Content-Based Indexing of Multimedia Databases

Jian-Kang Wu

Abstract—Content-based retrieval of multimedia database calls for content-based indexing techniques. Different from conventional databases, where data items are represented by a set of attributes of elementary data types, multimedia objects in multimedia databases are represented by a collection of features; similarity of object contents depends on context and frame of reference; and features of objects are characterized by multimodal feature measures. These lead to great challenges for content-based indexing. On the other hand, there are special requirements on content-based indexing: To support visual browsing, similarity retrieval, and fuzzy retrieval, nodes of the index should represent certain meaningful categories. That is to say that certain semantics must be added when performing indexing. ContIndex, the context-based indexing technique presented in this paper, is proposed to meet these challenges and special requirements. The indexing tree is formally defined by adapting a classification-tree concept. Horizontal links among nodes in the same level enhance the flexibility of the index. A special neural-network model, called Learning based on Experiences and Perspectives (LEP), has been developed to create node categories by fusing multimodal feature measures. It brings into the index the capability of self-organizing nodes with respect to certain context and frames of reference. An icon image is generated for each intermediate node to facilitate visual browsing. Algorithms have been developed to support multimedia object archival and retrieval using ContIndex. ContIndex has been successfully applied to two applications: A facial image retrieval system, CAFIIR, and a trademark archival and registration system, STAR.

Index Terms—Indexing, content-based retrieval, multimedia, image database, image analysis, neural networks, fusion of multiple feature measures.

1 Introduction

CONTENT-BASED retrievals are most preferable in multimedia database systems. For example, a policeman would like to catch a suspect with a face that is consistent with a witness's description from a very large criminal-face image database. The content-based retrieval like that has the nature of visual, fuzzy, and similarity-based. It is visual because image, video, graphics and text are visual. Content of audio can be visualized. It is sometimes fuzzy because many contents of multimedia objects cannot be exactly described (such as the size of eyes), or the user cannot provide an exact definition when specifying the query. It is similarity-based because "find something similar to that one" is one of the most preferable query types and because the content-based query processing is based on similarity measures.

Content-based retrieval of multimedia database has attracted wide interest among researchers from different fields and applications. For example, object recognition using content-based retrieval has been demonstrated by Grosky [10] and Wu [20]. Bach and Jain [1] utilized visual routines for both insertion and retrieval. Petrakis and Orphanoudakis [4] developed a methodology for the retrieval of medical images based on spatial relationships and properties of objects. Pentland et al. [3] described a photobook system, which is a set of interactive tools for browsing and searching images for face, shape, and texture. Wu and his colleagues have developed a content-based retrieval engine [5] and used it to build a facial image retrieval system for criminal identification [6] and a system for trademark archival and registration [7].

People in hypertext and hypermedia have also shown interest in content-based retrieval [11], [12]. For example, Lewis et al. extended the key of generic links from normalized text to content of images, video and audio in order to facilitate the navigation of the multimedia data.

In many applications, the databases may be very large. For example, there may be several hundred thousands facial images in a criminal identification system. Indexes are crucial for those large databases to speed up the retrieval. On the other hand, visual, fuzzy and similarity queries in those large content-based databases cannot be implemented using conventional indexing techniques such as B-trees and inverted files, which are proved to be very effective in traditional databases to index attributes and text. This is because the feature measures of object contents are complex and are usually multidimensional and multimodal (will be discussed in more detail in the next section). Conventional indexing techniques are based on individual keys, which are definite and not visual. For the purpose of handling complex feature measures, there have been researches to extend the concept of indexing using abstraction and classification [9], [10], [20], [8]. To handle multimodal feature measures, to gain self-organization and learning capabilities in indexing, we proposed and developed a Content-based Indexing (ContIndex) method for indexing multimedia objects.

• *The author is with the Institute of Systems Science, National University of Singapore, Singapore 0511. E-mail: jiankang@iss.nus.sg.*

Manuscript received 1 June 1995; revised 26 Mar. 1996.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number 104428.

ContIndex is defined by adapting the concept of tree classifier. A combination of LEP (Learning based on Experiences and Perspectives) neural network model [16] and Kohonen's Self-Organization Feature Map [13] is used to generate spatially self-organized nodes for ContIndex tree on multimodal complex feature measures. ContIndex technique can be readily applied to fuzzy indexing on multivariate fuzzy membership functions.

In the following sections, the concept of content-based indexing and the challenges of content-based indexing of multimedia objects are discussed in Section 2. The ContIndex method is presented in Section 3, followed by neural networks for fusion of multimodal feature measures in Section 4. Experimental results using the ContIndex method are described in Section 5.

2 Challenges of Content-Based Indexing of Multimedia Objects

2.1 Content-Based Retrieval

For completeness of the discussion, let us start from the multimedia object definition in [5] as follows:

Multimedia Object (MOB) can be defined using a six-tuple $O_{mob} = \{U, F, M, A, O^p, S\}$, where:

- U is multimedia data component.
- $F = \{F^1, F^2, ...\}$ represents a set of features derived from data. A feature F^i can be either numerically characterized by feature measures in feature spaces

$$F_1^i \times F_2^i \times \ldots \times F_n^i,$$

or conceptually described by a set of concepts.
- $M^j = \{M_1^j, M_2^j, \ldots\}$ represents the interpretation of features F^i, $i = 1, 2, ...$
- A stands for a set of attributes or particulars of O_{mob}.
- O^p is a set of pointers or links, and is expressed as,

$$O^p = \{O_{\sup}^p, O_{sub}^p O_{other}^p\}$$

are three type of pointers/links pointing/linking to superobjects, subobjects, and other objects, respectively.
- S represents set of *states* of O_{mob}.

Content of a multimedia object is the content of its data set U, which is restricted to a certain set of features F^i, $i = 1, 2, ...$ of the object and characterized by feature measure sets F_k^i, $k = 1, 2, \ldots$ and further described by concept sets M^j, $j = 1, 2, ...$ In many cases, feature measures are vectors and written as $F_j^i = \{x_1, x_2, \ldots, x_n\}^T$.

For example, representation of a facial image can be done by focusing our attention to some visual features such as chin, hair, eyes, eyebrows, nose, and mouth. To characterize eyes, we need to extract measures such as area, fitting parameters to a deformed template. These feature measures are vectors and can be considered as points in feature spaces. Eyes can also be described by a set of concepts such as big eyes, medium eyes, or small eyes. The concepts "big," "medium," and "small" are interpretations of facial feature "eyes."

Fig. 1 shows a representation hierarchy for images in content-based image databases. In image archival phase, a bottom-up process is performed to derive from the original image data the feature measures of regions-of-interest, and interpretations if necessary. This bottom-up process consists of three steps, namely, segmentation, feature extraction, and concept mapping. It performs information abstraction, and provides keys for easy access of large image data. In retrieval phase, the image data are accessed through their feature measures (similarity query) or interpretations (descriptive query), which are considered as keys from database point of view.

