then the objects are very similar. The strain energy associated with the i^{th} eigenmode is simply:

$$E_{mode_i} = \frac{1}{2}\tilde{u}_i^2{\omega_i}^2. \tag{7}$$

Since each eigenmode's strain energy is scaled by its frequency of vibration, there is an inherent penalty for deformations that occur in the higher-frequency eigenmodes. In our experiments, we have used strain energy for most of our object comparisons, since it has a convenient physical meaning. Strain energy also has the advantage that it places greater weight on the low-frequency eigenmodes, reducing the influence of the noise-susceptible higher-frequency eigenmodes.

Finally, we note that the first three eigenmodes are always translation and rotation, and the next few modes whole-body shear, compression, *etc.* Thus if it is desirable to make object comparisons rotation, position, and/or scale independent, we can accomplish this by ignoring displacements in the low-order or rigid body eigenmodes.

5.2 Database experiments

The first experiment is with a database of 60 images of 12 objects and non-rigid deformations of those objects, and includes variations in perspective, scale, and lighting. Silhouettes were first extracted and thinned from each tool image, and then the strongest corresponding contour points were found. Eigenmode amplitudes for the first 22 modes were recovered and used to compare each tool to all the other tools using the strain energy similarity measure.

Figure 5 illustrates two typical searches using Shape Photobook on this database; the user selected the image at the upper left, and Photobook returned the other images sorted by similarity from left to right, top to bottom. The similarity statistic appears below each match. Search accuracy over this database is 100%, that is, if there were n hammers in the database and the user searched using a hammer shape as the query image, then the n most-similar objects found were all hammers. Note that the matching is orientation and scale invariant modulo limits imposed by pixel resolution.

The fact that the similarity measure produced by the system corresponds to functionally-similar shapes is important. It allows us to recognize the most similar wrench or hammer from among a group of tools, even if there is no tool that is an exact match. Moreover, if for some reason the most-similar tool can not be used, we can then find the next-most-similar tool, and the next, and so on. We can find (in order of similarity) all the tools that are likely to be from the same *category*.

The second example shows two similarity searches using a database of 74 tropical fish images. Again, the user selected the image at the upper left, and Photobook returned the most similar images sorted left to right, top to bottom. Euclidean distance in strain-space was again used as the similarity metric. Matching is orientation and scale invariant modulo limits imposed by pixel resolution.

In Figure 6(a), a search was initiated to find fish shapes similar to the banded butterflyfish that appears at the upper left. As can be seen, the system correctly retrieved the fish shapes that were closest to the banded butterflyfish shape (*e.g.*, all the other butterflyfish). In Figure 6(b), a search was initiated to find fish shapes similar to the trumpetfish that appears at the upper left. Again, the system correctly retrieved the fish shapes that were closest to the trumpetfish.

As with the hand-tool database, we again see that the system's measure of shape similarity allows us to find objectively-similar objects. It allows us to recognize that two objects are similar even if there is no exact match. This has in turn allowed us to find all the fish that are likely to be from the same taxonomic category.

6 Texture Photobook

The Appearance Photobook and Shape Photobook employ similarity metrics that are related to RMS differences, either in the normalized image appearance (as illustrated by the face databases) or in the geometry of image features (as illustrated by the hand tools and fish databases). While RMS error seems to provide a useful metric for perceptual similarity based on shape or appearance, it is inappropriate for measuring texture similarity. We require a texture model whose parameters are close when two images are perceptually close, and which are not close otherwise. The model is successful if distances between its parameters correspond to ordering images by their perceptual similarity. It is also desirable that the model parameters correspond to semantic attributes of patterns, such as periodicity or randomness.

Picard and Liu [43] have therefore developed a new model based on the Wold decomposition for regular stationary stochastic processes in 2-D images [15]. If an image is assumed to be a homogeneous 2-D discrete random field, then the 2-D Wold-like decomposition is a sum of three mutually orthogonal components: a harmonic field, a generalized-evanescent field, and a purely-indeterministic field. These three components are illustrated in Figure 7 by three textures, each of which is dominated by one of these components. Qualitatively, these components appear as periodicity, directionality, and randomness, respectively.

The motivation for choosing a Wold-based model, in addition to its significance in random field theory, is its interesting relationship to independent psychophysical findings of perceptual similarity. Noteworthy is a recent study by Rao and Lohse where humans grouped patterns according to perceived similarity [45]. The three most important similarity dimensions identified in this study were repetitiveness, directionality, and complexity. These dimensions might be considered the perceptual equivalents of the harmonic, evanescent, and indeterministic components, respectively, in the Wold decomposition. A final reason for choosing the Wold decomposition is that it produces compact texture descriptions that preserve most of a texture's perceptual attributes [49]. The result of a reconstruction from Wold components is shown in Figure 1(c).

6.1 Wold-based representations

The Wold decomposition is based on a 1938 theorem by H. Wold for 1-D random processes. This theorem states that any random process can be written as the sum of two processes, one that can be predicted by a linear filter with zero mean-squared error (deterministic), and one which is regular [52] (indeterministic). Moreover, these two processes will be mutually orthogonal. In terms of the 1-D spectrum, these two processes correspond to the discrete part of the spectrum and the continuous part of the spectrum, respectively.

