Content-based Multimedia Data Management and Efficient Remote Access

Jim GRIFFIOEN, Brent SEALES, Raj YAVATKAR and Kevin S. KIERNAN

Abstract. The recent availability of low-cost powerful workstations and PCs has caused a staggering increase in the quantity and quality of digital data, which includes still images, video clips, and sound. The problem of managing such a large volume of information has become very important for researchers who need efficient and simple access to the data. In this paper we present a framework for managing the storage, analysis, and access of multimedia data. We address the problems of user access and data management via an extensible graphical user interface which is the front-end to a semantic object-oriented multimedia management tool. We present a new method for efficient content-based searching of image and video data, and we examine several proposed solutions to the problem of Internet access to large multimedia databases.

Résumé. L'apparition récente de stations de travail et de PC puissants à bon marché a entraîné une augmentation prodigieuse de la quantité et de la qualité des données sur support informatique, comprenant entre autres des images fixes ou des séquences sonores ou vidéo. Le problème de la gestion de tels volumes d'information est devenu particulièrement important pour les chercheurs, qui ont besoin d'un accès simple et efficace à ces données. Cet article propose un modèle de travail permettant de gérer la conservation, l'analyse et l'accessibilité des données multimédia. Nous discutons les problème de l'accès utilisateur et de la gestion des données via une interface graphique extensible, qui elle-même repose sur un outil de gestion multimédia sémantique et orienté objet. Nous proposons une nouvelle méthode pour une recherche de contenu efficace sur des images et des données video, et nous examinons diverses solutions proposées au problème de l'accès via Internet à de larges bases de données multimédia.

Keywords: Multimedia databases, sound, image, video clips, acces of multimedia data, multimedia databases management.

Mots-clés : Bases de données multimédia, image, son, vidéo, accès aux données multimédia, gestion de bases de données.

✉ Jim GRIFFIOEN, Brent SEALES, Raj YAVATKAR; University of Kentucky; College of Engineering; Department of Computer Science; 773, Anderson Hall; Lexington, Kentucky 40506-0046 (USA).
E-mail: griff@dcs.uky.edu, seales@dcs.uky.edu, raj@dcs.uky.edu

✉ Kevin S. KIERNAN; Department of English; University of Kentucky, Lexington, KY 40506 (USA).
Fax: +1 606 323 1072 E-mail: kiernan@pop.uky.edu or kiernan@beowulf.engl.uky.edu

1. Introduction

The Internet has seen explosive growth during the last few years and continues to grow at a staggering rate, adding new users from every imaginable profession each day. The primary reason for the enormous popularity of the Internet is the access it provides to vast and up-to-date repositories of information ranging from television footage to research publications. However, the large number of users and amount of information involved has prompted the need for development of utilities and software tools (termed *middleware*) that facilitate organization of repositories for easy and efficient access (searching, browsing, and retrieval).

More recently, applications and services such as web browsers, information brokers, white page servers, and many others have attempted to organize the masses of information and provide efficient access to the information space. Web crawlers and search engines continuously index the web to help user's locate documents related to their topic of interest. Although these accomplishments represent a significant advance, the rapid proliferation of new information media, particularly multimedia information such as images, video, and sound, will require new or enhanced middleware services to handle these emerging non-textual data formats effectively.

We are exploring new methods for enhancing middleware services to support effective and efficient access to multimedia information. Specifically, we are developing techniques that allow Internet users to access the information embedded within multimedia documents. The system allows users to manipulate, enhance, and annotate existing information and share these interpretations and modification with other users. Our design extends middleware services to support tools for multimedia data modeling and organization, embedded information extraction (semantic tagging), and efficient network transfer policies such as cooperative proxy server caching, automatic data prefetching, and hierarchical data transfers.

The following sections provide a high-level description of the problems to be solved and the goals we have for a multimedia modeling system. We also provide example scenarios to illustrate where such technology is applicable and useful. The remainder of the paper provides a detailed technical discussion of the issues facing multimedia systems and our proposed solution to the issues. In particular, we present a framework for searching and manipulating multimedia data, efficient content analysis techniques for multimedia data, and efficient network transport protocols and algorithms for accessing remote multimedia databases.

1.1. Multimedia Data Modeling Tools

Multimedia data contains an enormous amount of information. This information is in the form of identifiable "features" in the multimedia data. For example, a picture of a residential neighborhood may contain houses, trees, streets, sidewalks, cars, boys, girls, or other identifiable features. The content is exactly the information users want to search for and retrieve. Video data contains similar features but also contains timing data that can be used to track the movement of an object from frame to frame or to identify transitions between scenes. Similarly, audio data contains certain identifiable features such as words, sounds, pitches, and silent periods as well as timing information.