Content-based retrieval usually does not access the data through attributes A, or directly through the data component U. Instead, it operates on feature measures. Content-based retrieval is to find the best matches from large databases for a given query object. The best match is defined in terms of similarity measure. Since the contents of objects

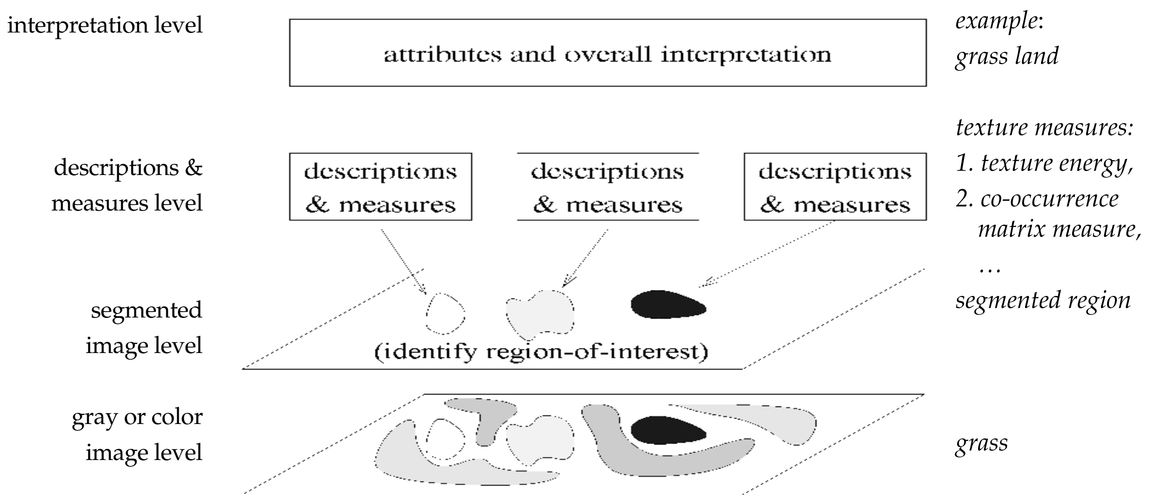


Fig. 1. Image representation hierarchy. To archive images into content-based image database, images are first segmented to identify regions of interest. Feature measures are then extracted from the image data within these regions. Interpretations can be finally generated by mapping of the feature measures into a set of concepts.

are represented by features, the similarity is then defined with respect to these features:

$$sim_{overall}(O^q, O) = w_1 \; sim\left(F_q^1, F^1\right) + w_2 \cdot sim\left(F_q^2, F^2\right) + \ldots \quad (1)$$

where w_i denotes the weight for ith feature, and $sim\left(F_q^i, F^i\right)$ denotes the similarity between the query object and an object in the database with respect to ith feature. Here we simply express the similarity between objects as a linear combination of the measures of their common and distinctive features [15].

Care should be taken when evaluating the similarity for each feature of the objects. Firstly, similarity depends on context and frame of reference, for example, as in the question, "how similar are person A's eyes and person B's eyes with respect to size?" Secondly, objects in real world are complicated and require more than one feature measure for their characterization. For example, to characterize eyes, we can perform principal component analysis (PCA) to eye images to generate a PCA coefficient vector for each eye image. We can fit a deformable template to an eye to get a few parameters as a measure vector [21]. Landmark coordinates around eyes can be also considered as measures. These three measures of feature "eyes" $F_1^{eyes}, F_2^{eyes}, F_3^{eyes}$ reflect different perspectives of the eyes, are derived using different methods, are interpreted differently. Therefore, we consider them of different modal. That is why we say "multimodal feature measures."

2.2 Content-Based Indexing

Content-based retrieval systems do not necessarily have content-based indexing. *Content-based indexing* is aimed to create indexes in order to facilitate fast content-based retrieval of multimedia objects in large databases. Generally speaking, an index consists of a collection of entries, one for each data item, containing the key for that item, and a reference pointer which allows immediate access to that item. To accelerate searching of specific data items, most database systems use a tree indexing mechanism. In an index tree the intermediate nodes are abstractions of their child nodes.

The index in traditional databases is quite simple. It operates on attributes, which are of primitive data types such as integer, float, and string. For example, to build a binary index tree on age of people in a database, the first two branches can be created for "age ≥ 35" and "age ¡ 35." Here the operation is simple and the meaning is definite and obvious. The situation becomes very complex in content-based indexing, which operates on complex feature measures. Let us take chin of faces as an example. In the face image system in [6], the chin is characterized by the first 16 coefficients of principal component analysis and five landmark coordinates. To create index tree, the boundaries among branches are very complex and the meaning of the index tree become vague. We cannot imagine how traditional indexing methods can be applied here.

The three issues of similarity, namely, objects are represented as collections of features, similarity depends on context and frame of reference, and features are characterized by multiple multimodal feature measures, have posed special requirements on content-based indexing algorithm. The challenges for content-based indexing are:

- The index must be created using all features of an object class, so that visual browsing of the object class is facilitated, and similarity retrieval using similarity measure, in (1), can be easily implemented.
- The context and frame of reference in similarity evaluation suggest that nodes in index tree show consistency with respect to the context and frame of reference. For example, if, in a level of an index tree, the similarity is evaluated with respect to eye size, the nodes in this level will represent object categories with various eye sizes. This implies that the index tree has similar property as classification tree.
- Multiple multimodal feature measures should be fused properly to generate index tree so that a valid categorization can be possible. Two issues must be addressed here: first, one measure only is usually not adequate because of the complexity of objects. Second, to ensure the specified context and frame of reference, care must be taken in feature selection process.

ContIndex method is designed to meet these challenges. The indexing tree is defined by adapting tree classifier concept. A special neural network model has been developed to create nodes using multiple multimodal feature measures. Algorithms have been developed to support multimedia object archival and retrieval using ContIndex.

3 Content-Based Indexing of Multimedia Object

We can see, from the challenges and special requirements discussed in the last section, that ContIndex shares some characteristics with classification tree. Therefore, we adapted the classification tree definition for ContIndex. To handle multiple features of object, we introduced horizontal links in ContIndex. Iconic images are also necessary for intermediate nodes. Those iconic images will visually represent the categories of the corresponding nodes and facilitate visual browsing. The final topic of this section is the retrieval algorithm based on ContIndex.

3.1 Definition

Let us now give a formal definition to ContIndex:

Assume Σ is a set of multimedia objects, $\Omega = \{\omega_1, \omega_2, ..., \omega_m\}$ represents a set of m classes to which Σ is to be classified. Assume also that Ω satisfies that

1) $\omega_i \neq \Sigma$ *for all* $i = 1, 2, ..., m$;
2) $\cup_{1 \le i \le m}\ \omega_i = \Sigma$;
3) $\omega_i \neq \omega_j$ *for* $i \neq j$;

The indexing process consists of recursive application of mapping $\Sigma \to \Omega$ denoted by $\Gamma = \eta\ (D, \Omega)$, where D is a set of parameters to define the mapping, and classes in Ω represent the categories of multimedia object set Σ, and are associated with nodes of the index tree $\{N_1, N_2, ..., N_m\}$.

In ContIndex tree, number of classes *m* is kept the same for all intermediate nodes for manipulation efficiency. In this case, the index tree is an m-tree. The mapping Γ is defined by *D* and Ω. According to the definition, Ω is a set of classes representing the goal of the mapping. *D* is related to a set of feature measures used for the mapping. When the mapping is defined, *D* is represented by a set of *reference feature vectors*. For simplicity, only one feature is used to create a level of the index tree.

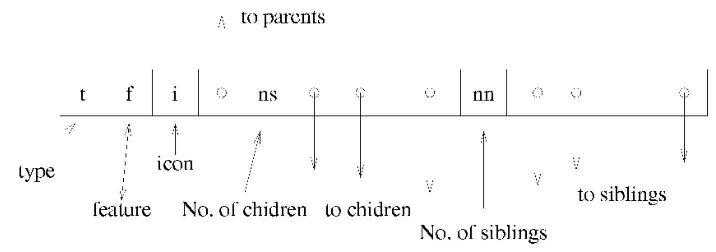
Fig. 2 shows the first three levels of a ContIndex tree. Features selected for creation of these three levels are: $F_{l0} = F^i$, $F_{l_1} = F^j$, and $F_{l_2} = F^k$. Nodes are labeled with a number of digits that is equal to their level number (the root is at level 0). For example, N_{21} is a node in second level, and is the first child of node N_2. N_{21}, N_{22}, ... are children of node N_2. They are similar with respect to feature $F_{l_0} = F^i$, inherit the reference feature vectors of feature F^i, and represent categories (ω_{21}, ω_{22}, ...) with respect to feature $F_{l_1} = F^j$. New reference feature vectors will be created for them upon the creation of these nodes.

