INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Université d'Ottawa · University of Ottawa

Query Interface for Multimedia Database System

by

Jiandong Cheng

A thesis submitted to the
School of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

M. A. Sc.
in
Electrical Engineering

Ottawa-Carleton Institute of Electrical Engineering

Department of Electrial Engineering
Faculty of Engineering
University of Ottawa
Ottawa, Ontario

February, 1998

© Jiandong Cheng

National Library of Canada
Bibliothèque nationale du Canada

Acquisitions and Bibliographic Services
Acquisitions et services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-28408-5

Canada

Table of Contents

TABLE OF CONTENTS ..... **I**

LIST OF FIGURES ..... **IV**

ABSTRACT ..... **V**

ACKNOWLEDGMENT ..... **VI**

CHAPTER 1 INTRODUCTION **1**

1.1 MOTIVATION 1

1.2 OBJECTIVES 3

1.3 THESIS ORGANIZATION 4

1.4 MAIN CONTRIBUTION 5

CHAPTER 2 MULTIMEDIA DATABASE SYSTEMS **7**

2.1 OVERVIEW 7

2.2 MULTIMEDIA INFORMATION SYSTEM 8

2.3 MEDIABASE PROJECT 11

2.4.1 Multimedia conferencing system and computer supported co-operative work *13*

2.4.2 Multimedia News System *18*

2.4.3 Multimedia Telelearning *21*

2.5 SUMMARY 24

CHAPTER 3 USER INTERFACE ISSUES **26**

3.1 INTRODUCTION 26

3.2 FUNCTIONALITIES OF USER INTERFACE FOR DATABASE SYSTEMS 28

3.3 QUERY FACILITIES FOR DATABASE SYSTEMS 31

3.4 QUERY LANGUAGES FOR DATABASE SYSTEMS 33
3.4.1 Query on Traditional Database *34*
3.4.1.1 SQL 34
3.4.1.2 Query-by-example (QBE) 35
3.4.2 Query on Multimedia Database *35*
3.4.2.1 Introduction 35
3.4.2.2 Image Database 38
3.4.2.2.1 Attribute-based 39
3.4.2.2.2 Text-based 39
3.4.2.2.3 Content-based 40
3.4.2.2.4 Icon-based 41
3.4.2.3 Video Database 43
3.4.2.4 Audio Database 44
3.5 RELATED WORK 45
3.6 SUMMARY 50

CHAPTER 4 THE GRAPHICAL QUERY INTERFACE FOR MULTIMEDIA DATABASE 52

4.1 INTRODUCTION 52
4.2 ARCHITECTURE OF THE USER INTERFACE 53
4.3 MULTIMEDIA QUERY LANGUAGES 55
4.4 GRAPHICAL QUERY INTERFACE 58
4.4.1 The Need for Graphical Query *60*
4.4.2 Intermedia Query *60*
4.4.2.1 Temporal Query 61
4.4.2.2 Spatial Query 63
4.4.3 Intramedia Query *64*
4.4.3.1 Static media 64
4.4.3.1.1 Outline query 65
4.4.3.1.2 Object Query 66
4.4.3.1.3 Color query 69

4.4.3.2 Dynamic media 70
4.4.3.2.1 Video temporal specification 71
4.4.3.2.2 Video content specification 72
4.4.4 Database Browser *74*
4.6 SUMMARY 75
CHAPTER 5 IMPLEMENTATION 77
5.1 INTRODUCTION 77
5.2 QUERY SPECIFICATION MODULE 78
5.2.1 Steps of Query Construction *79*
5.2.2 Data Structure *82*
5.3 QUERY TRANSLATION MODULE 86
5.4 QUERY EXAMPLE 90
5.5 EVALUATION 92
5.6 SUMMARY 94
CHAPTER 6 CONCLUSIONS 96
6.1 SUMMARY 96
6.2 SUGGESTIONS FOR FUTURE WORK 98
BIB BIBLIOGRAPHY 100
PUBLICATION 1007

List of Figures

TABLE 2.1 MULTIMEDIA CONFERENCING FUNCTIONS 16

FIGURE 2.1 FUNDAMENTAL ARCHITECTURE OF A MULTIMEDIA NEWS-ON-DEMAND SYSTEM 21

FIGURE 2.2 GENERIC ARCHITECTURE OF TELELEARNING SYSTEM 23

FIGURE 3.1 CLASSIFICATION OF IMAGE DATABASE SYSTEMS 38

FIGURE 3.2 SAMPLE OF INDIVIDUAL TEMPORAL RELATIONSHIPS 50

FIGURE 4.1. THE FUNCTIONAL ARCHITECTURE OF THE USER INTERFACE 54

FIGURE 4.2 MULTIMEDIA QUERY FORMULATION 57

FIGURE 4.3. QUERY DECOMPOSITION 59

FIGURE 4.4 TEMPORAL QUERY WINDOW 61

FIGURE 4.5 SPATIAL QUERY WINDOW 64

FIGURE 4.6 OBJECT QUERY EXAMPLE 68

FIGURE 4.7 COLOR QUERY WINDOW 70

FIGURE 4.8 VIDEO QUERY FLOW 73

FIGURE 4.9 DATABASE BROWSER 74

FIGURE 5.1 THE FLOW CHART OF THE USER INTERFACE 78

FIGURE 5.2 QUERY SPECIFICATION SEQUENCE 79

FIGURE 5.3 STARTUP WINDOW FOR THE USER INTERFACE 81

FIGURE 5.4 MAJOR COMPONENTS OF DATA STRUCTURES 86

FIGURE 5.5 ABBREVIATED ALGORITHM OF QUERY TRANSLATION MODULE 89

FIGURE 5.6 HIERARCHY OF TEMPORAL OBJECTS 90

FIGURE 5.7 QUERY EXAMPLE 92

FIGURE 5.8 QUERY TRANSLATION EXAMPLE 93

Abstract

Multimedia database system differs significantly from traditional database system. The introduction of non-numerical media types, such as image, video, and audio, requests the development of database technology to handle the challenge. At the same time, the demands for multimedia applications are growing rapidly. The central repository of almost every multimedia application is exclusively a multimedia database system. Thus, the efficient query and retrieval information from the database becomes an important issue. To fulfill the task of information retrieval, a multimedia query language, which takes into account the complex spatial and temporal relationships as well as media information, should be integrated with the database system.

One of the difficulties in dealing with multimedia data is how to effectively express its very rich semantics in a query. It might be with obstacle for some users to translate the perceptions of multimedia information into the expression of query language. Therefore, a friendly and easy-to-user graphical user interface is needed on top the query language to help the user to construct queries.

In this thesis, an interactive query interface for multimedia database named MEDIAQUINT is proposed. It focuses on supplying visual query facilities to end-users so that they can formulate the query based on their memory representation of the interested multimedia scenario. The system architecture and implementation of the graphical user interface are presented.

Acknowledgment

I would like to convey appreciation to my thesis supervisor, Dr. Ahmed Karmouch, for his continuous guidance throughout the course of this thesis. Dr. Karmouch provided a very good working environment in which to access the up-to-date technology.

I would like to thank every colleague in the MIRL lab. Their kindness and friendship make my stay in the lab unforgettable.

I would also like to express my thanks to Milkyway Networks Corporation for providing support to me.

Finally, I would like to thank my family and all the people who provided assistance to me.