In addition to basic content information, multimedia data contains an infinite amount of *derived* information. Derived information refers to information that can be construed from the feature information. For example, having identified in a comprehensive database all legible letter-forms of an ancient manuscript, it becomes possible to use that information without previous encoding to identify illegible letter-forms elsewhere, to reconstruct damaged text, or to search other digitized manuscripts for similar or identical handwriting. Given a video sequence from a news broadcast, identification of a man walking on the moon might indicate that it is footage from the first moon walk. In both examples, the derived information was not readily available in any of the basic content but instead it is derived or concluded from the identified components. In short, derived information is the set of facts that can be concluded when a user views all the features of an image as a whole. Because derived information depends on a user's interpretation of the features, each user can contribute their own (possibly conflicting) derived information. As a result, a single multimedia data item can potentially contain an infinite amount of derived data.

Users of multimedia data are interested in both the data's basic content and its derived information. Users want to query and manipulate the content and derived information just like they would standard textual data. In order to index and search multimedia, the system must first identify the content and derived information. Given the enormous and rapidly growing quantity of multimedia data, manual techniques that require human labeling of data are unacceptable. Instead, automated methods that extract embedded and derived information are needed. Automatic extraction typically involves a sequence of computer processing steps to identify the desired features. For example, identifying

the notes in a music manuscript require a series of image processing operations that first clarify the image (enhancement), then locate the connected regions in the image (segmentation), then classify the segments (shape detection), and so on until the notes have been identified and labeled. The goal of our content based multimedia data management system is to provide a framework in which users can define automated methods to identify the desired content. Once automated methods have been defined, data is processed automatically and entered into a database that can be queried using conventional search techniques. Finally, the integration of a processing engine with a database allows users to take the results of a query and apply interactive processing to analyze more fully or to manipulate the data under study.

We have developed a tool to manipulate interactively multimedia data, define user-specific views and annotations, and search the identified content using the semantic modeling methods we originated in the MOODS system (Griffioen *et al.*: 1993; Griffioen *et al.*: 1995a, 1995b). The system allows users to extract automatically or interactively the desired information from an image and assign user-dependent information to an image. The extracted and assigned information is then entered back into a database which allows users to search more efficiently and query the multimedia information.

Entering the content and derived data into the database and associating it with the original information gives other users automatic access to the conclusions, thoughts, and annotations of previous users of the information. The ability to modify, adjust, enhance, or add to the global set of information and then share that information with others is a powerful and important service. This type of service requires cooperation between the multimedia data manipulation tools described above and the information repositories scattered across the network. Generated or extracted information must be deposited and linked with existing information so that future users will not only benefit from the original information but also from the careful analysis and insight of previous users.

We have used our design to develop a simple prototype system that aids paleographers in analyzing ancient manuscripts. Although still an early prototype, the system allows a user to search for certain letter forms in a document and then pop up a window containing indicators where letters were identified. Additional image processing and enhancement can then be applied if desired.

1.2. Embedded Information Extraction Techniques

Conventional content extraction techniques typically involve manually labeling the data. For example, video footage is frequently annotated by hand for later searches and retrievals. We have developed a new efficient method for automatic extraction of semantic information from images and video. Our approach works on standard compression schemes such as JPEG, MJPEG and MPEG that are commonly used to capture, store, and transfer (via a network) video and images.

Our embedded information identification method extends the eigenspace approach which was originally proposed in a different context (Murase and Nayar: 1995; Pentland *et al.*: 1994). We show how the eigenspace approach can be modified to work on compressed image and video streams in their compressed format. The result is that we can search images and video very quickly for visual objects like shapes, faces and textures. The objects that are found can be entered back into the database automatically using tools based on MOODS, coupling this powerful search technique with a highly organized and flexible data management framework.

1.3. Enhancing Internet Middleware

Given the reality of a world-wide community of users, a content-based multimedia system must address the problems of multimedia access, search, retrieval, and manipulation in a global information environment such as the Internet. That is, we must integrate multimedia data manipulation services into the existing Internet data repositories and their access methods. In particular, we are integrating the multimedia modeling methods of MOODS and the data extraction techniques we have developed for images and video into the World Wide Web hypertext language (HTML) and the web's data access methods provided by the HTTP and FTP communication protocols.

To support efficient on-line interactive access to digitized information, we are developing enhancements to the HTTP/HTML interface that support multiple data formats with specific quality of service guarantees. For example, the system will automatically adjust an image, video, or audio clip, based on the performance of the user's current connection to the Internet. If a user attempted to access a color image from the WEB over a slow modem connection, the system might automatically convert to black and white or reduce the image's size to reduce the transmission time.

To further enhance on-line interactive access, we developed automated prefetching techniques that would "guess" the data a user will need and retrieve it before the user requests the data. Finally, we developed methods of parallel data retrieval and cooperative caching to reduce further Internet access times.