A top-down algorithm for the creation of m-tree ContIndex is summarized as follows:

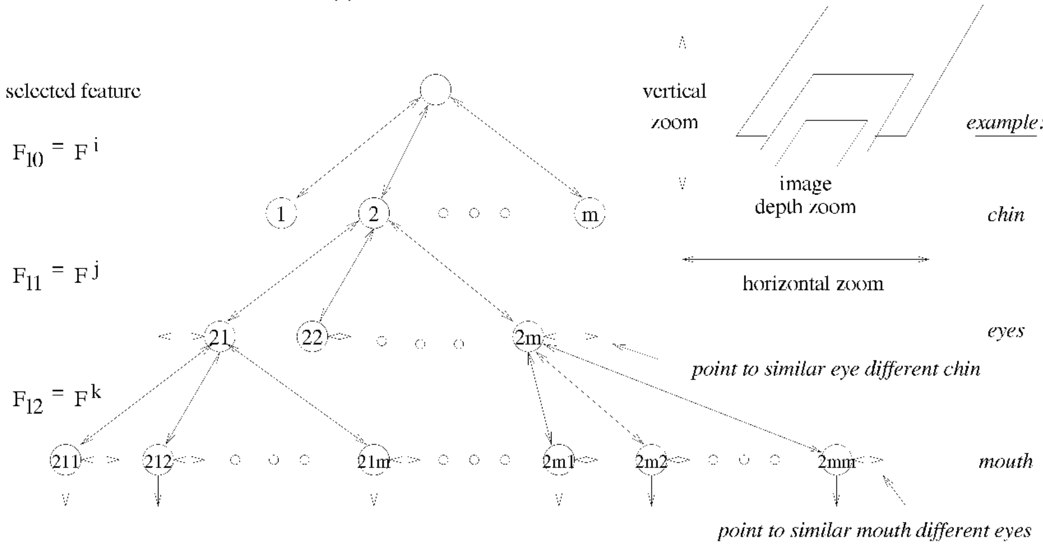
1) Attach all objects to root and start the indexing process from the root and down to leaf node level.
2) For each node at a level: Select a feature, partition the multimedia objects into *m* classes by using a set of feature measures, create a node for each class, and generate a reference feature vector(s) of the selected feature and an iconic image for each node.
3) Repeat the second step until each node has, at most, *m* descendants.
4) Start from second level, build horizontal links with respect to features which have been already used at the levels above.

3.2 Horizontal Links

When a user starts browsing the database using this index, he/she sees categories of the objects with respect to feature $F_{l_0} = F^i$. He/she then selects one node (assume N_2) and goes down one level. Now he/she arrives at the second level and gets a set of nodes N_{21}, N_{22}, ..., N_{2_m}. These nodes are associated with object categories with respect to feature $F_{l_1} = F^j$. These categories are numerically represented by reference feature vectors and visually represented by icon images. Since these nodes have the same reference feature vectors



(a) node structure of ContIndex tree



(b) ContIndex facilitates content-based retrieval and visual browsing. It is created by self-organized neural networks on multi-modal feature measures.

Fig. 2. The structure of content-based index ContIndex. As indicated in the figure, features selected for creation of these three levels of the index tree are: $F_{l_0} = F^i$, $F_{l_1} = F^i$, and $F_{l_2} = F^k$ Nodes are labeled with the number of digits which is equal to their level number. For example, N_{21} is a node in second level (the root is at level 0). It is the first child of node N_2.

for feature F_{l_0} and different reference feature vectors for feature F_{l_1}. Their icon images should appear different with respect to feature F_{l_1} and similar with respect to feature F_{l_0}. One may ask, can I view object categories with respect to feature $F^i(F_{l_0})$ under the same feature $F^j(F_{l_1})$ category using the same index tree? Of course, one possible way is to create another index tree with $F_{l_0} = F_j$, $F_{l_1} = F_i$, and switch among index trees whenever necessary. But, this would require too many index trees with every possible selection of features. It would be also very inconvenient to switch among index trees.

The horizontal zooming in ContIndex offers a satisfactory solution to this problem. Horizontal zooming is facilitated by horizontal links between nodes in the same level. Let us have a look at nodes in the second level. Nodes at this level under the same parent N_p, N_{p_1}, N_{p_2}, ..., N_{p_m}, represent categories with respect to feature F_{l_1} and under the same category with respect to feature F_{l_0}. Now suppose user finds N_{p_q} is preferable with respect to feature F_{l_1} and wants to have a look at categories of feature F_{l_0}, which are represented by nodes N_{1_q}, N_{2_q}, ..., N_{m_q}. To achieve that, we can simply create horizontal links among these nodes. The algorithm is as follows:

Horizontal link creation

- At level l:
  Node labeling convention: $N_{p_1 p_2 \cdots p_l}$
  for ($p_l = 1, p_l < m + 1; p_l + +$)
  - for feature F_{l_0}, find nodes $N_{1_{p_2 \cdots p_l}}, N_{2_{p_2 \cdots p_l}}, \ldots, N_{m_{p_2 \cdots p_l}}$, which represent categories with respect to feature F_{l_0} and are similar with respect to all other features. Make bidirectional horizontal links among them.
  - perform same operation for features $F_{l_1}, \ldots F_{l_{l-1}}$.
- Repeat the above labeling process for all levels.

3.3 A Practical Example

Fig. 3 is a practical example of ContIndex indexing tree created for a face image database system CAFIIR. In the face image database, each image is described by six facial features, namely, hair, eyebrows, eyes, nose, mouth, and chin. Feature measures (16 principal components and landmarks) are extracted from the image regions which cover these facial features. The index was created by using one facial feature at each level. Three levels are shown in the figure. At the level three, there are two types of horizontal links. These two types of horizontal links create two virtual indexing trees as shown in boxes.

3.4 Iconic Image Construction

According to the definition, an intermediate node of ContIndex tree is an abstract representation of all its child nodes. ContIndex is content-based, the content of this intermediate node should be a representative of the contents of its child nodes. That is to say that an image (referred to as icon image), a set of feature measure vectors, and interpretation of the content must be created upon creation of an intermediate node. As far as feature measures are concerned, feature measure vectors are computed as centroid

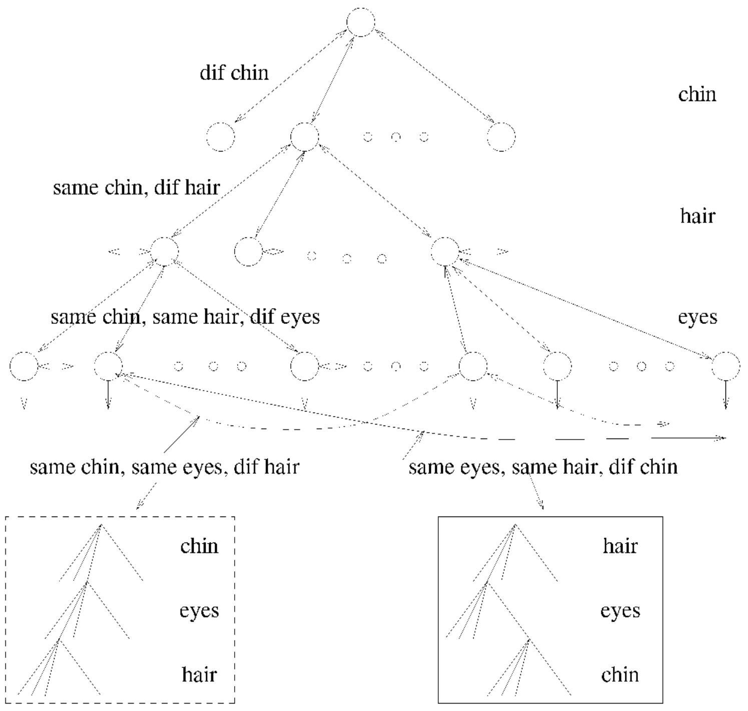

dif chin
chin
same chin, dif hair
hair
same chin, same hair, dif eyes
eyes
same chin, same eyes, dif hair
same eyes, same hair, dif chin
chin
eyes
hair
hair
eyes
chin

Fig. 3. ContIndex indexing tree and its horizontal links.

of its child nodes. The interpretation is then the name of the category this intermediate node represents.