Chapter 1

Introduction

1.1 Motivation

Database management systems have received growing interests in recent years with the advent of the multimedia systems. Text has been the dominant medium for the users to interact with computer systems for a long time. Indeed, the interests of computer users have switched from the traditional alphanumerical data to a wide variety of media types such as image, video, audio and graphics. This resulted in the development of multimedia database systems, which

integrate information from diverse media sources and allow the user to interact with the real world in a broader and richer means.

Each medium type has its own characteristics. There also exist complex spatial and temporal relationships within some media types or among themselves. This results in difficulties for users to manipulate and access information stored in the database. Ideally, the multimedia database system should have the ability to efficiently organize and store the multimedia objects and model the complex relationships between them. One of the fundamental functionalities of a database system is to maintain information and to make the information available on user's demand. The query language is a high level language that can be used by the user to specify the query information that the user wants to retrieve from the multimedia database. The multimedia query language, differentiating itself from the traditional query language, which only deals with texts and numbers, takes into account the complex spatial and temporal relationships of multimedia information. Such a versatile query language should be equipped with the database system.

One of the difficulties for users in dealing with multimedia data is how to effectively express its very rich semantics in a query. Although there were a well-developed query language with high expressive power associated with a database system, it would also be difficult for some users to translate their perceptions of multimedia data into the query language, such as SQL-like query language. Thus, a graphical user interface is needed on top of the query language to relieve the user from tedious translation work while constructing the query. From this point of view, the user interface serves as a bridge between the end user and the query language of the

system. Without having the knowledge of the specific query language, the user could use visual methods provided by the GUI to specify the information to be retrieved from the database.

In the Multimedia Information Research Laboratory of University of Ottawa, a multimedia information and communications system named MEDIABASE is under development. It is composed of different components: a multimedia user interface for information retrieval, multimedia database client and servers for storing, retrieving and updating multimedia documents, and a production server to capture multimedia data and to create multimedia document.

One of the major issues in designing a multimedia information systems is providing users with a user-friendly interface so that the rich information stored in the database could be conveniently accessed on user's demands. However, the traditional user interface for database system only support texts and numbers, such as table-driven queries. Therefore, it is necessary to design an interactive user interface to deal with the very rich semantics of multimedia information. On such a user interface, users can easily construct query specifications based on their memory representations of the image of the real world.

1.2 Objectives

The main objective of this thesis was to design and implement MEDIAQUINT (MultiMEDIA QUery INTerface), an interactive query interface for multimedia database systems. Many multimedia information systems provide users with a browsing-based user interface. However, a typical multimedia database, especially if it is video and image intensive, might require hundreds of gigabytes physical storage. Only a browsing tool could not support the

goal of efficient access to such a rich database. This thesis is concentrating on supplying query facilities to end-users so that they could retrieve the information of their interests from the database without difficulties. The user could also make use of the database browser to gather some general information of the database, which in turn could help the user to better construct query specification.

Generally speaking, the user interface for the multimedia database has to possess the following characteristics:

- Straightforward and easy-to-use for the end users.
- Highly expressive power to handle the complex relationships, such as spatial and temporal relationships, embedded in the multimedia data.
- Using visual methods to relieve the user from tedious work of translating complicated queries into a specific query language.
- With a general purpose browser to briefly navigate through the database

1.3 Thesis Organization

The thesis is organized as follows:

Chapter 2 introduces the basic concepts of multimedia database system, including the requirements and characteristics of multimedia database. Some of its applications, such as news-on-demand and telelearning systems, are described. The MEDIABASE project, which is under development in our lab is also introduced briefly here.

Chapter 3 concentrates on issues concerning user interface of database systems. The necessary funtionalities a user interface should have are discussed. The query language and query facilities are also introduced. This chapter shows us the query specifications on different types of database system, including traditional relational database and multimedia media database.

Chapter 4 focuses on our graphical user interface for multimedia database, which is called MEDIAQUINT. The requirements and architecture are presented along with the multimedia query language of our database system. The design of the query interface is discussed in detail.

Chapter 5 reports the implementation of MEDIAQUINT. Two modules, says query specification module and query translation module, are described. The data structures and algorithm for translation module are also presented. Conclusions and suggestions for future work are given in Chapter 6.

1.4 Main Contribution

The main contribution of this research is the design and implementation of graphical user query interface for multimedia database system. It is designed on top of the proprietary multimedia query language. It focuses on providing a visual facility to the end users of a multimedia database to allow them to directly and efficiently retrieve interested information from the database based on the contents of the information they could see in their minds. The user can break down a complex query into several steps and construct the query on different query windows. The final query will be formulated by logically combining the various windows

together. MEDIAQUINT also provides some sort of browsing capabilities to help the user to navigate through the information in the database. The queries that are constructed on the query interface are translated into the format of query language on the database system. They will be further utilized by the query engine in search of the related multimedia information in the database, and the query results will be presented back to the user in a suitable format on the user interface.

Chapter 2

Multimedia Database Systems

2.1 Overview

The database system techniques experienced rapid development recently. A database system is basically a computerized information keeping system, whose overall purpose is to maintain information and to make that information available on user's demands. Almost all of the databases developed in the past were based on the relational approaches, which represent the dominant trend in today's market place. The recent advances of information compression and storage techniques and broadband networks technologies matures the conditions for developing

multimedia systems technically and economically [DNN91] [GDL91] [FEA91]. People no longer restrict themselves within the small range of information media. Instead, they are becoming more and more interested in combining various media types together to enrich the channels of interacting with the real world [COB93]. The most important reason for the recent popularity of multimedia information systems is that they provide an easier and effective way of human-computer interaction. This brought us the era of multimedia information.

2.2 Multimedia Information System

What is a multimedia system? It is quite self-explanatory from the meaning of the word "multimedia" that it must support a wide variety of media types. However, this is not sufficient enough to define the multimedia environment [NWI94]. More important, the abundant information resources should be efficiently integrated as an entity so that the end users could deal with an integrated framework instead of a combination of separated sub-systems.

Multimedia requires the integration of storage, communications, and presentation mechanisms for the diverse information types in a single platform so that the information can be manipulated and played-back simultaneously. From this point of view, there are two aspects of concerns: the forms of media available to an end user and the ability of the system to communicate with the end user using these media forms. The range of media used to carry information has been increased dramatically with the advances of computer technologies in recent years. The media types supported by multimedia information systems include both static media such as text, graphics, image, and dynamic media such as animation, video and audio. Each media type has its own characteristics that are quite different from one another. For

example, the storage requirement for video is very big. In order to store a video stream with the length of a second of, say 24 bits per pixel, 640x480 pixels per frame and 30 frames per second, it requires approximately 28 Mbytes disk space without using video compression techniques.

Historically, the development of multimedia information systems could be differentiated into three generations since the first multimedia application was introduced in 1989 [BOF94].

The first generation was characterized by bitmapped images and animations, JPEG compression techniques, hypermedia authoring tools and local area networks. The second generation made use of the JPEG and MPEG-1 video compression, high speed networks and multimedia authoring tools. Currently, we are at the transition stage from the second to the third generation, which will utilize more powerful processors, full-motion video, MPEG-2, 3, or 4 compression techniques, ATM networks and objected-oriented multimedia authoring tools that integrated video, audio, image, graphics and text together.