The remainder of this paper examines the technical details involved in the design of the type of multimedia system outlined above. It describes the issues involved in developing a multimedia system that achieves the goals defined above and presents our proposed solutions to these issues. Section 2 elaborates on the object-oriented framework for modeling multimedia data called MOODS. Section 3 describes one of our new techniques for efficiently identifying objects in images and video clips. Section 4 describes techniques to improve network access to large digital databases. Finally, Section 5 concludes with a summary of our contributions.

2. The MOODS Multimedia Management System

The first goal of our research was to develop a highly flexible information management system capable of providing full access to multimedia content. The system must support both automatic and manual extraction of embedded information, support for multicomponent multimedia information, user and application specific views of the embedded information, and support for evolving views of the information. As a result, we have developed the MOODS system.

The MOODS system takes an innovative approach to the design of information management systems. Unlike conventional database systems which only support alpha-numeric data or limited access to visual data (e.g., simple storage, retrieval, and display), MOODS provides full access to all the information contained in multimedia data and treats multimedia data like a firstclass object. MOODS combines a database management system with an interactive multimedia data processing system that allows users to access and manipulate multimedia content. The resulting semantic information can then be queried and searched using conventional database primitives. Not only does the resulting system support interactive queries and retrievals, but it also provides the functionality required to extract dynamically new or additional information, the functionality to reinterpret existing information, and functionality to define new data interpretation and models.

2.1. System Components

The MOODS system allows application writers quickly and easily to write new domain-specific multimedia database systems with content-based search capabilities. Users of a database constructed via the MOODS framework can search the data's content or define user-specific interpretations of the content.

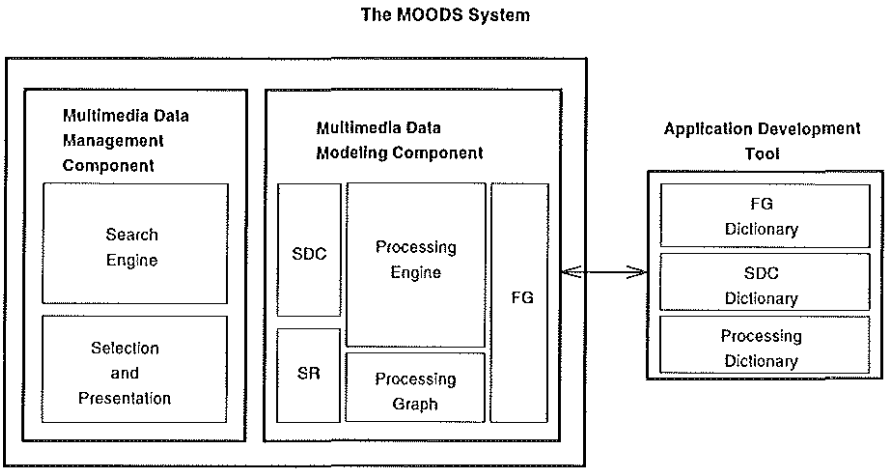

Fig. 1.– The components of the MOODS multimedia information management system

Figure 1 illustrates the various components of the MOODS system. The system consists of two main components: a *database* component and a *data modeling* component. In addition, an *application development tool* aides in the construction of modeling systems and facilitates reuse of functions an modeling systems.

The *data modeling* system distinguishes MOODS from conventional information management systems by providing the capability to extract content information from stored data. The ability to process data dynamically is important for several reasons. First, the processing engine provides each user and application domain with the ability to extract dynamically the information that is of importance to the user or domain. Second, a dynamic processing engine allows views (*i.e.*, the meaning of the content) to be changed or to evolve as new processing techniques become available. Third, dynamic processing reduces the data storage requirements without limiting the amount of information that can be accessed. Recall that the amount of embedded information is limitless.

Static processing schemes severely limit the potential information that can be accessed. This is one of the primary problems plaguing conventional multimedia databases that statically link keys to embedded information. Under MOODS, the system extracts the desired information on demand.

The data modeling system consists of five components. A set of *function groups (FG)* defines the logical operations that can be applied to the data. The types of semantic information known to the system are represented by a set of *semantic data classes (SDC)*, and a set of *semantic relationship (SR)* definitions describe the logical relationship between the various semantic data classes. The *processing graph* defines the transformations that a data item may legally undergo over its lifetime. Finally a runtime *processing engine* allows users of the system to apply interactively functions to extract and identify embedded information subject to the constraints imposed by the five components described above.

The *database* component is tightly integrated with the modeling system. After the modeling system has extracted the content information, it is automatically entered into the database and becomes available to the database for search or browsing via conventional database techniques. The transfer of information from the modeling system to the database means that any new information identified (or reinterpreted) by the modeling system will be made available to the user(s). Using the SR definitions, the database also supports queries on data that has not been processed or has not been processed to the level required by the query. For example, a query for a "Chevrolet Camaro" may be satisfied by locating all data items that contain a car and analyzing them further to determine the car's model. In this case, the database dynamically invokes the services of the modeling system.