In two dimensions, it is possible to have discontinuity in both dimensions, continuity in one dimension with discontinuity in the other, or continuity in both dimensions. Corresponding essentially to these three cases, the Wold theorem for a 2-D random field, $\{y(m,n)\}$, $(m,n) \in \mathcal{Z}^2$ yields a decomposition into three processes [20]. This decomposition can be formulated as a linear prediction problem. Let $\hat{y}(m,n)$ be the projection of $y(m,n)$ on the Hilbert space spanned by all the sam-

(a)

(b)

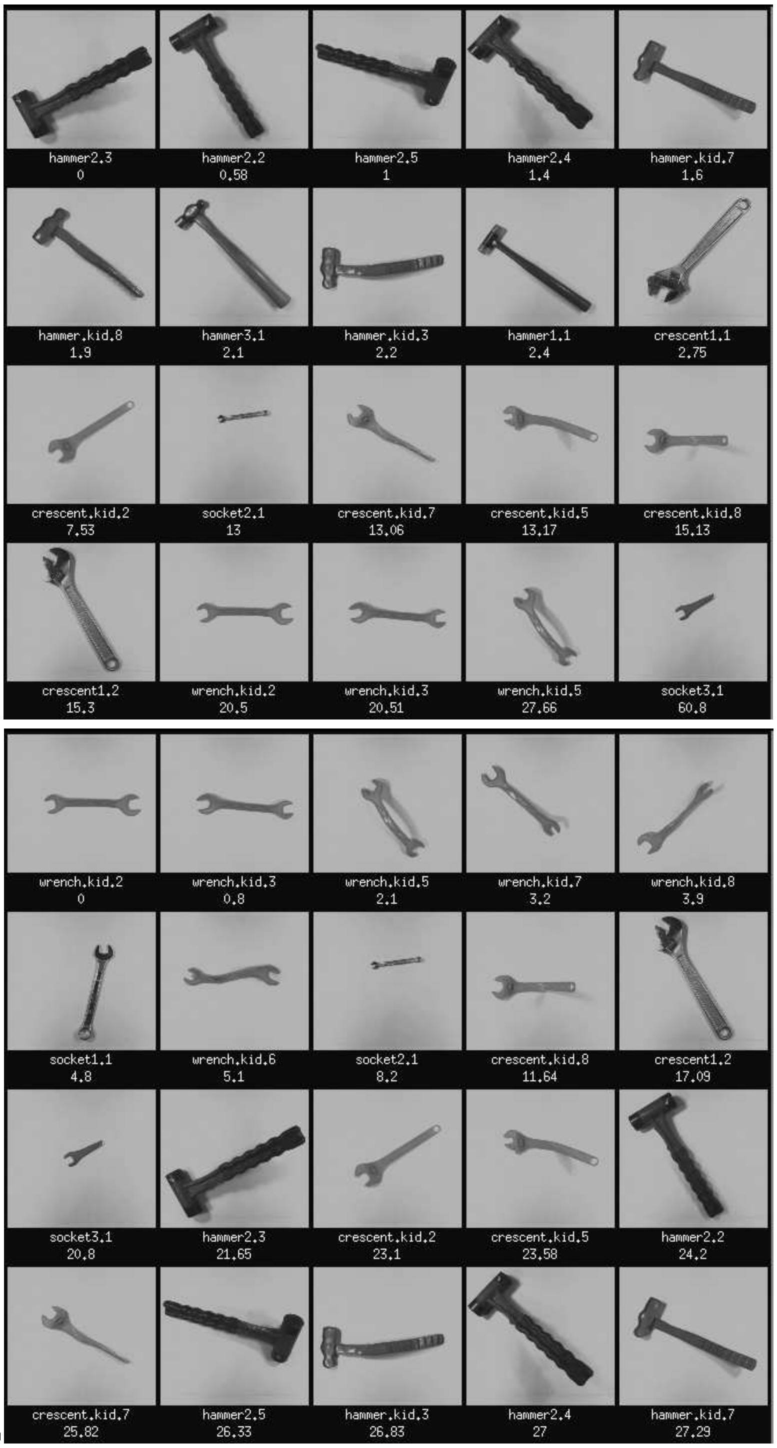


Fig. 5. In these two examples the user selected the image at the upper left, and Photobook returned the remaining images sorted by shape similarity. Images were preprocessed by extracting silhouettes from each tool image and finding corresponding contour points. The eigenmode strain energy was then used to measure similarity between the different hand tools; this statistic is shown below each image.