The research and development of multimedia information systems falls under two categories. One category focuses on the stand-alone applications, such as computer-aided learning and music composition, which were the early stages of multimedia applications. The other category emphasizes more on distributed applications, which represents the trend of today's multimedia development. The merging and evolvement of computer communication networks and multimedia computing technologies laid the ground for the development of this type applications. Such applications utilize computer networks to transport the multimedia data from the information servers to the clients. The plentiful media information could be stored on physically-separated servers connected by high-bandwidth communication networks, such as ATM. A simple scenario is to store each type of media on different servers that are distributed

over a network. The users could retrieve information on their demands from the servers as if they are retrieving data locally without noticing the existence of the network. New applications based on such ideas include distance learning, news-on-demand, video conferencing and collaborative work.

There are several key components of distributed multimedia information systems. Since multimedia information can originate from a wide range of sources, it is highly essential to have a powerful data generation and processing module associated with the system which is responsible for capturing, creation, manipulation and processing the different elements of different information types. The high voluminous characteristics of multimedia data determine that effective compression and storage components are required. Normally, the multimedia data are physically stored in the hard disc of a computer. Without the adoption of data compression techniques, a physically large hard disc can only store a relatively small amount of information. For instance, a minute-length of video signal will require approximately 1.8 GB hard disc space. In order to store more information in the restricted physical space, the image/video compression technologies, such as JPEG and MPEG, which could reach a compression ratio ranging from 50:1 to 200:1, have to be introduced to attain the goal of effective storage.

Since the purpose of any information system is to keep record of the information and present them on user's demand. Therefore, the retrieval and presentation component is another key elements of the system. The rich information stored in the system might not be available to the users without the help of an effective retrieval and presentation module. The user interacts with the system through the graphical user interface (GUI) on which the user could formulate the query specifications, browse the query results, and playback the desired information. Here, it

requires simultaneously dealing with temporal and spatial relationships of different media types. For instance, in order to present the various types of information stored at physically different locations, the issue of real-time synchronization of this information has to be taken into account.

The last major component of such a system is the communication network that connects the physically different-located components together. Broadband networking is one of the catalytic forces driving multimedia forward. A typical scenario of a multimedia application might follow this: a user's workstation may have multiple windows simultaneously: a video window supporting interactive video conference, another window displaying multimedia e-mail which might include pictures, and a text window displaying the instruction messages. Different information types have their own requirements and quality of service (QoS) while being transferred over the networks. For instance, the audio is not delay tolerant, which means it could allow some bit losses. On the contrary, video could tolerate more delay and less bit losses. In order to support the requirements originated from different information types, a reliable communication network with high bandwidth, low error rate and guaranteed QoS, such as Fiber Distributed Data Interface I and II (FDDI I, II), Fast Ethernet, Asymmetric Digital Subscriber Lines (ASDL), and ATM, is highly required. The ATM network will be positioned in dominant place in developing multimedia applications in the near future.

2.3 MEDIABASE project

MEDIABASE is a distributed multimedia information system under development at the Multimedia Information Research Laboratory of University of Ottawa [KAA93]. The MEDIABASE is mainly composed of four components: (1) a production center for capturing

various types of information and creating multimedia documents; (2) a multimedia client and server for indexing, storing, updating and retrieving multimedia documents; (3) a multimedia user interface for queries, browsing, and presenting the multimedia documents; (4) high speed networks, such as FDDI and ATM for transmitting the multimedia document from servers to the clients' sites.

The MEDIABASE is a fundamental architecture. On top of such a generic platform, some major applications are under development, including multimedia news-on-demand system, interactive telelearning system, and intelligent multimedia news agent, etc.

2.4 Applications of Multimedia Systems

The advent of the multimedia technology, associated with other industries, such as computing, communication and broadcasting, force researchers to both create new application areas and improve what have already existed so far. Though the multimedia research is still at the beginning stage, there are a wide variety of applications already commercially available in the market or under development. The multimedia mail, multimedia conferencing and collaborative work, multimedia news, and telelearning systems are important applications, which are already in use and attract more enthusiasms from researchers and customers. Though there are some multimedia applications based on stand-alone workstation, such as music composition, financial consultation, people are paying much more attentions on distributed applications, which represent the unavoidable trend in the future.

Currently the range of multimedia applications spans a wide variety of fields, such as office automation, service industry applications, retail application, domestic application, science and engineering, and culture activities [NWI94]. In this section, let's take a brief look at some important multimedia applications, including multimedia conferencing system and Computer Supported Cooperative Work(CSCW), multimedia news system, and telelearning.

2.4.1 Multimedia conferencing system and computer supported co-operative work

The term "Computer Supported Cooperative Work" (CSCW) has been used since 1984 to describe how computers support group working. A simple framework has been defined, with

reference to the time and the spatial relations involved in the applications, to classify CSCW. For example, the CSCW solutions, which span a period of time and happen at different places, might be in the formats of email, newsgroups and group writing, etc. Multimedia conferencing system falls under another category of CSCW: happening at the same time and involving different places, which means synchronous interaction between geographically separated participants [ASE92].

Multimedia conferencing system enable a number of participants in local or remote locations to exchange various kinds of multimedia information, such as video, audio, graphics, document, and data, via high speed networks [VHM91]. Each participant has a multimedia workstation. The workstation has the ability sending and receiving information, and performs certain collaborative activities as well. A typical PC-based multimedia conferencing system would have following functions,

- Desk-to-desk conferencing with local and remote participants.
- Shared-screen workspace to collaborate on documents, designs, software, presentations, and so on.
- High-speed media transfer.
- Call-management features.

In such a system, users interact with each other asynchronously. From this point, simultaneous (or real-time) interaction is essential for the system. Taking a real-time conference

as an example, each participant is sitting by a workstation in his/her own office, together with some other facilities, such as a microphone, a speaker and a camera. The screen of each participant can be divided into several windows for displaying various kinds of information, which might be the gesture of the remote participant, the document needed to collaborate, etc. Participants of discussion and negotiation can use the voice equipment. The displayed information can be dynamically edited or updated by the specified participants. The reference information, which may be related to the current discussion, can be retrieved for display at any time. Each participant has his/her own private workspace on the screen to view or edit the relevant data before submitting it to the shared workspace in the conference. Table 2.1 details the functions of a multimedia conferencing system.

Function	*Characteristic*
Capture	Media capturing devices, such as real-time camera, microphone, etc.
Storage	Data compression techniques for storage and transmission over the networks Multimedia database and document structure to store the infomation
Retrieval	Either store-and-forward or real-time usage of stored material, such as query and playback of media clips
Presentation	Display the information on the platforms
Transfer	High-bandwidth networks, such as FDDI, frame relay, or cell relay
Coordinator	Dynamic allocation of network resources, synchronization of shared workspaces, and call initialization and maintenance, etc.
Group sessioning	An effective mechanism to connect multiple participants, including multipoint control units, point-to-point and multipoint-to-multipoint real-time signaling

Table 2.1 Multimedia Conferencing Functions

Systems that provide some of the features above already existed as early as 1968. The name of the system is NLS, which provides a shared-screen mode for simultaneous collaborative authoring of structured documents. The commercial successor of NLS is a system named "Tymshare's Augment", which supports "virtual" terminal linking across different types of terminals.