There are several ways to generate an icon image for an intermediate node. One possibility is to average icon images of its child nodes. The icon image obtained this way may be blurred due to averaging. To avoid icon image blurring, we can choose one of its child node which is the nearest to the centroid, and use the icon image of this child node as the icon image of this intermediate node.

Principally, icon image of an intermediate node must show certain characteristics of the category this intermediate node represents. For this reason, in some applications line drawings are used as icon image to illustrate the categories.

3.5 Content-Based Retrieval Using ContIndex

One major type of content-based retrieval is that for a given query object find the best matches from a database. The query object can be given by using a sample object, or by descriptive or numerical specification. Suppose now that from a given query object, feature measures can be derived. The retrieval process is then to find the best matches using ContIndex.

Since the criterion used to search the best matches is the similarity measure given in (1). This similarity is the weighted summation of similarities on features of the objects. The weights are usually adjustable by users. Therefore, these weights cannot be built into index. ContIndex offers flexibility for the content-based retrieval. The retrieval process is a top-down classification process starting from the root of the tree. At each node, the process chooses from the child nodes one or more nodes which are the nearest to the query object with respect to the feature used for creation of this node in the index creation process. How many child nodes should be chosen depends on the weight for the feature. A higher weight implies that the feature is more critical. Less child nodes should be chosen. The retrieval algorithm is as follows. For explanation convenience, let us quote (1) with minor annotation modifications:

$$sim_l(O^q, O) = w_{l_0} sim_{l_0}(F^q_{l_0}, F_{l_0}) + w_{l_1} sim_{l_1}(F^q_{l_1}, F_{l_1}) + \dots \quad (2)$$
$$w_{l_0}, w_{l_1}, \dots = 0 \text{ for all } l_0, l_1, \dots > l$$

where subscripts l_0, l_1, … denote the levels of the tree. For example, F_{l_1} denotes the feature used in level l_1 for index creation, w_{l_1} is the weight for the feature F_{l_1} in the query. Weights are normalized so that the summation of them equals to 1. Since the retrieval process is top-down. The similarity measure $sim_l()$ stands for similarity at level l. It counts features which have been used at levels above l and level l. Therefore, w_{l_x} is set to zero if lx is below l. The procedure for content-based retrieval by ContIndex tree traversal is given as follows:

- Start the search from the root and examine all chosen nodes level by level.
- For all chosen nodes in a level, compute the similarity between the query object and child nodes with respect to the features used at that level and all levels above in index creation process using (3).
- Examine the total weights in the similarity computation. If it is small, more nodes will be chosen. Less nodes will be chosen otherwise.
- Repeat the second and third steps until leaf nodes are reached. The selected leaf nodes will be ranked according to the similarity and presented as the final retrieval result.

Insert and delete operation on ContIndex tree are much simpler than the retrieval operation described above.

4 ContIndex Creation by Self-Organization Neural Networks

Having discussed the procedure for index tree construction, we need to explain the mapping $\Gamma = \eta(D, \Omega)$ which is used to perform clustering in a self-organization manner based on multimodal feature measures. As discussed in previous sections, the mapping must be able to generate valid categories using multimodal feature measures, and nodes created must be spatially self-organized to facilitate visual browsing.

The first issue is crucial because multimedia objects, for example, images, have their visual features according which domain experts categorize them. To be consistent with domain expert, firstly, extracted feature measures should well capture the characteristics of these visual features, and secondly, the clustering algorithm should be able to generate desired categories based on these feature measures.

Visual features are usually complex, and cannot be represented by one simple feature measure. That is to say that multimodal feature measures are needed for most cases. For example, to classify hand-written characters, one needs to combine several feature measures (Fourier descriptor, critical points, grey image, etc.) to obtain high recognition accuracy [16], [19].

Notice also that feature measures may contain diverse visual information while the similarity used for index creation must have certain context and frame of reference. When we cluster images into categories using these feature measures, the result may not be as anticipated. It is because these feature measures may contain information on other aspects of the image than defined context and frame of references only. For example, Principal Component Analysis (PCA) has been widely used for feature extraction. PCA is defined on a set of images, say eye images, and represents the principal variations of this set of images. These principal variations do not necessarily fully coincide with a context and frame of reference in the index, say, the size of the eyes. Usually, the information of the context and frame of reference is contained in PCA, but is mixed with other information. Therefore, proper "feature selection" (here we use pattern recognition terminology) should be performed before using these feature measures to create index. To accomplish that, we use LEP (Learning based-on Experiences and Perspectives) neural network model to fuse m*ulti*modal feature measures and to self-organize index nodes.

In the following sections, we will first brief the structure of LEP neural network which perform the mapping, discuss multimodal feature measure fusion using LEP neural network, and then explain spatial self-organization using Kohonen's self-organization map. The bidirectional learning will be discussed in the last section.

4.1 LEP Neural Network Architecture

The structure of the LEP network model for ContIndex index creation is shown in Fig. 4. It consists of two macro-layers: feature selection layer (supervised learning) and self-organization layer (unsupervised learning). Feature measures (F_j^i, $j = 1, 2, ...$) are fed into their respective feed-forward networks. These networks have the same number of input units as the dimension values of the feature measures. The number of output units of these three networks are the same, which is the number of categories of the training data set. The outputs of the feature selection units of these networks contain information with respect to the desired categories contributed by these measures.

The self-organization macro-layer generates node parameters using the output from feature selection macro-layer. For $m - tree$ ContIndex, the number of output nodes is m. When the application requires certain categories appear in one level of the index tree, the user needs to prepare a training data set, and uses it to train the feature selection macro-layer for the initial set up. The number of classes in the training data set equals to the number of output units of the feature selection network. It can be different from m provided the training data set reflects the desired categories.

4.2 Fusion of Multimodal Feature Measures

The main idea of LEP neural network model can be summarized as follows [16]: Conceptually, properties of an object appear as perspectives. Perspectives of an object depends on viewers, view points, and means of observations. We call that as the first type of perspective. For example, weight, height, and photos of a person are first type perspectives of this person; to examine the heart of a patient, images and signals can be obtained by means of microwave resonance imaging, computer tomography, angiography, electrocardiograpy, etc. These images and signals are measures of the heart, and represent first type perspectives of the heart properties. The second type of perspective refers to the way to evaluate the data of the first type perspectives. For example, when measuring the similarity between heart problems based on these images and signals, similarity measures such as distances and correlations should be developed according to the domain experts' knowledge. Each of these similarity measures evaluates the data in a particular way and has a particular interpretation.