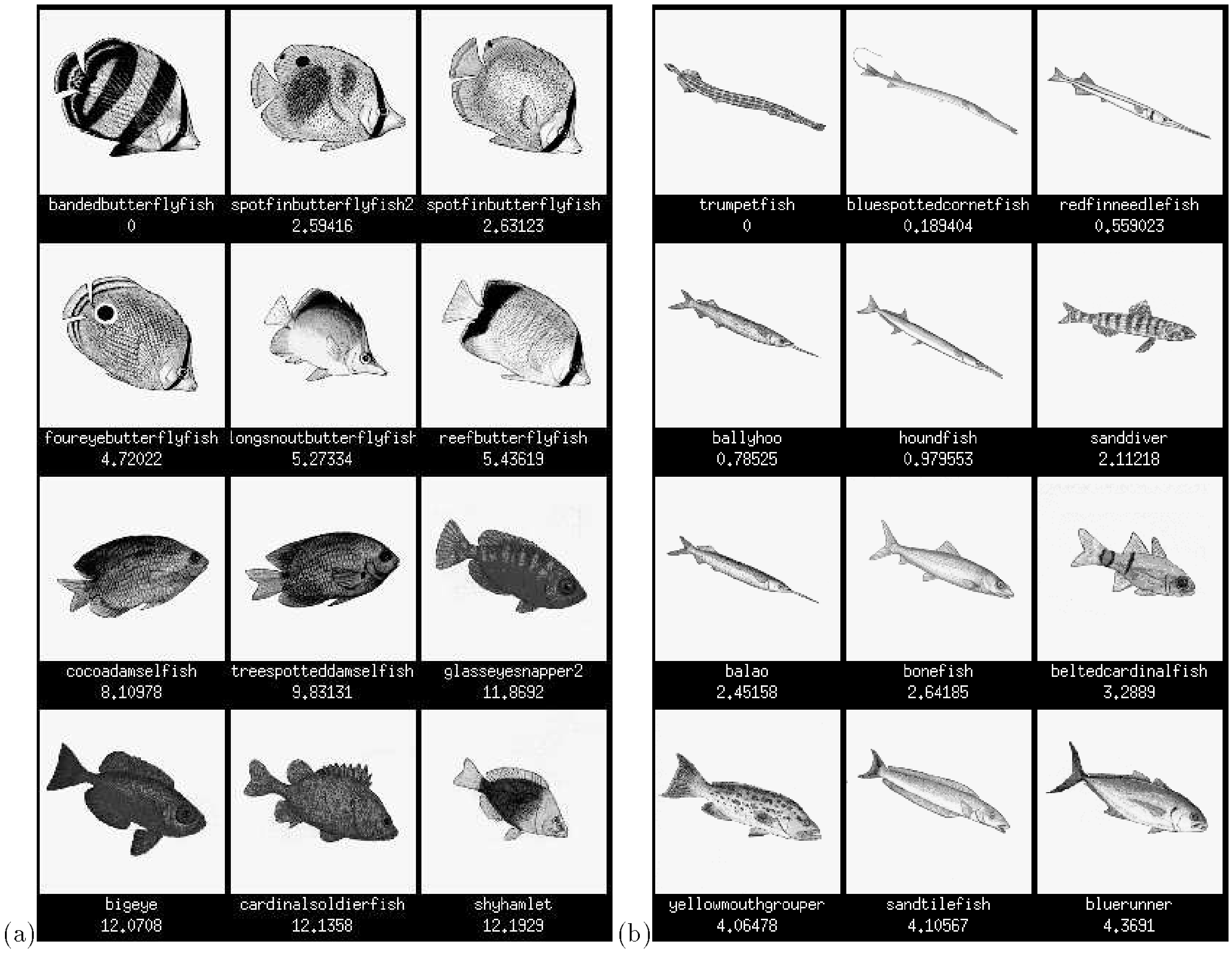


Fig. 6. Ordering fish shapes in terms of shape similarity to a user-selected fish image. In these two examples the user selected the image at the upper left; Photobook returned the remaining images sorted by similarity from left to right, top to bottom. As can be seen, the system correctly retrieved fish shapes that appear to be in the same taxonomic class.

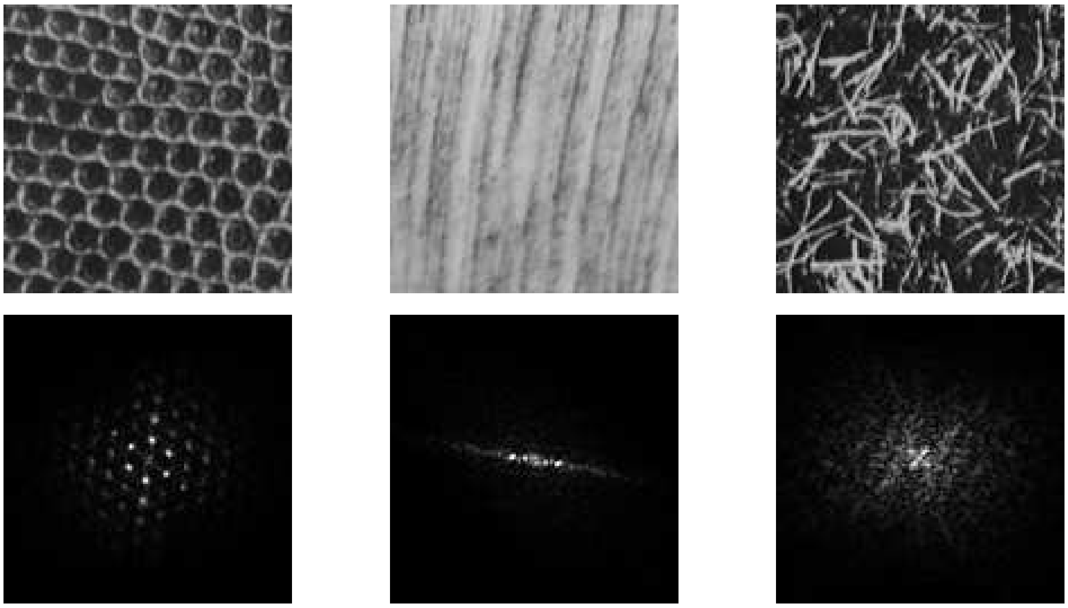


Fig. 7. The Wold decomposition transforms textures into three orthogonal components: harmonic, evanescent, and random. The upper three textures illustrate these components; below each texture is shown its DFT magnitude.

ples in the "past" of (m, n) where the implied spatial ordering is with respect to the non-symmetric half-plane (NSHP) neighborhood [15]. If the innovation field $\{u(m,n) = y(m,n) - \hat{y}(m,n)\}$ vanishes, then $\{y(m,n)\}$ is *deterministic*; else, it is *regular.* If $\{y(m,n)\}$ is regular and spans the same Hilbert space as its innovation field then it is *purely-indeterministic.* However, it may be regular and still not be purely-indeterministic.

In 2-D, a family of NSHP neighborhoods can be defined whose boundary lines are of rational slopes. With respect to each neighborhood in the family, there may exist in the corresponding deterministic field, an **evanescent** subfield due to the presence of nonzero row-to-row innovations within that deterministic field. The linear combination of all these evanescent fields is called a *generalized evanescent* field. When a deterministic field has no such innovations, then it is *half-plane deterministic.*

For any regular homogeneous random field $\{y(m,n)\}$, the 2-D Wold decomposition can be uniquely represented by:

$$y(m,n) = p(m,n) + g(m,n) + w(m,n), \quad (8)$$

where field $\{p(m,n)\}$ is half-plane deterministic, field $\{g(m,n)\}$ is generalized evanescent, and field $\{w(m,n)\}$ is purely-indeterministic, Fields $\{p(m,n)\}$, $\{g(m,n)\}$, and $\{w(m,n)\}$ are mutually orthogonal. Field $\{w(m,n)\}$ has a moving average representation, which we will exploit in the Wold-based model for Photobook:

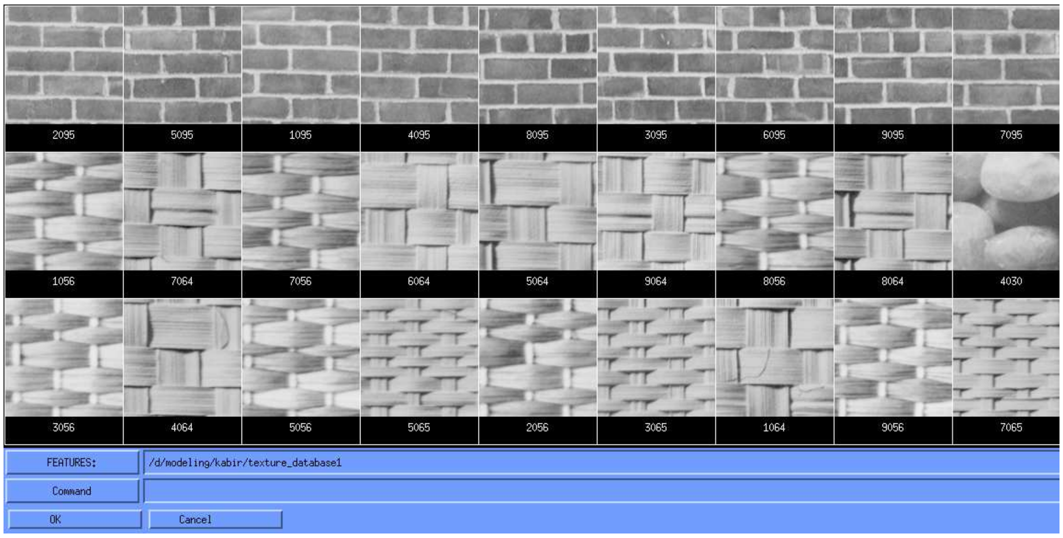
$$w(m,n) = \sum_{(0,0)\preceq(k,l)} a(k,l)u(m-k, n-l), \quad (9)$$

where $\sum_{(0,0)\preceq(k,l)} a^2(k,l) < \infty$ and $a(0,0) = 1$. The innovation field $\{u(m,n)\}$ is white.

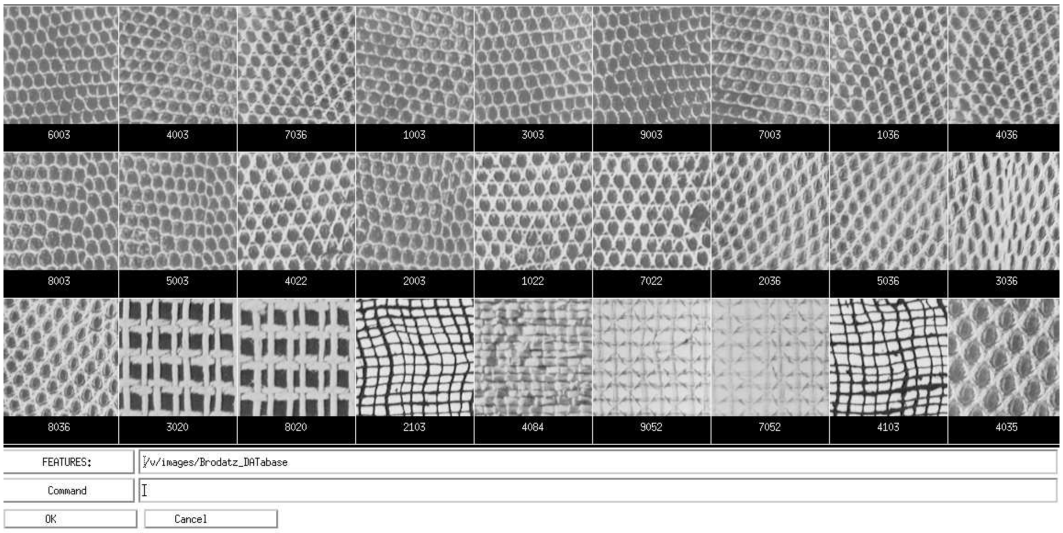
In estimating the Wold features, we also exploit the dual relationship between the 2-D Wold decomposition and the decomposition of the spectral distribution function of a regular homogeneous random field. Let us define all spectral functions on the rectangular region $\left[-\frac{1}{2}, \frac{1}{2}\right] \times \left[-\frac{1}{2}, \frac{1}{2}\right]$. Let $F_y(\xi, \eta)$ be the spectral distribution function of a regular homogeneous random field $\{y(m,n)\}$, and let $F_y^s(\xi,\eta)$ denote the singular part of $F_y(\xi,\eta)$. Let $F_p(\xi,\eta)$, $F_g(\xi,\eta)$, and $F_w(\xi,\eta)$ be the spectral distribution functions of the half-plane deterministic, the generalized evanescent, and the purely indeterministic components of $\{y(m,n)\}$. Then function $F_y(\xi,\eta)$ can be uniquely represented as

$$F_y(\xi,\eta) = F_p(\xi,\eta) + F_g(\xi,\eta) + F_w(\xi,\eta). \quad (10)$$

where function $F_p(\xi,\eta) + F_g(\xi,\eta) = F_y^s(\xi,\eta)$ is singular with respect to the Lebesgue measure and function $F_w(\xi,\eta)$ is absolutely continuous.



(a)



(b)

Fig. 8. Comparison of ordering over the 1008 images in the Brodatz texture database via parameters of Wold model. In each display, the images are ordered by their distances from the image in the upper left. The coefficients in (a) employ the random and directional components. In (b) only the harmonic components are used, with invariance applied for different rotations.

Thus the decomposition of the deterministic and the purely-indeterministic components of a regular homogeneous random field can be achieved by separating the singular and the absolutely continuous components of the spectral distribution of the random field. This is known as Lebesgue decomposition. The orthogonality of the two components allows them to be treated separately.