Multimedia conferencing systems are a development of the original electronic mail systems, which allowed a user who is using a central machine to transmit textual messages to other users within the same system. As Wide Area Networks (WANs) designed to support computer communication became widespread, both the complexity and the functionality of the

electronic mail system improved rapidly. This development resulted in the formation of the multimedia conferencing system. Until recently, voice, video and data communications have been handled with different communication networks. The primary carrier of voice is the public switched telephone network, video is transmitted by cable television network, and data by specialized computer networks (LAN's and WAN's). The reason for the separation of the information is their fundamental differentiation in their characteristics. Voice and video characteristics, such as sensitivity to delay, high-bandwidth requirement, and the ability to tolerate high error rates, contrast with the data requirements. Hence, voice and video transmissions use circuit switching of analog signals. On the other side, data transmission uses digital packet switching networks. Advances in communication and computer technology have made it possible to integrate these different types of media together to form a composite stream, which is transmitted through a single network.

Prototypes of multimedia conferencing system include MMConf, implemented in the Diamond systems; RTCAL(real-time calculator), developed at MIT; Rapport, developed at AT&T; the European Collaborative projects MIAC (Multipoint Interactive Audio visual System); and IBM's person-to-person. Several years ago, people made a great effort to develop mechanisms for person-to-person conferences in the range of shared text-oriented workspace. For the time being, the emphasis has been switched to multipoint conferencing [CWJ92]. At present, desk-to-desk multimedia conferencing system has seen only limited commercial introduction. Both lack of standards and unavailability of cost-effective high-speed networks contribute to hold back the application.

2.4.2 Multimedia News System

Multimedia news applications provide the users with the access to the multimedia news documents that are stored in a distributed multimedia database system. By subscribing to the news providers (such as television networks, newspapers, magazines, or combination of these media suppliers), the users can retrieve and read multimedia documents, which are transmitted over broadband networks, of their own interests.

Most distributed multimedia applications adopt client-server architecture. So does the multimedia news application [FKA96]. It follows the multiple servers and multiple client models. The client and server communicate with each other over the networks in the form of messages. There are many types of messages classified on the basis of their purposes, such as request message, polling message, reply message, etc. The client might send a request message, asking some kind of transactions, to the server. On the server side, a daemon program is running all the time to listening to the inbound messages from the networks. Upon message arriving, the server will interpret the message and perform the corresponding transactions on the request of the message. Finally, the result of the transaction will be sent back to the client, such as a multimedia news document requested by the client. Figure 2.1 shows the fundamental architecture of a multimedia news-on-demand system. The source of all the multimedia news is the production center, which is responsible for capturing, editing, and authoring the news resources. A powerful multimedia authoring tool capable of automating the process of news integration and organization is highly required at the production site.

New issues are raised in such a news-on-demand distributed multimedia environment. In many cases, it requires continuous presentation of a media stream and the synchronized playback of multiple data streams. As a result, the allocation of the system resources, both at the server site and client site, becomes one of the most important aspects of system design. In other words, a well-designed system must be able to maintain balanced tradeoffs between granting the requirements from clients and still providing the overall performance of the system. The client site is responsible for synchronizing the multimedia packets sent from the server and delivering to the output devices. In case of necessity, the client might skip or drop some packets to maintain the synchronization requirement. This raises the problem of buffer management. More over, the client sometimes might send back messages to the server for schedule changes. On the other side, the server is client-driven, which means the server transmits the data on the requests from the client. The quality of service (QoS), such as guaranteed bandwidth allocation, is also the job of the server.

Based on the concepts mentioned above, [FAB95] proposed a multimedia news delivery system over ATM networks. The architecture of the system is similar to Figure 2.1. It is a multimedia newspaper application. The news articles in the multimedia newspaper are composed of multiple media that is opposed to the traditional newspaper containing only text and graphics. The user can use a Netscape-like user interface to get access to the news server and browse the news sources. The articles that interest the user could be downloaded to the local platform for further reading. The user could also associate links to documents much like the way the bookmarks of Netscape does. At the local site, the system provide the user with a powerful video browsing tool, which support the user to browse the content of a video stream on a video tile. The video tile decompose the video stream into many, says 12, segments and display the first

frame of each segment on a tile on the screen. Each tile, which represents a segment, can be further decomposed and shown on tiles. This provides a more convenient mechanism to browse the content of video without watching all the frames.

With the rapid increase of the volume of information, users experienced difficulties in picking up valuable information for the reason of the diversity and abundance of the news resources. This resulted in the concept of agent software, which could help the users to collect interested news from different news servers over the network so that the users could utilize the news servers in a much more efficient way [KWK95]. The agent software, which are located at client's sites, employ content analysis and capture techniques to select information that closely matches clients' interests, and then schedule the selected information for clients' preferred viewing times. Based on the clients' profiles constructed by monitoring clients' choices and behaviors, the agents determine whether the information is relevant to client or not. The agents also make use of feedback from the clients to refine the profiles in order to make them much more relevant to the clients' interests. Agent-helped news system might be the developing trend of multimedia news applications in the next few years.

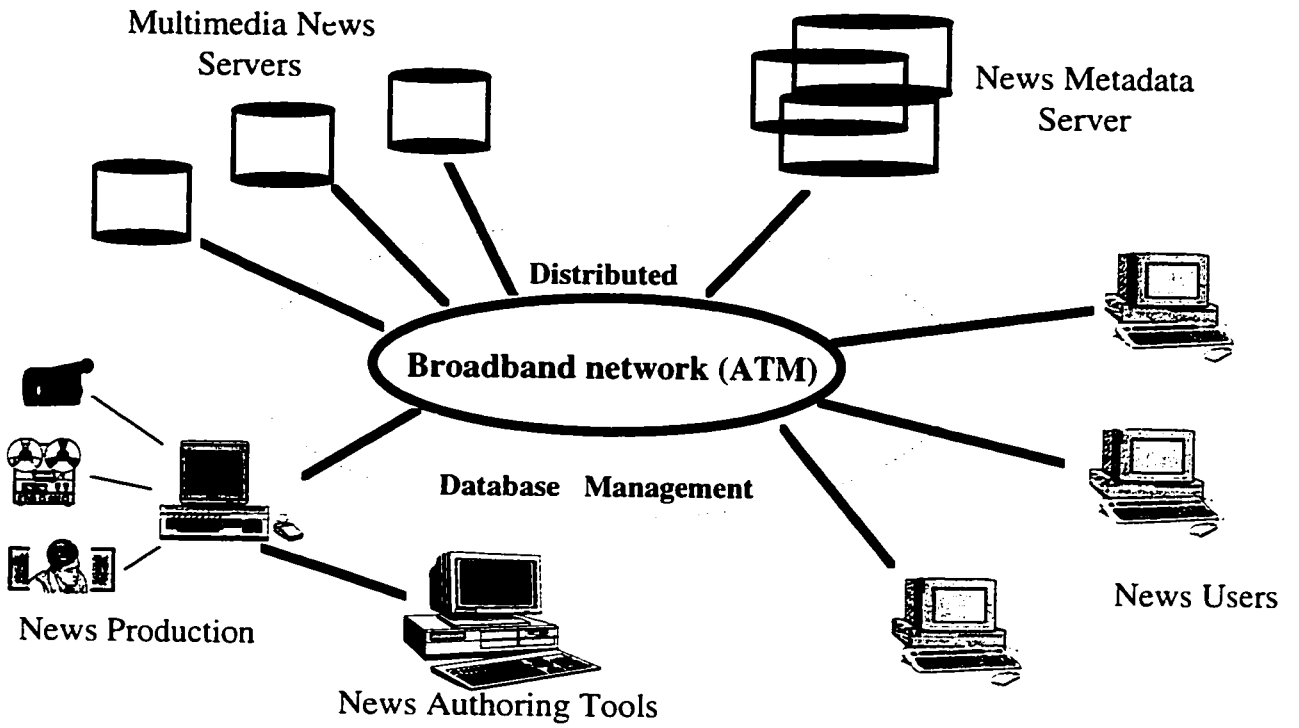


Figure 2.1 Fundamental architecture of a multimedia news-on-demand system

2.4.3 Multimedia Telelearning

Telelearning can be also called distance learning, or distance education. It means the delivery of teacher's intruction to students beyond the constraints of the classroom. The traditional way of telelearning in most people's mind is the radio and television broadcasting of teacher's teaching in the classroom. The major drawback of the method is the lack of 2-way communication channel between teacher and student. The role of student is passive information absorber. The teacher can't get the feedbacks from students instantly to check out the teaching effectiveness.