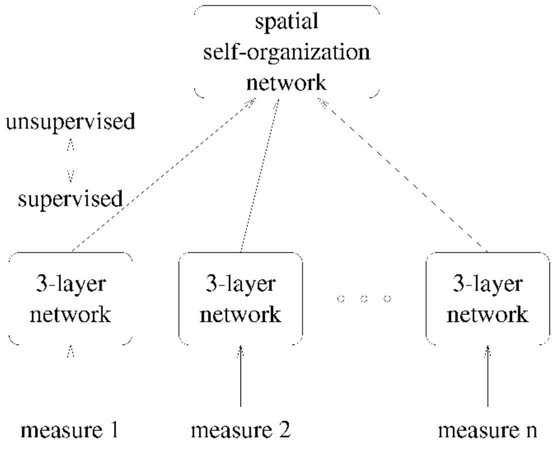


Fig. 4. LEP (Learning based on Experiences and Perspectives) neural network model for ContIndex index creation.

On other words, the first type perspectives of objects correspond to multimedia data and multimodal feature measures, and the second type perspectives address the importance of similarity measures when evaluating the feature measures. How to fuse multiple perspective data and what similarity to use are two major problems in ContIndex creation. Use of multiple similarity measures to improve self-organization will be discussed in the next section.

Multiple perspective data are multimodal. To fuse multimodal feature measures, feature selection is the first step. We use three layer feedforward networks for feature selection. From a view point of pattern recognition, feature selection is aimed to reduce the feature dimensionality, to save the implementation cost, and to avoid so-called peaking phenomenon. In the case of data fusion, it is aimed to focus to certain information which is most relevant to specified categories. Let us see PCA for eye images again. What PCA does is just to capture the principal variations in terms of grey scale changes of images. When it is applied to images of eyes, it does not necessarily just contain information used to generate specified eye categories such as eye size. Therefore, to filter out all irrelevant information, we need to perform feature selection on PCA coefficients of eye images. It can be considered as a type of "focus attention."

To demonstrate that three layer feedforward network is suitable to the feature selection for data fusion, let us formally define feature selection [22]. Assume that the feature measure is a n dimensional vector and is written as

$$x = [x_1, x_2, ..., x_n]^T \tag{3}$$

As defined in Section 3.1, each x may belong to one of m possible classes $\omega_1, ..., \omega_m$. It is further assumed that these data are generated by a random process and that the model of the process can be characterized by class-conditional density function $p(x \mid \omega_i)$ and a priori class probabilities $p(\omega_i)$. In the case of supervised feature selection here, the optimization is performed over all admissible mappings Θ, which "selectively" map the feature measure onto a new space $(y_1, y_2, ..., y_n)$.

$$J(\Theta_{optimal}) = \max_{\Theta} J\{\Theta(x)\} \tag{4}$$

Once Θ is determined, the selected feature vector $y = [y_1, ..., y_n]^T$ is then given by

$$\mathbf{y} = \Theta(\mathbf{x}) \tag{5}$$

The criterion function J for optimization here is defined as probability of misrecognition and given as

$$e = \int [1 - \max_{i} p(w_i|\mathbf{y})]^2 p(\mathbf{y}) d\mathbf{y} \tag{6}$$

Fortunately, the error criterion function coincides with the one defined in backpropagation learning for multilayer feedforward networks [14]. With backpropagation learning algorithm we can find the optimal mapping which is defined by the learned weights of the network.

It has been proven [18] that multilayered feedforward networks with as few as one hidden layer using arbitrary squashing functions (e.g., logistic, hyperbolic tangent) are capable of approximating any Borel measurable function from one finite dimensional space to another, to any desired degree of accuracy, provided sufficient hidden units are available. Generally speaking, the input-output mapping of such a network is analytic and can represent all continuous mappings. Therefore, we choose to use three layer networks in the feature selection layer.

4.3 Spatial Self-Organization

For visual browsing of databases, spatial organization of nodes are preferable. For example, to view types of eyes with respect to eye size, we prefer all icon images are displayed with the size from largest to smallest on the screen. For this purpose, Self-Organizing Map (SOM) by Kohonen [13] is an effective neural network paradigm for our ContIndex creation. On the other hand, we have to incorporate multiple perspective concept into SOM to guarantee reliable learning.

The second type perspective concepts is implemented here by combining several similarity measures. for example, in the competition among output units for self-organization, the winning unit c is selected based on both a correlation and minimum distance basis [16], [6]:

$$a_c = \max_i a_i$$
$$a_i = dis(\mathbf{x}, \mathbf{p}_i)/cr^k(\mathbf{x}, \mathbf{w}_i)$$
$$dis(\mathbf{x}, \mathbf{p}_i) = \left[\sum_j \left(x_j r_{ij} - p_{ij}\right)\right]^{1/2}$$
$$cr(\mathbf{x}, \mathbf{w}_i) = abs\left(\sum_j x_j w_{ij}\right) / \left[\sum_j x_j^2 \sum_j w_{ij}^2\right]^{1/2} \tag{7}$$

where a stands for activation of the output units in the self-organization network. $\mathbf{w}$, $\mathbf{p}$ are weight vector and template vector, respectively. r_{ij} defines the relative importance of the elements in a feature vector, and $dis()$, $cr()$ are distance and correlation functions, respectively. k is a parameter to adjust the effect of normalized correlation to the whole similarity measure. The detailed discussion of (7) and LEP neural network model can be found in [16, chapter 5].

The output units are arranged as a two-dimensional array in the spatial self-organization network in Fig. 4. The number of output units is m for m-tree index. Suppose there are K input units and L output units in the network. Each input unit is connected to every output unit with a certain synaptic weight $\{w_{k_l}, k = 1, 2, ..., K; l = 1, 2, ..., L\}$. For lth output unit, a template vector $\{p_{k_l}, i = 1, 2, ..., K\}$ and a weighting vector for input feature vector to define relative importance of their elements $\{r_{k_l}, k = 1, 2, ..., K\}$ are stored. They will be matched against input vectors during learning. Both link weights and template vectors are long-term memory items and are stored as two link weights from input units to output units.

Let $\mathbf{x} = (x_1, x_2, ..., x_K)^T$ be the K-dimensional real input feature vector presented to the input array at time t = 1, 2, 3, ... Then the output units begin to compete with each other. The winning unit c is selected based on both correlation and minimum distance basis defined in (7).

After a winning unit is selected, all the units within its neighborhood are updated. Let $N_c(t)$ be the neighborhood around unit c at time t. N_c is usually set very wide in the beginning to acquire a rough global order, and then shrink monotonically with time in order to improve the spatial resolution of the map. This procedure is crucial for the topological ordering. The weight vector updating formula is

$$\mathbf{p}_i(t+1) = \begin{cases} \mathbf{p}_i(t) + \alpha(t, d_i)[\mathbf{x} \cdot \mathbf{r}_i - \mathbf{p}_i(t)] + \alpha_c \beta(e_c) & \text{if unit } i \text{ is in } N_c(t) \\ \mathbf{p}_i(t) & \textit{otherwise} \end{cases} \tag{8}$$

where $\alpha(t, d_i)$ is an *adaptation gain* and decreases with time. It is a function of both time t and distance d_i, and usually has the shape of Gaussian function. As d_i becomes large the update becomes smaller.

a_c represents the confidence measure of the winning unit. If a_c is approximately equal to 1 the unit wins with high confidence therefore the template update can be large. $\beta(e_c)$ is a function of experience record e_c, which counts the number of times the unit has won. $\beta(e_c)$ is inversely proportional to the experience record e_c. If a unit has experienced a lot of learning the templates should not vary too much. On the other hand, experience records are attenuated by the so-called forgetting function, which is a simulation of human forgetting phenomena. Forgetting enhances the adaptability of the neural network. The forgetting function takes the form of exponential form $e_c e^{-\lambda^t}$.