The model implementation used in Photobook consists of three stages. The first stage determines if there is strong periodic (or nearly periodic) structure. Although highly structured textures may contain all three Wold components, their harmonic components are usually prominent and provide good features for comparison. Not only are harmonics more salient than the other components (agreeing with Rao and Lohse's ordering of the three texture dimensions) but they are also the quickest to compute.

The second stage of processing occurs for periodic images on the peaks of their Fourier transform magnitudes. An algorithm is implemented to first estimate the location of large local maxima and then extract the fundamental frequencies of all harmonic peaks. The direction of the harmonic frequency that is closest to the origin is regarded as the main orientation angle of the texture. Rotations and other transformations may be applied to the peaks to align them into a sort of "generic view" image before further comparison. Applying the transformations to the peaks incurs markedly less computation than applying them to the entire image.

The third stage of processing is applied when an image is not highly structural. This stage approximates the finding of the two less salient dimensions identified in the study of Rao and Lohse, the directional and complexity components. The number of dominant orientations is estimated via steerable filtering and a decision process based on thresholding orientation histograms, as described in [41]. Only the textures which possess the same number of main orientations are subsequently compared by examining the Wold complexity component. The complexity component is modeled by use of a multiscale simultaneous autoregressive (SAR) model, whose parameters are estimated using the process of Mao and Jain [32]. The SAR parameters of different textures are compared using the Mahalanobis distance measure. These final stages of processing are the most computationally costly part of the procedure, and can be omitted to result in substantial savings if the previous stage indicates that the harmonic information is sufficient for a given task.

Computer code implementing this proceedure, together with technical reports providing additional detail, is available by anonymous FTP from whitechapel.media.mit.edu.

6.2 Database experiments

The illustrations here are from experiments run on the Brodatz database, which consists of 1008 non-overlapping texture patches cropped from all 112 images of the Brodatz Album [7]. Each Brodatz texture provides nine 128×128 subimages in 8-bit gray levels. This collection of natural textures exhibits large variety, including many inhomogeneous patterns not usually included in texture studies formed from small subsets of the Brodatz collection. The database therefore provides a challenge to traditionally homogeneous image models.

Figure 8 shows results of some experiments with the new Wold-based model. Figure 8 (a) illustrates a search on a brick pattern, with the result that Photobook finds all nine of the brick patterns in the database. Note that the next most similar images are similarly structured, with two predominant orientations. This result is typical; for this texture database the "most similar" texture found was another subimage of the same Brodatz image 83% of the time, and 90% of the time the "most similar" texture was of the same semantic category (*e.g.*, both lace, although not from the same Brodatz texture image). In (b) Photobook again fills the first row with reptile-skin patterns, and the next most similar images have perceptually similar structure, despite rotations.

7 Other Issues

7.1 Combining and Developing Models

We have described three methods for compactly describing and searching image content. Two, appearance and shape, are intended for comparing "things" (*e.g.*, faces, cars, fish, hand tools); the third, texture, is intended for comparing "stuff" (*e.g.*, trees, clouds, cloth, grass). The examples presented above have been selected to demonstrate both the possibility and the effectiveness of developing semantics-preserving image compression methods for image database search.

However, we do not mean to suggest that these examples cover the range of possibilities. Rather, we suggest that these tools be thought of as three *general* methods for developing compact, class-specific representations suitable for image database search. For any particular semantic class (*e.g.*, cars, clouds, or crowds) we can train each of these three types of description (appearance, shape, and texture). This is accomplished by collecting a set of training examples, and characterizing the mean and range of appearance, shape, or texture parameters.

For instance, we have built appearance models for 2-D images of eyes, hands, cars, 1-D sound signals, and 3-D MRI data. We have also built shape models for 2-D images of rabbits, hands, heads, heart X-rays, 1-D sound signals, 3-D voxel data, and 3-D range data. We have built texture models of 2-D images such as paintings, grass, clouds, city buildings and 1-D sound signals such as copier noise and applause.

Nor do we mean to suggest that these techniques must be applied only to simple grey-level images. For instance, we have also applied each of these techniques to color images and edge images. The simplest method (illustrated below) is to consider the color and edge images as additional image data appended to the grey-level data; for instance, to consider an n x n 24-bit color image as $3n$ x n 8-bit image data. This wider image is then subjected to correlation analysis, spectral decomposition, or shape analysis of the additional regions.

For instance, Figure 9 shows using an appearance description on color video keyframes. In this example n x n 24-bit color video keyframes were extracted, and the eigenvectors of the correlation matrix of the $3n$ x n 8-bit data were obtained. These eigenimages therefore describe the general spatial and color layout of the keyframe. In Figure 9 we see the results of a search among 365 keyframes for those similar to the one at the upper left; the appearance description Photobook used to conduct this search might be loosely translated into English as "find keyframes with a pink and red blob on the left with a beige background". Figure 10 shows another color keyframe search; however, this time the keyframes are compared using the periodic texture components of the red, green, and blue channels. The texture description used for this search might be (very) loosely translated into English as "find keyframes with similar vertical and horizontal edge structure in all the color channels".

Finally, we also do not mean to suggest that these techniques should be used in isolation, as in the previous examples. All three methods can be used for any particular image, either separately, in a wide range of combinations, or in conjunction with text annotations.

For instance, Figure 11 shows an example where 3-D voxel data of human ventricles was analyzed in Photobook by combining an eigenmode shape description with an eigenimage appearance description. In this example the shape description was first used to normalize for the effects of overall head shape, and then the appearance description was used to compare ventricle shape. Using this shape-and-appearance approach allowed us to more accurately characterize subtle shape differences such as occur between Alzheimer's disease and normal pressure hydrocephalus disease [36].

7.2 Detection and Preprocessing

No matter how well one can describe appearance, shape, and texture, there is still the question of finding the things (or stuff) to be described. That is, where exactly is the face that is to be described? The bunch of trees? The fish? In many real-world applications, these questions are the most difficult of all.