The *application development* system is an auxiliary tool that aids in the development of new modeling systems. In particular, it maintains a complete database of all known (previously defined) function groups, semantic classes, semantic relationship models, and processing graphs. The tool allows users to construct new data models by defining their own semantic classes, relationships, and graphs, or by incorporating and combining existing structures. Data model reuse is fostered by allowing newly developed data models to be added to the tool's database for use in future models. The system also allows users to quickly modify or convert an existing model to meet their personal needs or the needs of their application domain.

2.2. The Runtime System

To utilize the information present in multimedia data, the data must first be processed (automatically or manually) to obtain its meaningful components and the information it conveys. We refer to these basic components as *structural objects*. Example structural objects in an image might be regions of uniform intensity, an edge boundary, curved segments, or regions of uniform texture. An audio stream might contain regions consisting of a single pitch or frequency or regions containing bursts of sound. These structural objects contain no semantic information other than basic structural features. However, given an application domain (e.g., medical imaging, cartography, or meteorology), additional semantic information can be associated with a structural object or set of structural objects. Each structural object corresponds to some object or semantic category in the current application domain. We call the domain-dependent categories *domain objects*. Each application domain typically has a reasonably well-defined set of domain objects (e.g., ventrical or tumor in medical imaging) that provide useful domain-dependent information about the structural object. It is precisely this *structural object* $\leftrightarrow$ *domain object* mapping that must be established during the embedded information extraction process.

To illustrate the MOODS runtime operation, consider a meteorology system. Initially an (expert) meteorologist would use the application development tool to construct a multimedia data model that can extract weather information from satellite photos, heat maps, etc. This involves defining the image processing transformations and image processing functions that can be applied to extract and assign semantic meaning to the embedded information in the images.

Having defined the set of legal transformations a satellite photograph may undergo to extract the semantic meaning, the meteorologist must then define data-specific transformations that will later be used to enter data into the database. For example, the meteorologist may define one transformation process for satellite images taken at close range of smaller regions (e.g., the state of Kentucky) and another transformation process for satellite images of larger regions (e.g., the continental United States). Once defined, new images will be automatically processed according to the appropriate transformation model and the resulting semantic information will be entered into the database.

A different meteorologist may then search the database for images containing storm activity. Using the extracted semantic information,

the system will return a list of images containing stormy activity. The meteorologist may then decide to invoke the processing engine on one of the returned images to further identify the intensity of a particular storm in the image. This information will then be entered back into the database and will respond to searches asking for storms with a particular intensity.

2.3. The Data Modeling Engine

The MOODS data modeling component supports an object-oriented model for manipulating multimedia data. To extract the content information from a multimedia object, a user manually or automatically pushes the multimedia object through a series of transformations to identify the important semantic information in the object. At any given point in the transformation process, a meaning, or interpretation of the data, is attached to the object. This semantic interpretation is stored in the database. As a result, database queries can be performed over the semantic interpretations using conventional search and query strategies. To support semantic objects that can be manually or automatically manipulated by the user, the MOODS system provides an enhanced object-oriented programming environment. The following paragraphs give a brief technical overview of some of the salient features of the MOODS system:

Dynamic Data Semantics: The semantics associated with the data in an object will typically change often over the object's lifetime. Therefore, it is important to change dynamically the set of functions (operations) associated with an object after it is instantiated. That is, the set of functions associated with an object depends on the object's current semantic meaning. As the object is processed its semantic meaning changes resulting in a new set of functions that can be applied. For example, a satellite photograph of urban city may initially have only have the semantic meaning "an image" and applicable functions "enhance", "segmentation" and "edge-detect". However, after executing an edge-detection function, the semantic meaning may have changed to "an image of edges" and now has available new functions to identify edges that represent roads and functions that identify other edges as buildings. As the semantics of the image evolve, the set of functions available to the object become more specific and powerful. Also, functions that no longer make sense may be removed. For example, segmentation on an "edge-image" makes no sense and would be removed.

Abstract Function Types: Given an image, one usually has a wide range of functions available that can perform a particular image processing operation. An example of such an operation is edge detection. Several edge detection techniques exist and new methods continue to emerge. Each technique implements a different/new algorithm and may be appropriate under different circumstances. Thus, given an image object, it is advantageous not to bind the data to a particular function, but rather to an abstract function class and dynamically select an appropriate function from the abstract class at runtime.

Abstract functions simply define a logical operation, not the implementation, and postpones the binding of the actual implementation until runtime. In many cases, binding can be done automatically at runtime based on the current semantics of the data. As a result, the abstract function completely hides the implementation of the function from the user much like an abstract data type hides its implementation.

Inheritance: Given a raw image, two or more users (or applications) might process the same image and obtain different semantic data to be used for different purposes. For example, a satellite image might be of interest to both a geologist and a military analyst. Initial processing of the raw image (such as noise removal, enhancements) might be common to both applications before domain-specific processing takes over. In such situations, it is important to group common operations and objects together in a class and then let individual applications inherit the image attributes in a subclass avoiding duplication of efforts.