Similarly, the update of correlation weights is defined as:

$$\mathbf{w}_i(t+1) = \begin{cases} \dfrac{\mathbf{w}_i(t) + \alpha(t, d_i) a_i \beta(e_c) \mathbf{x}(t)}{\left\|\mathbf{w}_i(t) + \alpha(t)\mathbf{x}(t)\right\|} & \textit{if } i \in N_c(t) \\ \mathbf{w}_i(t) & \textit{otherwise} \end{cases} \tag{9}$$

As a result of competitive learning with a dynamic neighborhood window, the weight vectors (templates) tend to approximate the probability density function of the input vectors in a spatially ordered fashion.

4.4 Bidirectional Learning on Experiences

Multimedia objects in the database represent event/object cases. ContIndex performs abstraction/generalization of these events/objects cases and produces content-based index. Intermediate nodes in the index tree represent categories of cases. They are generalization of cases, and cases are instances of these categories. If, for example, under a category there are similar patients a doctor has been cured, this category represents the experience of this doctor regarding this type of patients. In general, an intermediate node represents a certain concept, which is an abstraction of cases under it. To capture the validity of the concept, for each intermediate node, a record of confidence is maintained. The confidence record of a concept is high if the number of cases supporting it is large. The validity of these concepts are also subject to updating and changes of cases, and consequently must undergo feedforward learning (learning from instances).

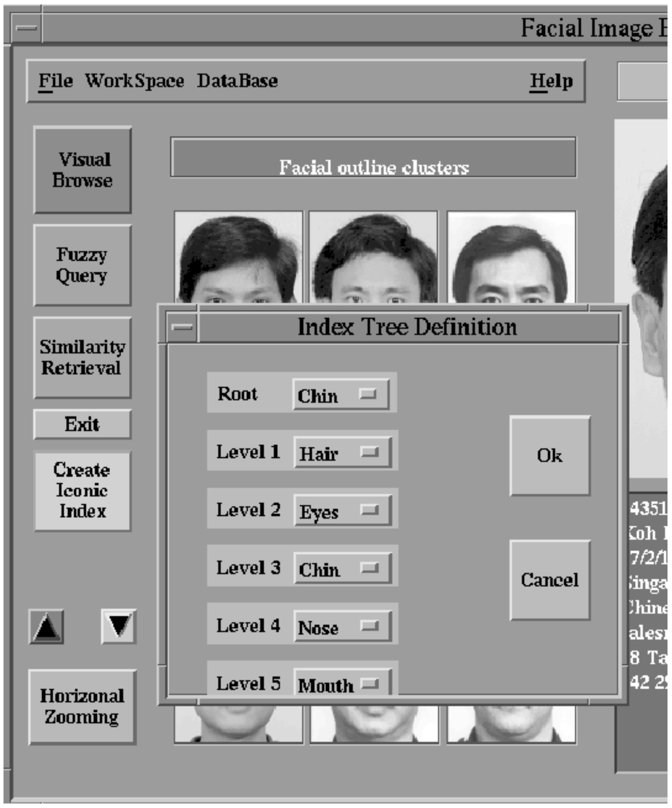
On the other hand, concepts must be verified by domain experts. When a domain expert makes use of a concept represented by an intermediate node or a particular event/object case at leaf node level of the index, this concept/case is meant to be successfully verified. Its confidence level should be raised. It is referred to as feedback learning (learning by commitment).

Real world data is spatially and temporally varying. High confidence records for today does not mean much for tomorrow. Therefore, we introduced *forgetting process* to update the confidence record. It also controls the degree of adaptability of the content-based index. The forgetting process can be adjusted by an attenuation factor $\gamma(T)$

$$\gamma(T) = e^{-\kappa T} \qquad (10)$$

where T is a temporal parameter. It can be the total number of input cases the network has processed since last forgetting process. κ is a small constant used to adjust the degree of forgetting. The forgetting processes are performed periodically.

After a forgetting process, the confidence records of some nodes in the index may become very small. This suggests rare use of those concepts, and they can be discarded from the index tree. The forgetting process further expands adaptability of the index: in the long term, as the environment changes, outdated concepts are erased, and new concepts are developed.



Root	Chin
Level 1	Hair
Level 2	Eyes
Level 3	Chin
Level 4	Nose
Level 5	Mouth

Fig. 5. Interface for creating indexes interactively. When the button "Create Iconic Index" is clicked, a push-down window "Index tree definition" appears. User can then select features for each level of the index. The creation process is activated by clicking at the button "Ok".

5 Experimental Results

ContIndex has been applied to index face images [6] and trademark images [7]. The experimental results here will be given and discussed using face images. A Computer-Aided Facial Image Inference and Retrieval (CAFIIR) system was developed primarily for criminal identification. Each face image was characterized using six features: chin, eyes, hair, eyebrow, nose, and mouth. Retrieval of similar face images is based on the similarity measure of all six features as defined in (1). For visual browsing purpose, we created 9 – *tree* ContIndex so that icon images can be displayed with 3×3 array.

The first prototype of CAFIIR was completed on July 1993, when there are only about 150 face images, that are donated by police officers and our colleagues and students. Now the system has been tested against 1,000 face images.

When creating an index, features used at levels are interactively selected, see Fig. 5. Several indexes can be created for one database. As an example, Fig. 3 shows an index created using chin, hair, eyes, etc. The horizontal links in this index brings much flexibility to the index. As shown in the figure, at third level, with the help of these two horizontal links, the index can provide virtual views of other two indexes each of them has different order of these three features for these three levels.

Let us have a look at the creation process of the first level of the index in Fig. 3 by using LEP neural networks. For the sake of explanation, we need to say a few words about *system configuration*. By system configuration, we mean to set system parameters so that the system can work properly for an application site. In the case of a face retrieval system, assume that the system is installed in a city's police headquarters for criminal identification. There, the police has their own collection of face images, and their own way of categorizing face features. To obtain proper system parameters for face image categories, a set of sample images must be collected and used to train the LEP networks in the system. This is the *configuration phase* of the system. After setting of system parameters, the system is in its *operating phase* and ready to operate.

In CAFIIR, face images are categorized according to six features. Therefore, six sets of parameters must be learned for the LEP network in the configuration phase. These parameter sets will be used to create levels of the index tree. To create the first level of the index tree in Fig. 3, parameters for chin categories must be invoked.

In our experimental study, the collection of face images are donated by our colleagues and students. Nine chin types were identified in the training data set. They are: pointed, rounded, tapered, squared, bony cheek, short-chin, long-chin, jowls [17]. For computer categorization of face images, landmark points are semi-automatically identified, the face images are then normalized, a "U" shape chin image is extracted for each face image. Principal component analysis (PCA) is then performed on these chin images, and the first 16 components are used as chin feature measures. In the experimental study, landmark coordinates around chin are also considered as one measure.

According to the concept in Section 4.2, we have two feature vectors, PCA, and landmarks, which are of different modals, and are results by looking at the chin from two first type perspectives. The detailed architecture of the LEP network is illustrated in Fig. 6. The uppermost layer is a self-organization layer and the other four layers are supervised learning network used to fuse these two feature vectors. There are two separate three-layer feedforward networks for PCA and landmarks, respectively. The fourth layer is used to combine these two-feature measures. The network units in these four layer are: (16, 10); (9, 9); (9, 9); and 9. That is: the PCA network has 16 units in the first layer and 9 units in the second and third layer; landmark network has 10 units (five landmark points) in the first layer and 9 units in the second and third layer. The fourth layer acts as both the output layer for supervised learning network and the input layer for self-organization network. The configuration process is aimed to learn all weights in this four layer network, which will be used to calculate the input to the self-organization network. On other hand, parameters of self-organization network, such as sample vectors of output units, will be learned during indexing process, and stored as node parameters of index tree.