Fortunately in image and video database applications it is often acceptable to accomplish this step either by hand, or heuristically. For instance, for the tool and fish databases, the background was sufficiently simple that grey-level thresholding yielded a good outline of the shape. The point-to-point correspondences were determined automatically, as described in references [47, 48].

We are also fortunate that, at least in the case of video, it is relatively easy to use motion and color changes to help find things and stuff of interest. This was illustrated by Figure 2, and is discussed more fully in references [11, 55, 12].

However, we can also draw on our framework of semantics-preserving compression to address this problem. Recall that our basic approach for representing specific classes of interest is to use a prototype(s) and the smallest possible set of parametric variations or deformations. Such a representation is very good at describing the signals it was trained on, but is quite bad at describing other signals. This fact allows us to recast the problem of finding instances of models as one of *detection*, that is, if a model can accurately describe some portion of an image, then it is very likely to be an *appropriate* representation of that image data.

This concept is easiest to illustrate in the case of appearance descriptions. For an appearance de-

Fig. 9. An example of using appearance to sort keyframes from a video database; the sort is by general layout and color distribution.

Fig. 10. An examples of using texture to sort keyframes from a video database; sorting is by similarity of the periodic components of texture in each of the RGB channels.

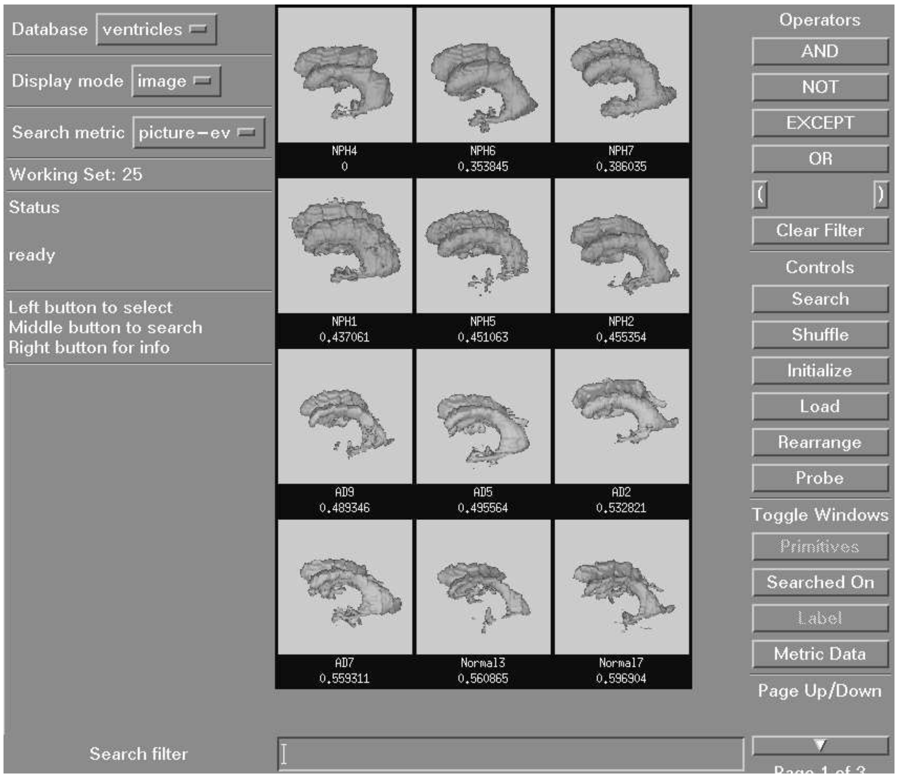


Fig. 11. Combining shape and appearance descriptions to search NMR data, from reference [36].

scription, we use example images to calculate a mean image Ψ and eigenimages $\mathbf{u}_k$. These define a small parametric space which contains most of the variation among the training images. Each image is described by a few coefficients ω_k, which are the image's projection onto the eigenimages. The range of possible appearances that exist within the training images is efficiently and accurately characterized by the distribution of the ω_k, together with the mean and eigenimages.

Thus if an image has most of its energy within the subspace spanned by the eigenimages, and its projection coefficients are typical of the training images, then it is visually similar to the training images. This allows us to detect instances of models by searching for parts of the image that can be efficiently encoded by the model. The search process can be made surprisingly efficient by appropriately arranging the order in which we calculate the ω_k. We have used this approach both for finding a wide variety of "things" (including eyes, cars, roads, *etc.* [34]) and "stuff" (including sky, trees, buildings, *etc.* [44]). For further detail see references [34, 44]. These and related references are available by anonymous FTP from whitechapel.media.mit.edu.

7.3 Labels, Knowledge Representation, and Context

So what does all this have to do with semantics? It seems clear that detecting "a face that looks like John" or finding "a patch of burlap-like texture" has some semantic content, especially if humans agree that the image really *does* look like John or burlap. However, this still seems very different from the image content that a photographer or a knowledge representation researcher would talk about.

The difference stems from our choice to *avoid* addressing the unsolved problems of meaning and context. We instead are working to derive word-like primitives from images, rather than whole sentences. We make this choice because we have observed that people are nonlinear time-varying systems whose behavior depends on unknown internal states.

For instance, a human may label the same scene differently at different times, and expect different regions to be recognized as similar when his or her goals change. Human judgment of image similarity can be perceptual or semantic, and can be influenced by culture, context, and personal preference.

Given these influences, it seems to us premature to look for some universal measure of similarity within pictures. In restricted applications, *e.g.* inspection for a mark of a particular size and shape, similarity matches can be made quite precisely. But in general picture retrieval, at least for the immediate future, there are important reasons to keep the human in the system, *i.e.*, to make semi-automated tools.

We therefore assume that there is neither one model that will be optimal for recognizing and annotating pictures, nor is there a unique non-overlapping arrangement of labels that users will want to use to annotate a picture. This departs from the traditional computer vision approach of using one model to segment an image into non-overlapping regions before assigning labels to the regions. Instead, we assume that a user might assign multiple labels to possibly overlapping regions.