Multimedia telelearning refers to utilize various types of media, especially video, audio and image, in the learning process in an effort to provide a more interesting and effective learning environment to the student. With the advances of interactive personal computing and the availability of software to incorporate various kinds of media, the multimedia learning becomes technically practical. The simplest application might be a hypermedia document containing the introductory information of a topic in the forms of image, sound and video. With a multimedia platform equipped with corresponding accessories such as sound card, media player, etc., the user could playback the information and absorb it more easily and vividly than otherwise in the form of textbook. This, to some degree, could be called multimedia education. But it is not what we want to emphasize.

The amount of knowledge and skills that needs to be learned in the information-evolving world is growing rapidly, and often becomes obsolete quickly. In the mean while, it is not practical for the people from different organizations to come together for the same courses, for both the physical and financial reasons. As a result, the need of distributed delivery of education and training courses is highly emphasized recently. The multimedia education makes extensive use of image, video and sound integration and computer networks facilities to supplement and enhance the traditional teaching methods so that the learners could grasp the contents of the courses much more easily.

By taking a look at the evolution of the learning behavior of human beings, we could find that some aspects are in the process of shifting, especially after multimedia technologies were adopted to enhance the education. The learning material and learning place are no longer restricted to the printed textbooks and the traditional classrooms. The role of the students has

been changed from the passive knowledge-receivers to more active, communicative players. The role of teachers is also shifting: from knowledge deliver to knowledge promoter, collaborator and creator of the education environment. Figure 2.2 shows us the generic architecture of the telelearning system, which is similar to that of the multimedia news system [WRK96]. It is mainly composed of courseware production center, courseware database server, courseware author,

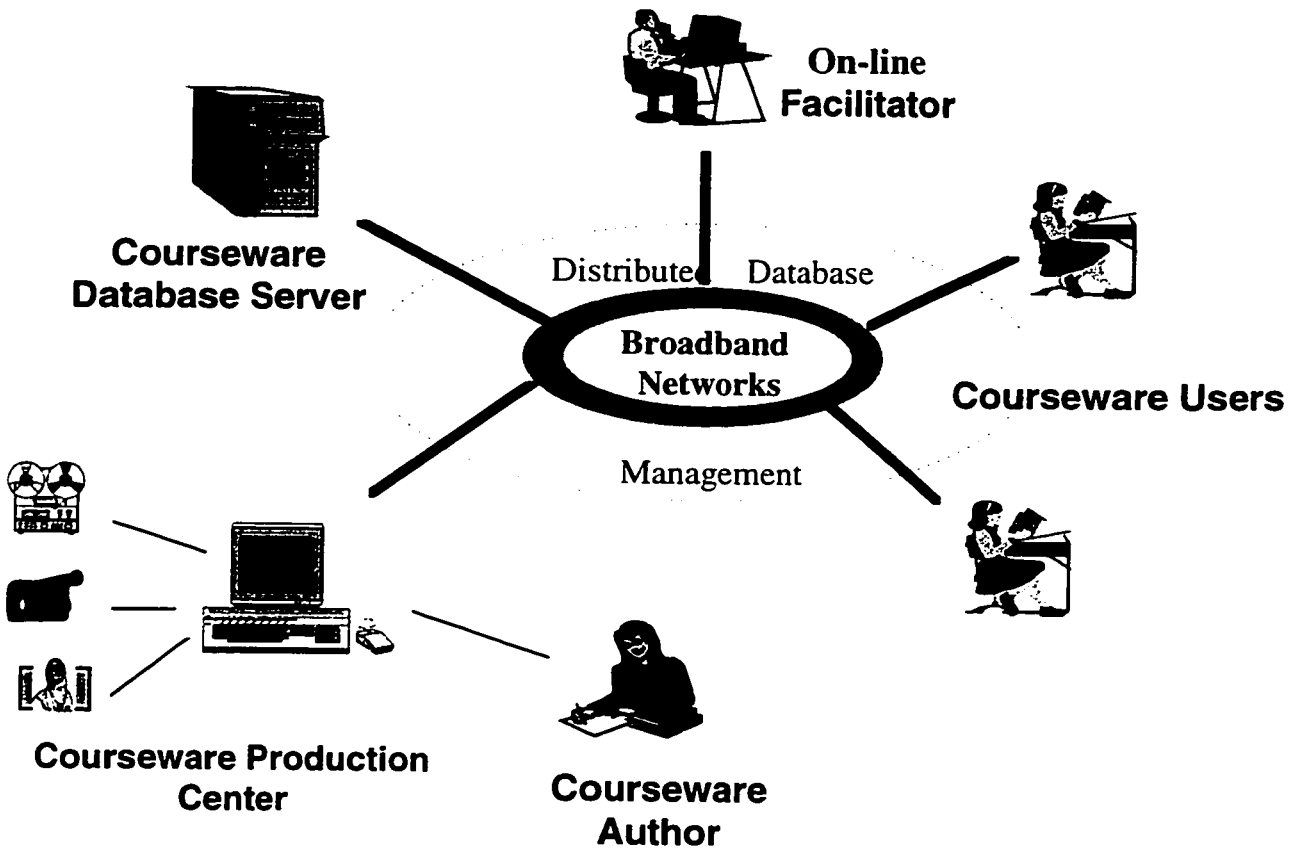


Figure 2.2 Generic Architecture of Telelearning System

courseware facilitator, and courseware user workstations, all of which are connected by broadband networks and a distributed database management system. In order to achieve the goal

of interactive media integration in such a telelearning system, an interactive multimedia document architecture has to be adopted to reorganize the learning material in a more lively and natural way for people to get knowledge from. It could support interactive multimedia document playback as well as hypermedia-styled choices. As a result, the learners are able to learn knowledge more actively and non-linearly by controlling the learning process under such an active learning environment.

2.5 Summary

In this chapter, we discussed some issues of multimedia database systems, especially distributed multimedia information systems, and several multimedia information applications, including multimedia conferencing system, multimedia news, and multimedia telelearning systems. Though these applications are from different domains, they share some common characteristics: the major components of these systems are almost the same. They are mainly composed of production server, database server, graphical user interface, and high-speed networks.

One of the real challenges of multimedia application development lies in the integration of various types of media on heterogeneous platforms. The developments of media processing techniques, high-capacity storage techniques, and broadband networks have provided the necessary vehicles to convey the multimedia applications. In order to make the abundant information available to end-users, a powerful graphical user interface is highly required. Otherwise, no matter how rich and colorful the information is available there, the users will not be aware of its existence. The components of a graphical user interface might include a

multimedia authoring interface to create and edit multimedia documents, a multimedia query interface to specify and retrieve users' interested data from the repositories, and a presentation interface to playback the multimedia documents in the temporally synchronized manner. Although multimedia data involves complicated spatial and temporal relationships which makes them more difficult to deal with compared with the traditional data types, a user-friendly and powerful user interface can reduce the complexity of the multimedia an end user might feel about and make the multimedia information easier to access. In the next chapter, we will discuss issues that are involved in the user interface of multimedia database system.