Composition: Composition refers to the merging of two or more distinct objects into a new object. For example, two independent pictures of the same scene may be merged together to produce additional information about the scene (e.g., the depth of objects in the scene). Alternatively, a single facial image might be processed to extract the individual facial features, modify some of the features, and then merge the modified facial features back together to create an object that can be easily searched for certain combinations of facial feature characteristics. In other words, the system must provide the ability to combine processed objects together into compound objects with new functionality that did not exist when the objects were independent.

History mechanism: As discussed earlier, an image typically goes through a series of transformations that extract information from the image

or compute new information based on the image. Thus, the state of an object must contain not only the current data and associated semantics, but also the sequence of operations that were applied to the original object to bring it to the current state. Such a history is especially useful in an interactive programming system where a user may wish to test a variety of alternatives for processing the visual information. Note that a history mechanism also includes the notion of saving the results of past processing so that it can be reused by future users/applications. Alternatively, the original user may wish to backtrack to a previous version of the data and resume processing from that point onward.

2.4. A Prototype System

We have implemented a prototype system that allows users to define abstract function groups, semantic data classes, and a semantic processing graphs that can be used to process images interactively and hence extract the embedded semantic information. The semantic information, along with the associated images, are then entered into a database (Illustra) for later retrieval. To demonstrate the power of the system, we have used the prototype to construct a processing tool that does simple feature recognition, music recognition, document title/author identification, and paleographic letter form identification in the Beowulf manuscript.

3. Semantic Searching

Identifying the content of digital data forms can be done most reliably by hand, but the large volume of data that is now available makes autonomous techniques necessary. Researchers in image processing and computer vision have long addressed problems such as pattern classification, object recognition and motion detection. Very few techniques, however, have been applied to the problem of extracting and storing information from video clips.

Our new approach for tagging semantic information in video clips is based on the Karhunen-Loeve (KL) transform (Fukunaga: 1990), which has been used by others for face recognition (Pentland *et al.*: 1994) and object recognition (Murase and Nayar: 1995). The key idea to the

efficiency of our approach is that we exploit the fact that video clips are stored digitally in a transformed, compressed format. The potential loss of fidelity is completely controllable, ranging from almost no loss to a large amount of distortion. This transformed format is the basis for the video storage standards that are now in place, including the full motion JPEG (MJPEG) and MPEG formats.

The standard MJPEG and MPEG formats are actually algorithms that transform a video through many steps with the goal of reducing the storage size. Classical approaches to the video semantic search problem completely decode (uncompress) each frame of the video and then search the pixels of the frame for objects to classify or recognize. Our approach can classify objects in the compressed stream without fully restoring (uncompressing) the video to the individual pixel level. Avoiding the complete restoration to the pixel level improves the system's computational efficiency without a loss in classification ability.

3.1. Recognition using Eigenspace

The basic idea behind the eigenspace method is to represent a large number of "training" images with a lower-dimensional subspace. When a new image is seen, it can be classified as being similar or very different from the training images with just a few operations in the pre-computed subspace. For example, one could compile a large number of images of an object like the White House. Once these training images are distilled using the Karhunen-Loeve (KL) transform, any new image can be classified as "containing the White House" or "not containing the White House".

Specifically, let the input to the KL process be a set of images $\mathbf{f} = \{\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_k\}$, represented as vectors, where each image in $\mathbf{f}$ has $n = x \times y$ pixels. We treat an image as a vector by placing the pixel values in scanline order. The first step in computing the KL transform is to subtract from each input vector the average value

$$\mathbf{m} = \frac{1}{k} \sum_{i=1}^{k} \mathbf{f}_i \tag{1}$$

of the input vector set. This generates a new vector set $\hat{\mathbf{f}} = \{\hat{\mathbf{f}}_1, \hat{\mathbf{f}}_2, \ldots, \hat{\mathbf{f}}_k\}$ where $\hat{\mathbf{f}}_i = \mathbf{f}_i - \mathbf{m}$ for $i = 1, 2, \ldots, k$.

Now we compute the covariance matrix $\mathbf{C}$ of the mean-adjusted input vectors $\hat{\mathbf{f}}$:

$$\mathbf{C}_{n\times n} = \mathbf{U}\mathbf{U}^T = \begin{bmatrix} \hat{\mathbf{f}}_1 & \hat{\mathbf{f}}_2 & \dots & \hat{\mathbf{f}}_k \end{bmatrix} \begin{bmatrix} \hat{\mathbf{f}}_1^T \\ \hat{\mathbf{f}}_2^T \\ \vdots \\ \hat{\mathbf{f}}_k^T \end{bmatrix} \tag{2}$$

The KL transform is obtained from the eigenvectors and eigenvalues of $\mathbf{C}$ by solving an eigenstructure decomposition of the form:

$$\lambda_i \Lambda_i = \mathbf{C}\,\Lambda_i \tag{3}$$

This decomposition produces n eigenvectors $\Lambda = \{\Lambda_1, \Lambda_2, \dots, \Lambda_n\}$ and their associated eigenvalues $\lambda = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$. Because $\mathbf{C}$ is real and symmetric, the eigenvectors are complete and orthogonal, forming a basis that spans the n-dimensional space. Thus, using the eigenvectors as a basis, the original input vectors $\mathbf{f}_i$ can be described exactly as

$$\mathbf{f}_i = \sum_{j=1}^{k} p_{i_j} \Lambda_j + \mathbf{m} \tag{4}$$

where, as before, $\mathbf{m}$ is the mean of the input vector set.