This fits nicely with the detection paradigm for finding instances of models. When we find an image region that looks like clouds, we annotate it as such. But it might also be shaped like a fish, or a subregion of it might look like a face. There is nothing wrong with multiple labels in our framework. To be useful in the real world, we must be able to capture such varied notions of similarity.

So rather than attempting to automatically parse the full semantic structure of a signal, we instead rely on interactions with the user to define the semantic scope and interrelations of image primitives. We provide the user with primitives such as model-specific detection and perceptual similarity, and use relevance feedback to learn what relations are valid in this particular context [44].

8 Conclusion

The Photobook system is a set of interactive tools for browsing and searching images and image sequences. The key idea behind this suite of tools is *semantics-preserving image compression*, which reduces images to a small set of perceptually-significant coefficients.

We have developed three fairly general approaches to constructing semantics-preserving representations. When searching for "things," we can use variations on the Karhunen-Loève transform to derive optimally-compact representations for either appearance or shape. When searching for "stuff" we have shown the utility of the Wold transform for decomposing signals into compact, perceptually salient textural descriptions. By combining these representational methods with text annotations in an interactive framework, Photobook provides users with a sophisticated and efficient utility for database search based on image content.

Acknowledgment: The work described here was funded by BT (British Telecom). We also wish to thank Fang Liu for her work in developing the Wold implementation, Baback Moghaddam, Thad Starner, and Matthew Turk for their work in developing the eigenface implementation, John Martin for his ventricles example, and Tom Minka, Peter Gast, Bradley Horowitz, and Thad Starner for their work in developing the PHOTOBOOK user interface.

References

[1] E. Adelson and J. Bergen, "The Plenoptic Function and the Elements of Early Vision," in: M. Landy and J. A. Movshon, (eds) *Computational Models of Visual Processing*, MIT Press (1991).

[2] ACM SIGIR. *Proceedings of International Conference on Multimedia Information Systems*, Singapore, 1991.

[3] D. Ballard and C. Brown. *Computer Vision.* Prentice Hall, 1982

[4] E. Binaghi, I. Gagliardi, and R. Schettini. "Indexing and fuzzy logic-based retrieval of color images." In *Visual Database Systems, II, IFIP Transactions A-7,* pages 79-92.

[5] W. E. Blanz, D. Petkovic, and J. L. Sanz. *Algorithms and Architectures for Machine Vision.* ed. C.H. Chen, Marcel Decker Inc., 1989.

[6] T. Breuel, Indexing for Recognition from a Large Model Base, M.I.T. Artificial Intelligence Laboratory Memo 1108, August 1990

[7] P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover, New York, 1966.

[8] C. C. Chang and S. Y. Lee. "Retrieval of similar pictures on pictorial databases." *Pattern recognition,* 24(7):675- 680, 1991.

[9] C-C. Chang and T-C. Wu. "Retrieving the most similar symbolic pictures from pictorial databases." *Information Processing and Management,* 28(5):581-588, 1992.

[10] Z. Chen and S-Y. Ho. "Computer vision for robust 3D aircraft recognition with fast library search." *Pattern Recognition,* 24(5): 375-390, 1991.

[11] T. Darrell and A. Pentland, "Robust Estimation of a Multi-Layer Motion Representation", in *Proceedings IEEE Workshop on Visual Motion*, pp. 173-177, 1991.
longer version available as M.I.T. Media Laboratory Perceptual Computing Technical Report No. 163

[12] T. Darrell, P. Maes, B. Blumberg, and A. Pentland, "A Novel Environment for Situated Vision and Behavior," *IEEE Workshop on Visual Behaviors* pp. 68-72, Seattle, WA., June 19, 1994.

[13] S. Smoliar, and H. Zhang, "Content-Based Video Indexing and Retrieval," *IEEE Multimedia Magazine*, Vol. 1, No. 2, pp. 62-72, 1994.

[14] R. Duda and P. Hart *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.

[15] J. Francos "Orthogonal Decompositions of 2-D Random Fields and their Applications for 2-D Spectral Estimation", *Signal Processing and its Applications*, pp. 287-327, N.K. Bose and C.R. Rao (eds.), North-Holland., 1993.

[16] P. Gast, "Integrating Eigenpicture Analysis with an Image Database," *M.I.T. Bachelors Thesis, Computer Science and Electrical Engineering Deptartment*, Advisor: Alex Pentland, 1993.

[17] W. I. Grosky, P. Neo, and R. Mehrotra. "A pictorial index mechanism for model-based matching." *Data and Knowledge Engineering,* 8:309-327, 1992

[18] K. Haase, "FRAMER: A Portable Persistent Representation Library," *Proceedings of the AAAI Workshop on AI in Systems and Support, Am. Asso. for AI*, 1993.

[19] K. Haase, "AI in Service and Support: Bridging the Gap", Haase, *Proceedings of Am. Asso. AI*, 1993.

[20] H. Helson and D. Lowdenslager, "Prediction Theory and Fourier Series in Several Variables.II", *Acta Mathmatica*, Vol. 196, pp. 175-213, 1962.

[21] K. Hirata and T. Kato. "Query by visual example," In *Advances in Database Technology EDBT '92, Third International Conference on Extending Database Technology,* Vienna, Austria, March 1992. Springer-Verlag.

[22] M. Ioka. "A method of defining the similarity of images on the basis of color information," Technical Report RT-003 0, IBM Tokyo Research Lab, 1989.

[23] M. A. Ireton and C. S. Xydeas. "Classification of shape for content retrieval of images in a multimedia database," In *Sixth International Conference on Digital Processing of Signals in Communications,* pages 111-116, Loughborough, UK, 2-6 Sept., 1990. IEE.

[24] H. V. Jagadish. "A retrieval technique for similar shapes," In *International Conference on Management of Data, SIGMOD 91,* pages 208-217, Denver CO, May 1991. ACM.