Chapter 3

User Interface Issues

3.1 Introduction

The user interface of any computing application is responsible for human-computer interaction. The purpose of such an interface is to provide a friendly operating environment and explicit functional context for the user to interact freely with computer facilities [BMM94]. The evolution of the user interface underwent the stages from the primitive command line driven, to today's highly graphical user interface. In 1960's, when the computer was at its early stage,

most computer systems made use of the typewriter-like Teletype (TTY) terminals, which used paper as the display medium. Shortly after, people designed cathode ray tube (CRT) monitor to display text and graphic. The Teletype (TTY) prototype for computer display is the basis for the command-line screen display, such as MS-DOS and UNIX operating system, which is still in wide use nowadays. The command-line-based interface is easy to develop, powerful, and flexible. Since it requires only keyboard entry, it is error-prone, and quickly becomes tedious to user to some novice users. It also requires the user to remember the complex syntax of each command. Although people were eager to come up with ideas for more interactive user interface designs, the electronic and computer technologies at that time hindered some bold thoughts. Not until 1970's saw the conceptual foundation for the current graphical user interface developed at Xerox's Palo Alto Research Center (PARC).

The graphical user interface could also be called direct manipulation interface, since the user uses the mouse as a pointing device whose movement corresponds to the cursor movement on the screen. There are two major factors influenced the design and development of modern graphical user interface: the direct manipulation of graphic objects on the display screen, and the creation of appropriate metaphors which could help the user better understanding the computer systems. A well-designed graphical interface sets up consistent and predicable behavior of the system for the user, with the help of the metaphors drawn from the world of everyday experience [SCB92]. As the benefit of such an interface, users come to treat the on-screen representations as if they were real, manipulative objects like physical documents, buttons, and tools in the real world. The interface prototype designed at Xerox PARC in 70's built the foundation of the visual and functional models of the current user interface. It is no exaggerate to say that the research

work done at PARC is the ancestor of the current Macintosh graphical interface, Microsoft's Windows, and various X-window interfaces on UNIX platforms, such as Motif and Openlook.

The database system is one of the most important applications of computer system. Therefore, the user interface for the database system should possess the common characteristics of the user interface of any application, as well as its own attributes.

3.2 Functionalities of user interface for database systems

A database system is an electronic data repository. The purpose of the database system is to keep record of useful information and make them available on users' demands. Database technologies have great impact on the growing use of computer systems. In other words, computer and database have been bound together as a whole entity. It is no doubt that databases are playing a more and more critical roles in almost all fields involving computer technologies, such as business, education, and engineering, etc. From this point, the usability of a database system is the most important issue of all. The usability of a system reflecting to users is how easily and effectively users can communicate with the system to attain some tasks. Since the user interface is the sole medium responsible for interaction between users and database system, the design of an interface with the property of high usability should be given priority. The factors influencing the usability of a user interface include ease of use, simple context, powerful functionality, efficiency, ease of remembering, and user friendly, etc.

In designing a user interface acceptable by the end user, the following general guidelines should be taken into account:

- **Simplicity**: The user interface is for both experienced users and ordinary users. Therefore, it should be designed with a minimal cognitive load from the user. Piling up to much information and functionality on the interface to the user might confuse and distract the user's attention. The ordinary user should be able to grasp it within a reasonable period of time without feeling tiresome about it.

- **Availability**: The database stores various types of information. The user interface should have the ability to support the user to explore the contents of the database to the greatest extent so that the resources provided by the system could be utilized efficiently.

- **Consistency:** The designing methodology throughout the user interface should be consistent in order to make it easier and more predicable for the user to follow. This also helps to achieve the goal of simplicity of the interface, since it reduces the user's efforts of getting familiar with it, which might take longer time on an inconsistent user interface.

- **Fun**: The user interface should attract the users' interests and make them feel fun on playing with it. This will encourage more people to make use of the system, which is one of the most important concerns of any application.

What mentioned above are the general guidelines applying to the design of any user interface. However, for a specific application, even a specific topic within an application, it may have its own unique requirement. For example, the multimedia telelearning application might

involve many areas and topics, such as computer engineering, literature, education, and business, etc. Each particular area has unique attributes that distinguish it from one another. For example, the business emphasizes on financial analysis and speculation, policymaking, and so on; while the engineering emphasizes on technology evolution and lab works. The user interface designers for telelearning should keep these factors in mind and reflect them on the user interface.

No matter what type of application a user interface is for, it should possess some functionalities in common to other applications [BMD92]. The general high-level functions required on the interface can be outlined as follows:

- **Query facility**: The volume of information stored in the database is very large. It is impractical and impossible for the users to filter their interested materials from the voluminous repository just by browsing through one item after another. This would make users feel bored and finally lose confidence to the system. The query facility for the database is the tool provided to the users to interact with the system. The users could make use of query language associated with the system, such as SQL-like query language, to specify queries to the database, based on some criteria, such as keyword, category, etc. The multimedia database integrates various types of media, which makes query specification more complex than the traditional system. Thus, a more powerful query facility is required to communicate with the system.

- **Browsing facility**: Browsing refers to both browsing the content of the database and browsing the query results presented to the users by the system. The related materials retrieved to the users from the database as the result of a query might still be in great

amount. The users could do finer tuning to the result with the help of the browsing facility. An alternative is to specify further query on the previous query result. The matching documents retrieved as the result of the initial query might still contain too much information. In order to filter out more irrelevant query result, the user could further issue query specifications on the initial query result. We call it the recursive query. For some casual users to the system, they might just want to browse through the database briefly, similar to surfing on the internet, to find out what might be in the database. Thus, an interactive browsing facility is necessary.

- **Presentation facility**: How to present the multimedia document to the user on the interface is another key issue. The problem lies in the temporal synchronization of different media. It is unacceptable that playing back multimedia document without synchronization.

- **Others facilities**: Though other functions, such as customizing the system settings, could be taken into account as an interface issue, they are not within our concerns in the thesis.

3.3 Query Facilities for Database Systems

Acquiring document through navigation or browsing is effective only for small information system. For a large and mature database system, information retrieval through queries is crucial. It is obvious that a too general query might yield lots of items and a too

specific query may retrieve almost nothing. Thus, an ideal query facility combined with users' experience may bring us efficient system performance.

The query facilities provided by the database system can be classified as follows:

- **Natural language query facility**: This type of query interface accepts inputs written in English or other languages and attempts to understand and interpret the meaning carried by the natural language. If the interpretation is done successfully, a high-level query corresponding to the natural language request will be generated, and it will be passed on to the query engine for query processing. Otherwise, the user will be prompted to further explain the request.

- **Menu-based query facility**: A menu-driven query interface provides the users with a list of options through which they formulate the queries. In this way, the query is composed step by step by selecting system-provided options, which relieves the users from the burden of memorizing the syntax of the commands and query language.

- **Form-based query facility**: Instead of providing menus, this type of query interface presents forms to the user. The users can fill out either all of the entries in the forms or some of the entries and let the system use default values for those unfilled entries [CFU80]. Many database management systems have special languages, which is called forms specification languages, to help users to specify entries in such forms. Some systems also have utilities that define a form by allowing the user interactively construct a sample form on the screen.