To learn the parameters of supervised learning network, the first three layers are trained independently using PCA and landmark training data sets. In this training process, the weights from the first layer to the second layer, and from the second layer to the third layer are learned. Then, both PCA and landmark data sets are applied simultaneously to train the weights from the third layer to the fourth layer. When the training is completed, the weights are saved into a file. These weights will be loaded into the network when an index creation process is stimulated.

Fig. 7 is a screen dump of a visual browsing. It is at the second level of the index. It can be seen from the panel for "index hierarchy" that the second level is highlighted and that "hair" was used at this level. At the first level, icon image with long chin was selected. This can be also seen from the connection of the first level and the second level in the index tree graph. Nine icon images on the left side display panel represent categories of facial images with respect to hair thickness are shown. The facial hair thickness is arranged in a descending order from bottom-left to top-right. Note also that all these nine icon images have long chin. This property is inherited from their parent node.

At the second level, if we select an icon image and then click "horizontal zooming" button, images with similar hair and different chin will appear on the display buttons. The horizontal links enable the flexibility of the indexes. The user can browse through the database freely without the limitation of the initial order of features selected to create the index.

The efficiency of ContIndex is similar to conventional index trees since they have the same tree structure. That is, for an L level of $m - tree$, it reduces the number of search operations from m^L to mL. On the other hand, ContIndex support similarity retrieval, to avoid missing of possible similar candidates when the given sample is at the boundaries among tree branches, more than one (k) candidates will be selected for further examination. The total number of search operations in a retrieval will be $m + mk(L - 1)$. For a face image database of 6×10^5, a $9 - tree$ ContIndex has six levels. Assuming that, at each level, four candidates are selected, then one retrieval needs 189 matching operations. The searching time is reduced by a factor of 3,175. Since the number of candidates to be selected at each level depends on the total weights at that level, the searching time will slightly depend on the weight assignment. In practice, one can choose to create several indexes with different sequences of features. To reduce the searching time, the

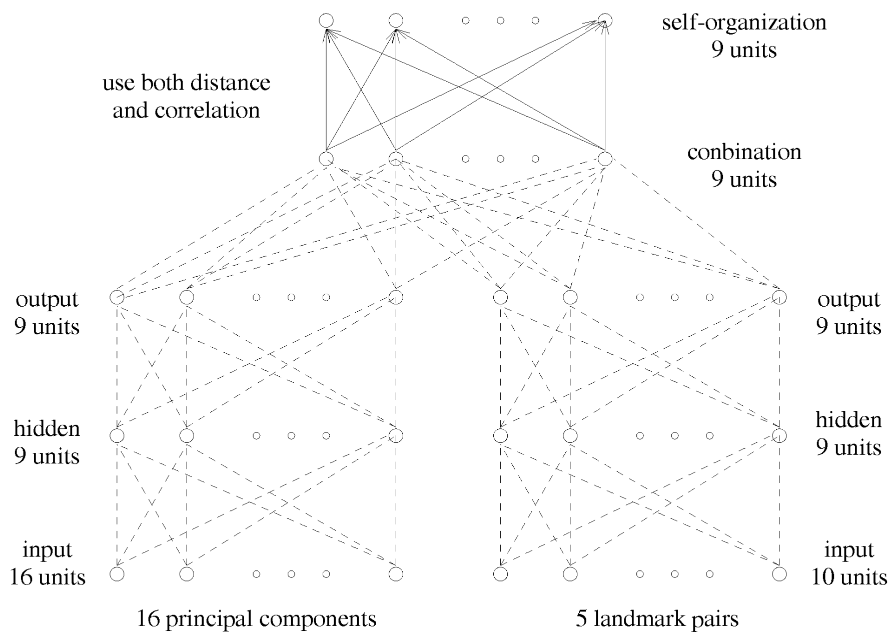
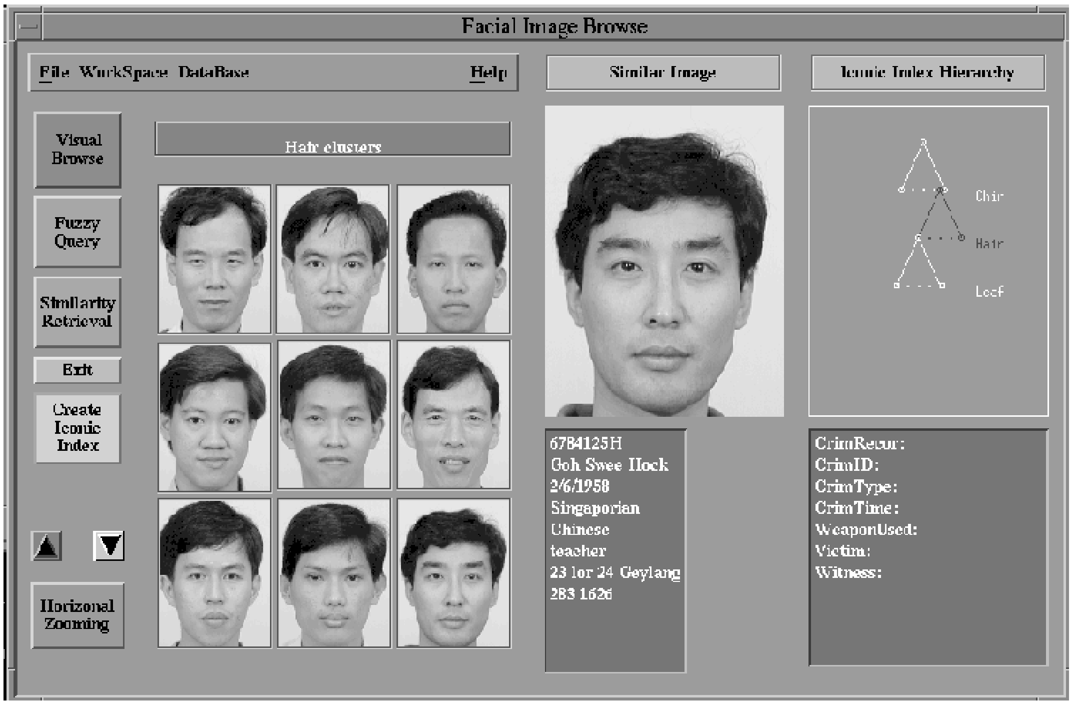


Fig. 6. Network architecture to create ContIndex in CAFIIR. All weights indicated by dashed lines must be learned in the configuration phase.



6784125H
Goh Swee Hock
2/6/1958
Singaporian
Chinese
teacher
23 lor 24 Geylang
283 1626

CrimRecur:
CrimID:
CrimType:
CrimTime:
WeaponUsed:
Victim:
Witness:

Fig. 7. Browsing the facial image database. It is now at the second level of ContIndex tree. As shown by the icon image in the display window, face image categories are spatially arranged according to the hair feature with respect to thickness: from bottom-left to top-right the hair thickness is in a descending order. The user can select any category by clicking at the icon image and then downward arrow to go down one level. Horizontal zooming is carried out by selection of icon image and clicking at the horizontal zooming button.

searching algorithm can choose the index with the feature at the first level which has the maximum weight assigned.

ContIndex will not show much improvement for small databases, for example, database of 100 images. In this case, searching the two level tree needs about 45 operations, which is the same order as 100.