[25] R. Jain and W. Niblack. NSF workshop on visual information management, February 1992.

[26] T. Kato, T. Kurita, H. Shimogaki, T. Mizutori, and K. Fujimura. "A cognitive approach to visual interaction. In *International Conference of Multimedia Information Systems," MIS '91,* pages 109-120. ACM and National University of Singapore, January 1991.

[27] Y. Lamdan and H. J. Wolfson. "Geometric hashing: A general and efficient model-based recognition scheme," In *2nd International Conference on Computer Vision (ICCV),* pages 238-249, Tampa, Florida, 1988. IEEE.

[28] S-Y. Lee and F-J. Hsu. "2D C-string: A new spatial knowledge representation for image database systems," *Pattern Recognition,* 23(10):1077-1087, 1990.

[29] S-Y. Lee and F-J. Hsu. "Spatial reasoning and similarity retrieval of images using 2D c-string knowledge representation," *Pattern Recognition,* 25(3):305-318, 1992.

[30] A Lippman. "Semantic bandwidth compression," *Picture Coding Symposium*, 1981.

[31] P. McLean, "Structured Video Coding," *M.I.T. Masters Thesis*, Advisor: Andrew Lippman, 1989.

[32] J. Mao and A. Jain, "Texture Classification and Segmentation using Multiresolution Simultaneous Autoregressive Models", *Pattern Recognition,* Vol. 25, No. 2, pp 173-188, 1992.

[33] R. Mehrotra and W. I. Grosky. "Shape matching utilizing indexed hypotheses generation and testing," *IEEE Transactions of Robotics and Automation,* 5(1):70-77, 1989.

[34] B. Moghaddam and A. Pentland, "Face recognition using view-based and modular eigenspaces for Identification And Inspection of Humans, " *SPIE Conf. on Automatic Systems*, San Diego, July 1994

[35] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, and P. Yanker. "The QBIC project: Querying image s by content using color, texture, and shape," In *IS & T/SPIE 1993 International Symposium on Electronic Imaging: Science & Technology,, Conference 1908, Storage and Retrieval for Image and Video Databases,* February 1993.

[36] J. Martin, A. Pentland, and R. Kikinis "Shape Analysis of Brain Structures using Physical and Experimental Modes," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 752-755, Seattle, WA., June 1994.

[37] A. Pentland and S. Sclaroff "Closed-Form Solutions For Physically Based Shape Modeling and Recognition." *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 13, No. 7, pp. 715-730.

[38] A. Pentland, R. Picard, G. Davenport, R. Welsh, "The BT/MIT Project on Advanced Image Tools for Telecommunications: An Overview," *ImageCom '93, 2nd International Conference on Image Communications*, Bordeaux, France, 23-25 March, 1993.

[39] A. Pentland, B. Moggadam, and T. Starner, "View-Based and Modular Eigenspaces for

Face Recognition," *IEEE Conf. Computer Vision and Pattern Recognition,* pp. 84-90, Seattle, WA, June 1994

[40] R. W. Picard "Random Field Texture Coding," *Society for Information Display International Symposium Digest*, Vol XXIII, May 1992, pages 685–688.

[41] R. W. Picard and M. Gorkani. "Finding perceptually dominant orientations in natural textures." *Spatial Vision* Vol. 8, No. 2, pp. 221-253, 1994.

[42] R. W. Picard and T. Kabir. "Finding similar patterns in large image databases." *Proc. ICASSP*, Minneapolis, MN, Vol. V, pp. 161-164, 1993.

[43] R. W. Picard and F. Liu, "A new Wold ordering for image similarity," *IEEE Conf. on ASSP,* Adelaide, Australia, April, 1994.

[44] R. W. Picard and T. P. Minka, "Vision Texture for Annotation" *ACM/Springer-Verlag Journal of Multimedia Systems*, to appear.

[45] A. R. Rao and G. L. Lohse, "Towards a Texture Naming System: Identifying Relevant Dimensions of Texture," *IEEE Conf. on Visualization 1993*, San Jose, CA.

[46] L. Sirovich, and M. Kirby, "Low-dimensional procedure for the characterization of human faces," *J. Opt. Soc. Am. A*, Vol. 4, No. 3, March 1987, 519-524.

[47] S. Sclaroff and A. Pentland, "A finite-element framework for correspondence and matching," 4th International Conference on Computer Vision, pp. 308-313, May 11-14, 1993, Berlin, Germany.

[48] S. Sclaroff and A. Pentland, "Modal Matching for Correspondence and Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* to appear.
Also available as: M.I.T. Media Laboratory Perceptual Computing Technical Note No. 304.

[49] , R. Sriram, J. M. Francos and W. A. Pearlman, "Texture coding Using a Wold Decomposition Model," sl Proc. 12th IAPR Int. Conf. Pat. Rec., Jerusalem, Israel, Oct. 1994.

[50] M. Swain and D. Ballard, "Color indexing". *Int. J. of Computer Vision,* 7(1):11-32, 1991.

[51] S. Tanaka, M. Shima, J. Shibayama, and A. Maeda. "Retrieval method for an image database based on topographical structure." In *Applic. of Digital Image Processing,* Vol. 1153, pages 318-327. SPIE, 1989.

[52] *Discrete Random Signals and Statistical Signal Processing*, C. W. Therrien, Prentice-Hall, Englewood Cliffs, NJ 1992.

[53] M. Turk and A. Pentland, "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, May 1991.

[54] K. Wakimoto, M. Shima, S. Tanaka, and A. Maeda. "An intelligent user interface to an image database using a figure interpretation method." In *9th Int. Conference on Pattern Recognition,* volume 2, pages 516-991, 1990.

[55] J. Y. A. Wang and E. H. Adelson, "Layered Representation for Motion Analysis" *IEEE CVPR '93.*
Longer version available as: M.I.T. Media Laboratory Perceptual Computing Technical Report No. 228.