- **Graphical query facilities**: A graphical interface always displays some visual facilities such as diagrammatic database schema, special-meaning icons [KDA93] [DAM93]. The user can directly manipulate the visual objects with the help of a pointing device, such as a mouse, to compose the query. The graphical query interface is considered to be the user-friendliest interface for the reason of its straightforward nature. Therefore, in order to meet the requirements of users, especially those users with less computer experiences, more and more systems provide graphical user interface.

3.4 Query Languages for database systems

The graphical interface is the closest layer of the system to the end user. But the graphical interface itself is not able to communicate with the database system directly. What the system can understand is a high level language, which is called query language, situated between the graphical interface and the system. There is a query parsing engine, which performs like a pipe, taking the input from the graphical interface and interpreting it into the format of the query language, which can be recognized by the system. The user could also directly make use of the query language to interact with the system without the help of the graphical interface and parsing engine. Therefore, the query language is such a high level language that can be used by the users, or by the parsing engine to communicate with the database.

A query language can either be in the form of a proprietary language specific to the given system or be embedded into some other programming language. No mater in what format it may exist, it is normally composed of two subordinate data languages: a data definition language

(DDL), which supports the definition and declaration of database objects, and a data manipulation language (DML), which supports the manipulation or processing of the objects stored in the database. One particular example of such data language is Structured Query Language (SQL), which is the standard query language of the relational database system [ESL89]. Since we are more concerned about how to efficiently get access to the information in the database, we will emphasize on the query functionality of the query language, which belongs to the data manipulation sub-language. In other words, query specification is our major concern.

3.4.1 Query on Traditional Database

3.4.1.1 SQL

One of the basic tasks of a database system is retrieval of the information stored in it. In order to retrieve desired information from database systems, the user has to be provided with a high-level query language with which the user can make his/her query specifications clearly. The majority of current database systems are both relational and SQL systems. Originally spelled SEQUEL, SQL was first defined by IBM Research Laboratory, and has been adopted as the standard interface to relational systems by ISO and ANSI [ERN94]. SQL is a combination of two sub-languages: data definition language (DDL) and data manipulation language (DML). The principal DDL operations are as follows:

- CREATE TABLE CREATE VIEW CREATE INDEX
- ALTER TABLE
- DROP TABLE DROP TABLEDROP INDEX

With regard to DML, SQL provides four statements: SELECT, UPDATE, DELETE, and INSERT.

SQL is both an interactive query language and a database programming language, which means any SQL statement that can be entered at the command line can also be alternatively embedded in a application programmed with some specific programming language, such as C, Tcl, etc.

3.4.1.2 Query-by-example (QBE)

As mentioned above, SQL possesses a fixed syntax. As a pre-requisite to use it correctly, a user has to keep in mind the complex grammar of SQL. From this point of view, it is not user-friendly and efficient enough to interactively deal with the system. As a solution to the problem, people developed user-friendly interfaces situated between the user and the standard SQL. Query-By-Example (QBE) is one of the solutions. Its syntax is intuitively very simple, because it is based on the idea of making entries in tables instead of writing linear statements.

3.4.2 Query on Multimedia Database

3.4.2.1 Introduction

Query on multimedia database is much different from the traditional database query for the reason of its complex media types and large volume of data. The query language of traditional database only deals with exact-match queries on alphanumerical data types. This might be extended to support multimedia databases that are based on metadata and annotation of

multimedia data [OET93] [JRA94]. But information retrieval from multimedia database should not be restricted to the means of keyword search. The real challenge of query on multimedia database lies in the content-based information retrieval, which requires fuzzy query on database instead of the exact-match query on traditional database.

Generally speaking, three different kinds of multimedia query methods have been proposed for multimedia database system: text-based query, visual query, and content-based query. Text-based query is the easiest and most commonly used method [LDP93]. The fundamental of this method is to use keywords to associate with image or video segments. Another reason to use text-based query is that the query can be easily formulated with the SQL-like query language. There are limitation and drawbacks for text-based query due to its simplicity nature. The biggest problem is the loss of information. It is impossible to illustrate the features of an image or video stream only through the means of keyword. Many important attributes, such as temporal and spatial relationships, are hard to enumerate.

Visual query is helpful for the users to formulate complex queries. The visual query system provides the user with some visual samples to choose from to formulate the query. What the user needs to do is to select the interested samples from the candidates, and the system will retrieve the matched information from the database. One example of such kind of system is IBM's Query-By-Image-Content (QBIC) image database system [NIW93].

Content-based query is the most challenging approach in terms of data indexing, pattern searching and its access structure [SSZ94]. For example, the content of an image is usually expressed by its features, the attributes of objects in the image and relationships between objects.

Content-based query on an image database allows the user to specify the query by the semantics of the image including color, shape, texture and spatial relations of objects.

Both visual and content-based queries are designed to support fuzzy query. Most multimedia systems combine the three query methods together to help the user get access to the database easily.

Up until now, there is not a standard, which is similar to SQL as a standard query language for traditional database, for query specification for retrieving information from such a multimedia database. Researchers have designed quite a few query languages for a variety of applications, for example, medical, financial, geography, environment, biology, and so on. Most of the query languages are application specific, and very few generalized systems have been developed.

Querying on various types of information is based on the data representing the contents of the information stored in the database. Within multimedia environments, a variety of objects that have different properties and forms of appearance has to be maintained. Textual descriptions are still the principal technique for the purpose of indexing and retrieving multimedia objects. More multimedia-oriented methods, supporting automatic, media-adequate techniques, are receiving more and more attentions. For example, icon-based environment for image database querying, content-based image and video indexing and retrieval, etc [YAK94] [GYZ94]. What follows, we will discuss some types of database systems and how the query specifications could be formulated to such databases.

3.4.2.2 Image Database

Images are becoming an important asset, and managing them for efficient retrieval has been an challenge to the database community. To handle the large amount of unstructured image information, such systems must possess more functionalities than do the traditional database systems, which typically include: (1) the capability to analyze the image to extract key features such as object, color, and shape from the image; (2) the support for feature-based indexing; (3) the support for content-based retrieval; and (4) the ability to handle fuzzy queries. Existing image database systems can be classified into categories according to the image features used for indexing. Figure 3.1 shows the classification of image databases.