Principally, ContIndex has similar dynamic property to conventional index trees. At the time the ContIndex tree is created, the tree is well balanced because of the neural network's self-organization property. After many insertions and deletions, the tree may tend to be unbalanced. Therefore, after certain period, the indexes need to be rebuilt.

One may ask, can we use elements of feature measures as attributes, and then use indexing facilities provided by ordinary database systems such as B-tree to index these images? Let us not talk about categories and visual browsing which B-tree cannot offer, just think about how many B-tree indexes we must create for these six facial features, each of them are characterized by 16-dimension feature vectors in our experiment. With so many indexes, the retrieval cannot be efficient and balancing among indexes with respect to given weights are also very tedious.

Since the iconic and horizontal rooming of the ContIndex cannot be evaluated quantitatively, we have been purposely collecting feedback from whose we have demonstrated the system. During past three years, the system has been demonstrated to more than 100 times to groups of visitors which are from academic institutes, industrial sectors, or governments. The iconic browsing has been able to attract their attention and to provide useful and intuitive means for browsing the database. On the other hand, horizontal browsing needs certain effort of explanation (usually less than 30 minutes) for a novice to grasp it. They like it as soon as they know how to use it.

As far as retrieval accuracy concerned, it is sole dependent on the feature measures extracted, and similarity function used in the retrieval. The indexing method presented here does not affect the retrieval accuracy. Evaluation of content-based retrieval is a critical issue in developing a good content-based retrieval engine. We have studied the problem and proposed a method [23].

6 Conclusion and Remarks

The content-based indexing ContIndex presented in this article has some salient features. It can handle multiple features of multimedia objects while offering retrieval flexibility. Using self-organization neural networks, ContIndex can create index by fusing multimodal feature measures with desired context and frame of references. More work is needed to improve its dynamic property.

Acknowledgment

This research was sponsored by Real World Computing Partnership.

REFERENCES

[1] J. Bach, S. Paul, and R. Jain, “A Visual Information Management System for the Interactive Retrieval of Faces,” *IEEE Trans. Knowledge and Data Eng.*, vol. 5, pp. 619-628, 1993.

[2] W.I. Grosky, “Multimedia Information Systems,” *IEEE Multimedia*, pp. 12-24, Spring 1994.

[3] A. Pentland, R.W. Picard, and S. Sclaroff, “Photobook: Tools for Content Based Manipulation of Image Databases,” *Proc. SPIE Conf. Storage and Retrieval of Image and Video Databases*, vol. 2, no. 2,185, Feb. 1994.

[4] E.G.M. Petrakis and S.C. Orphanoudakis, “Methodology for the Representation, Indexing and Retrieval of Images by Content,” *Image and Vision Computing*, vol. 11, pp. 504-520, 1993.

[5] J.K. Wu, A.D. Narasimhalu, B.M. Mehtre, C.P. Lam, and Y.J. Gao, “CORE: A Content-Based Retrieval Engine for Multimedia Databases,” *ACM Multimedia Systems*, vol. 3, pp. 3-25, 1995.

[6] J.K. Wu, Y.H. Ang, C.P. Lam, H.H. Loh, and A.D. Narasimhalu, “Inference and Retrieval of Facial Images,” *ACM Multimedia J.*, vol. 2, no. 1, pp. 1-14, 1994.

[7] J.K. Wu, C.P. Lam, B.M. Mehtre, Y.J. Gao, and A. D. Narasimhalu, “Content-Based Retrieval For Trademark Registration,“ *Multimedia Tools and Applications*, vol. 3, no. 3, pp. 245-267, 1996.

[8] C. Faloutsos, M. Flickner, W. Niblack, D. Petkovic, W. Equitz, and R. Barber, “Efficient and Effective Querying by Image Content,” Technical Report, IBM Research Division, Almaden Research Center, RJ 9453 (83074), Aug. 1993.

[9] S.K. Chang, C. Yan, D.C. Dimitroff, and T. Arndt, “An Intelligent Image Database System,” *IEEE Trans. Software Eng.* vol. 14, pp. 681-688, 1988.

[10] W.I. Grosky and R. Mehrota, “Index-Based Object Recognition in Pictorial Data Management,” *CVGIP*, vol. 52, pp. 416-436, 1990.

[11] S.R. Griffiths, R.J. Wilkins, P.H. Lewis, H.C. Davis, and W. Hall, Media-Based Navigation with Generic Links, *Proc. ACM Hypertext ’96*, Washington, D.C., pp. 215-223, Mar. 1996.

[12] K. Hirata, Media-Based Navigation for Hypermedia Systems, *ACM Hypertext ’93*, pp. 159-173, 1993.

[13] T. Kohonen, “The Self-Organizing Map,” *Proc. IEEE*, vol. 78, pp. 1,464-1,480, 1990.

[14] J.L. McClelland and D.E. Rumelhart, *Explorations in Parallel Distributed Processing*. Cambridge, Mass.: MIT Press, 1986.

[15] A. Tversky, “Features of Similarity,” *Psychological Rev.*, vol. 84, pp. 327-352, 1977.

[16] J.-K. Wu, *Neural Networks and Simulation*. New York: Marcel Dekker, 1994.

[17] Y.H. Ang, “Enhanced Primal Sketch Face Recognition,” *Looking at People: Recognition and Interpretation of Human Action*, A. Pentland, ed., *Proc 13th Int’l Joint Conf. Artificial Intelligence*, Chambery, Savoie, France, 1994.

[18] K. Hornk, M. Stinchocombe, and H. White, “Multi-Layer Feedforward Networks are Universal Approximators,” *Neural Networks*, vol. 2, pp. 359-366, 1989.

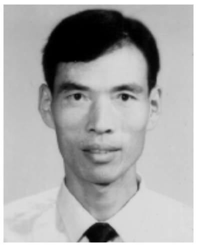
[19] C.Y. Suen, C. Nadal, R. Legault, T.A. Mai, and L. Lam, “Computer Recognition of Unconstrained Handwritten Numerals,” *Proc. IEEE*, vol. 80, pp. 1,162-1,180, 1992.

[20] J.-K. Wu, F. Gao, and P. Yang, “Model-Based 3D Object Recognition,” *Proc. Second Int’l Conf. Automation, Robotics, and Computer Vision*, Singapore, Sept. 1992.

[21] A.L. Yuile, “Deformable Templates for Face Recognition,” *J. Cognitive Neuroscience*, vol. 3, pp. 59-70, 1991.

[22] P.A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*. Englewood Cliffs, N.J.: Prentice Hall, 1982.

[23] J.K. Wu, C.P. Lam, and G. Senthilkumar, “Evaluation of Feature Measures and Similarity Measures for Content-Based Retrieval,” *Proc. SPIE Symp. Digital Image Storage and Archiving Systems*, vol. 2,606, Philadelphia, Oct. 1995.



Jian-Kang Wu received his bachelors degree from the University of Science and Technology of China, and his PhD degree from the University of Tokyo. He is now with the Institute of Systems Science (ISS) at the National University of Singapore. Prior to joining ISS, he worked at the University of Science and Technology of China as a full professor. He also worked as an Alexander-Humboldt research fellow at the University of Erlangen-Nunberg in 1989; as a visiting associate professor at the University of Pittsburgh in 1986; as a foreign research fellow at the University of Tokyo in 1982; and as an academic visitor at the Imperial College of Science and Technology, London, in 1980. Dr. Wu is now heading a group doing research on active multimedia and spatial information. His research interests include multimedia and spatial data modeling; flexible storage, retrieval and intelligent utilization of multimedia and spatial data; computer vision; and neural networks. He is the author of four books and more than 50 journal papers.