All k eigenvectors are needed in order for the equality to hold. But one attractive property of the eigenspace representation is that the eigenvectors can be ordered according to their associated eigenvalues. The first (largest) eigenvector is the most significant, encoding the largest variation of the input vector set. This ordering allows the original input vectors to be approximated by the first w eigenvectors $\{\Lambda_1, \Lambda_2, \dots, \Lambda_w\}$ with $w \ll k$:

$$\mathbf{f}_i \approx \sum_{j=1}^{w} p_{i_j} \Lambda_j + \mathbf{m} \tag{5}$$

Given two vectors that are projected into eigenspace, it is a well-known fact that the closer the projections in eigenspace, the more highly correlated the original vectors (Murase and Nayar: 1995). This gives us the following important fact:

Distance in eigenspace is an approximation of cross-correlation in the image domain.

The result is that the eigenspace gives a very powerful way to classify an image with respect to a set of known images. One can simply project the new image to eigenspace and measure its Euclidean distance to the other

known points in eigenspace. The closest neighbors are projected images which correlate the highest with the input.

3.2. The Compressed Domain

The main idea of our work is to formulate the eigenspace approach on input vectors that are *not* image pixel values, but have been transformed from the image domain to another representation. The semi-compressed domain is convenient since it is an intermediate form that compressed video frames and images must pass through during decompression.

The critical fact we have proven (Seales *et al.*; 1998) is that

Distance in the eigenspace constructed from semi-compressed vectors is an approximation of cross-correlation in the image domain.

One implication of this fact is that videos can be searched automatically for content using the eigenspace approach *without first decoding them.* This gives a big gain in efficiency without much loss of classification performance.

3.3. Experimental Results

We demonstrate the results of our method using a short (56 frame) video clip of a human face. The goal is to identify people (faces) that appear in an MPEG video clip without completely decoding the MPEG.

We constructed a semi-compressed-domain eigenspace from two poses of 26 people in the Olivetti Face Database[1] and two poses from a local subject. The local subject, the 27th person, was also the subject pictured in the test video clip. Figure 2 shows the 56 frame video clip on the left. The four pictures on the right show the poses from the database that were found to be present somewhere in the video. The top two images on the right are the two poses of subject 27 who was correctly identified. The two poses of subject 27 were taken on different days, with different clothes and under different lighting conditions than the video clip.

Matching was done in the semi-compressed eigenspace, using only the DCT frames of the MPEG sequence. We used only 10 eigenvectors (of the possible 54) for the results shown. The highlighted boxes in the video

[1] We gratefully acknowledge the Olivetti Research Laboratory for making their data publicly available at (http://www.cam-orl.co.uk/facedatabase.html).

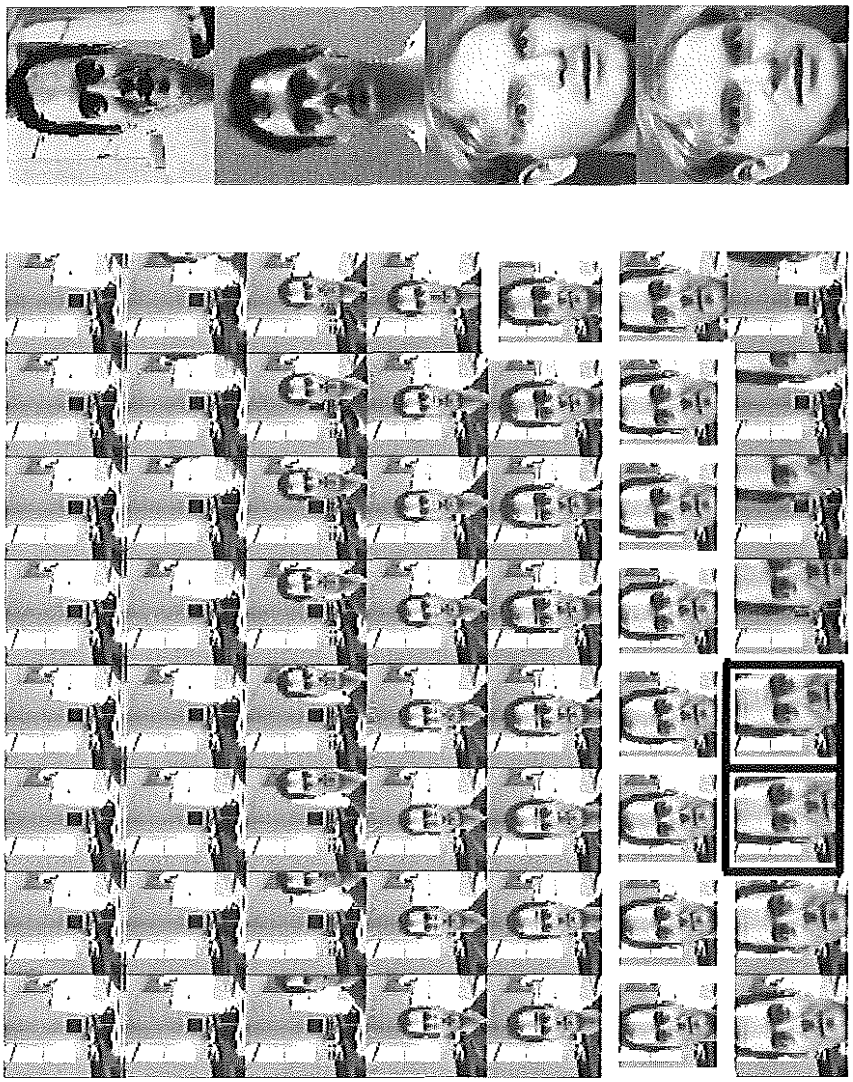

Fig. 2.– The input video sequence appears on the left while the faces identified appear on the right. The white boxed frames are those recognized as containing subject 27. Black boxed frames are false matches.

sequence on the left identify the frames that were found to match a face from the database. Frames 40–47 were correctly recognized as subject 27. Only frames 51 and 52 were incorrectly identified as matching subject 18 from the database. This misclassification was the result of image scale, since the scale of subject 18 matches those frames better than subject 27. Despite the efficiency and accuracy of our algorithm, this example shows that false positives are still possible and a higher-level thresholding policy must be used to eliminate these false matches.

Finally, we measured the computational time necessary for scanning the frames of an MPEG video clip. We instrumented the public domain MPEG software from UC-Berkeley, and compared the execution time for obtaining DCT frames (before the inverse DCT) to the time necessary to obtain the complete image (after the inverse DCT). We found that the dominant cost is the inverse DCT, and that obtaining only the DCT frames takes one half the time of completely decoding the images.

4. Network Access To Multimedia Data

Although existing Internet middleware tools such as the World-WideWeb, Gopher, Archie, and others provide access to multimedia information, we believe that they are not yet capable of dealing with the massive scale and size of typical digitized documents and their associated information. For instance, the Electronic Beowulf is a digital archive consisting of thousands of high fidelity images of the entire Beowulf manuscript with detailed subsidiary images acquired with special lighting techniques; the huge composite codex that contains the Beowulf manuscript; eighteenth-century transcripts of the manuscript before it was extensively damaged by fire; nineteenth-century collations of the manuscript with its first edition; and modern editions, transcriptions, and translations. The original digital images are each 21 MB or larger (Kiernan: 1994a, 1994b, 1995); although prepared to work on the Internet, the restrictions we discuss here make it necessary for now to publish the Electronic Beowulf archive on CD with radically compressed images. Existing tools are not designed to allow user manipulation of the information on this scale or to display, search, and browse the massive digitized environments we envision.

To support efficient on-line interactive access to such digitized information, we have developed algorithms that enhance existing HTML/HTTP middleware services.

4.1. Cooperative Data Caching and Parallel Retrieval

Currently, HTTP proxy servers support a limited form of information caching (Luotenen and Altis: 1994) in which a proxy server caches information accessed on behalf of several nearby clients, typically located on a LAN. However, proxy servers located within the same geographical region do not communicate with each other to share information in their caches. Instead, a proxy server contacts the original information provider (e.g., server at a given URL) whenever the desired information is not found in its cache. This organizational structure can still cause bottlenecks at popular URL servers and does not use network bandwidth efficiently. An alternative is to discover dynamically proxy servers in the immediate vicinity that currently cache the desired information. For scalability reasons, such cooperation among proxy servers should be as stateless as possible.

We have extended the HTTP proxy server mechanism to exploit the IP/UDP Multicast (group communication) to locate dynamically other proxy servers that have the desired URL document in their caches. To reduce the host processing bottlenecks at proxy servers and allow real-time retrieval and playback of multimedia documents, we have added extensions for parallel retrieval of parts of a document from different proxy servers.

4.2. Data Prefetching

Communication latencies are inherent in wide area networks (WAN's) which span a huge geographic area. It is possible to hide some of this latency from the user by predicting future access patterns based on the past history of accesses. Such a prefetching strategy can use mechanisms such as local storage or proxie caches to mask the latency of the WAN. Care must be taken, however, not to waste valuable (shared) Internet bandwidth on unnecessary data. Finding the balance between unnessary prefetching and the desire to avoid network latencies for data transfer is difficult.

We have examined automated methods for predicting future document accesses based on past access histories *prefetch-s-usenix-94*. Our heuristic-based prefetching algorithm that can, with high accuracy, retrieve multimedia documents before they are needed. The algorithms take a probabilistic approach by using a probability graph structure. This structure limits the amount of information that needs to be maintained

about past accesses. Our results show that documents can be prefetched with a high degree of confidence without significant computational overhead.

4.3. Hierarchical Data Transfer

We are currently investigating techniques for the use of hierarchical data encoding in digital libraries. The need for this is motivated by the responsiveness of an on-line data access system, which depends largely on the quality of the underlying communication link. Because various network technologies have proliferated, including wireless communication and slow-speed dial-up links, an on-line data access/retrieval system must be designed to perform well over slow speed links as well as congested network paths. One increasingly popular technique for accommodating heterogeneous networks is hierarchical encoding. Multimedia information is stored at multiple resolutions, and the appropriate level of resolution is selected and transferred automatically based on parameters such as the speed of the link.

As an example, consider the user who wishes to view an archived manuscript on-line. When the network is congested, the system automatically compensates for the low transfer rate by fetching a lower-resolution version of the image data. The ability to fetch the low-resolution version is dependent on the detection of a congested network situation as well as the storage of the data in a format that makes the lower-resolution data readily available. Users who wish to wait can still run the system in a fixed-resolution mode. However, quick access to lower resolution contents is often desirable when browsing a document. Solving these problems requires new encoding and storage formats and also adaptable communication protocols. This is the focus of part of our ongoing research.

5. Conclusion

Digitized images, video and audio are rapidly becomming commonplace. Such multimedia will soon become the primary data format in databases, augmenting or completely replacing conventional alpha-numeric data. New techniques are needed to access, manage, and search these new multimedia data types. We have presented a flexible multimedia data management system called MOODS that treats the embedded semantic information contained in a multimedia document as a first class entity that can be identified and searched for like conventional data. We also presented an efficient algorithm for automatic identification of semantic information in compressed images and video. Finally, we described extensions to existing middleware languages and protocols such as HTML/HTTP to improve remote network access to multimedia data, an increasingly important problem as we move to a completely networked world.

Acknowledgements

This work is supported in part by the National Science Foundation and the Center for Computational Sciences and the College of Arts and Sciences at the University of Kentucky.

Bibliography

FUKUNAGA (K.): 1990, *Introduction to Statistical Recognition* (Academic Press).

GRIFFIOEN (J.) and APPLETON (R.): 1994, "Reducing File System Latency Using a Predictive Approach", in *The Proceedings of the 1994 Summer USENIX Conference* (USENIX Association, June 1994), p. 197–207.

GRIFFIOEN (J.), MEHROTRA (R.) and YAVATKAR (R.): 1993, "An Objet-Oriented Model for Image Information Representation", in *The Proceedings of the Second International Conference on Information and Knowledge Management* (November 1993), p. 393–403.

GRIFFIOEN (J.), YAVATKAR (R.) and ADAMS (R.): 1995a, "An Object-Oriented Model for the Semantic Interpretation of Multimedia Data", in *The Proceedings of the ACM Multimedia '95 Conference.*

GRIFFIOEN (J.), YAVATKAR (R.) and MEHROTRA (R.): 1995b, "A Semantic Data Model for Embedded Image Information", in *The Proceedings of the 1995 IST/SPIE Symposium on Electronic Imaging Science and Technology*, February 1995, volume 2417, p. 414–425.

KIERNAN (Kevin): 1994a, "Electronic Beowulf", in *National Geographic.*

KIERNAN (Kevin): 1994b, "Scholarly Publishing on the Electronic Networks: Gateways, Gatekeepers and Roles in the Information Omniverse", in *Digital Preservation, Restoration and Dissemination of Medieval Manuscripts.*

KIERNAN (Kevin): 1995, "The Electronic Beowulf", in *Computers in Libraries.*

LUOTENEN (Ari) and ALTIS (Kevin): 1994, "World Wide Web Proxies", in *Proceedings of the CERN WWW Conference*, August 1994.

MURASE (H.) and NAYAR (S.): 1995, "Visual learning and recognition of 3-D objects from appearance", in *International Journal of Computer vision*, 14: 5–24.

PENTLAND (A.), MOGHADDAM (B.) and STARNER (T.): 1994, "View-based and modular eigenspaces for face recognition", in *Proc. IEEE Computer Vision and Pattern Recogn.*.

SEALES (W.B.), CUTTS (M.D.), YUAN (C.J.) and HU (W.): 1998, "Object recognition in compressed imagery", in *Image and Vision Computing.*