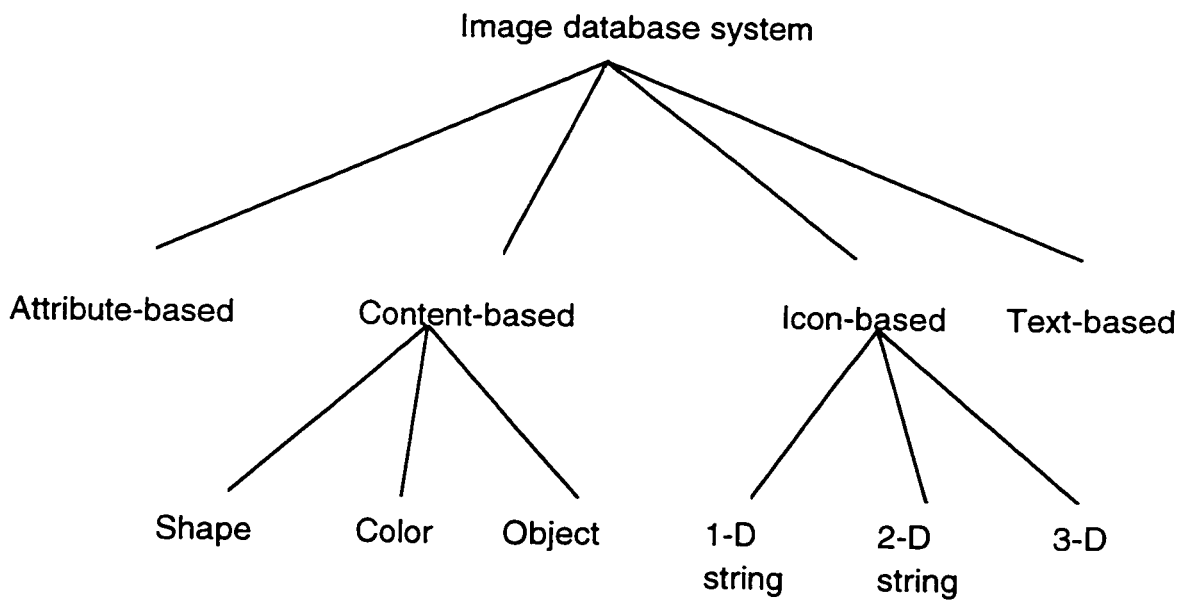


Figure 3.1 Classification of image database systems

3.4.2.2.1 Attribute-based

The most straightforward approach is to treat the images as large objects. An attribute-based image database system can be designed on the basis of the extension from the conventional database management system with the capability to handle the large objects. Access to the unstructured images is achieved through the structured attributes of the image [OVM95]. In this approach, no special effort is required to design the organization techniques, indexing mechanism (examples are B^+-trees and inverted files) and query processing method. An example of such kind of approach is Query-by-pictorial-example (QPE). In this approach, a relational database system for images was designed. It can be considered as an integrated database system interfaced with an image understanding system that consists of an image processing system and image recognition system. Structures and features are extracted from images. These extracted descriptions are then integrated into a relational database, while the original 2D images are stored in a separate image repository. Therefore, queries concerning images can be transferred into query the relational database. QPE adopts the tabular form of query formulation, which is similar to that is used in QBE. The basic idea is that the user formulates the query by entering a possible answer in the appropriate place in an empty table that is generated by the system. However, this approach lacks the capability to handling the more user-friendly queries, such as content-based query.

3.4.2.2.2 Text-based

Developments in text-retrieval research have prompted many researchers to apply the concepts of Document Retrieval System to image database systems. For each image in the

database, there is a text document associated with it, which contains the attribute features extracted from the image. Ideally, the work of extracting features from images should be done automatically with the help of the computer technology. But, the current stage of image processing provided by the computer technology is not sufficient enough to meet the requirement of feature extracting. Therefore, manual or semi-automatic methods need to be used as an amendment to the automatic method. Users could use free-text method to issue the their query specifications. The system compares the free-text with the documents stored in the database, and the matched images will be retrieved. Based on free-text queries, the image database system attempts to provide "content-based" functionalities by manual description of the image and treating the image description as a document. However, besides unable to facilitate content-based queries, there are other limitations to the approach: (1) a free-text description of an image is highly variable due to the ambiguities in the natural language associated with annotating images with text and the different interpretations of the image; (2) The image is so rich in semantics that the text is not sufficient enough to describe it completely; (3) The vocabulary of the person creating the index and the user's may not match.

3.4.2.2.3 Content-based

Researchers have proposed means of content-based approaches from several directions, and the difference among them lies in the content information which is being indexed on. The image content referred here is restricted to such parameters that can be feasible to compute by the capability of computer vision at current stage. Examples of the content used include color, shape, sketch, and texture [LRW94] [GYZ94]. Those contents, which are currently beyond the reach of pattern and image technologies, are not dealt with. Examples are the more complex

semantic descriptions such as "dog', "house", etc. Content-based searches have important distinctions compared to traditional searches in such a way that they are approximation and as a result there is no exact match. Images are typically sorted by similarity to the posed query and usually only a portion of top-ranked images is display on the screen. The user could use browsing facility to select the desired image from the collection of the query results. Generally, there are three major steps for content-based query: (1) Database population, which means the process of images being imported into the system. (2) Image representation computation, referring to that the image features and attributes are computed and stored in the database. (3) Query process, which means that the user creates, navigates and refines the query specifications until the user is satisfied with the result. It supports two ways of specifying a query by a user. One is direct query: the user specifies the desired color, shape, sketch, and/or texture directly, e.g., by picking colors from a palette on the screen. The other one is query by example: the user can choose one of the displayed images as a criterion, and ask for images similar to it.

3.4.2.2.4 Icon-based

Retrieval images from databases can be effectively performed through visual icon-based systems. This approach has been received much attentions recently and quite a few ways were suggested by researchers. In such a system, icons for querying images are defined through a computer system by the users. They maintain a symbolic relationship with objects in the real world. Either 2D or 3D structure could be used to build the iconic environment according to requirements of specific applications. For the images that are not complex, the 2D icons can reach a satisfactory outcome without causing any ambiguities. When dealing with complex images, a single 2D-visual scene may correspond to many arrangements of objects in a real-

world scene resulting from the 3-dimensional characters of real world objects. This drawback could be overcome by using icon overlapping, also could be called 2.5D icons, to reproduce the missing third dimension. The 2.5D icon, unlike 3D icon, doesn't carry the exact spatial information of the 3rd dimension. It just carries the relative spatial relationship between each pair of icons, such as a icon is on top of, or at the bottom of another icon.

However, this improvement works well only for simple 3D scenes. For relatively complicated image queries, 3D icons have to be adopted to express the meaningful spatial relations among objects in the scene [BAD93]. Using 3D icons to formulate the query is also in accordance with the typical behavior of the users, who express queries on the basis of the view of the scene they have in their minds, which are 3D. In such a system, image contents are represented referring to the spatial relationships of objects in the imaged scene. Objects are represented through their Minimum Enclosing Parallelepiped (mep) and by their axial plane. The mutual relationships between pairs of objects are expressed by a set of formulas with reference to a coordinate system. Two types of formulas are taken into account to describe the positional and directional relationships between objects. Positional formulas deal with relationships between mep orthogonal projections over an axis of the reference system, and directional formulas consider relationships between axial planes of the objects, which are collinearity, parallelism, and non-parallelism. Corresponding to the 3D representation of the objects in images, a collection of 3D icons, arranged into an inheritance hierarchy, is stored in an object-oriented database. When expressing the query, what users need to do is only pick the related icons and put them together to form a virtual scene which represents their desired real-world scene. Boolean operators can also be used to formulae more complicated query from two separated queries. It is intuitive and user-friendly, since the user experiences a mental shift from the

sensation of interacting with schematic representation of reality to the sensation of directly interacting with reality.

3.4.2.3 Video Database

Video has become an important element of multimedia computing and communication environments. The video can be considered as composed of a large number of image frames. It can only become an effective part of multimedia component when we can use it with the same facilities that we currently make use of text. The video technology has developed thus far as technology of images, and little has been done to help us use it effectively because the effective use of its content is still beyond our reach. We still have problems of manipulating video efficiently to perform such tasks as browsing, authoring, and retrieving video content. Most of current video applications are based on three simple video functions: record, access, and replay.

In a video database, several functions should be possessed by the database management system: (1) Parsing, which segments the video stream into generic clips, also called shots. A video shot can be indexed with a semantic description using existing knowledge-representation techniques. (2) Indexing, which tags video clips when the system inserts them into the database. (3) Retrieval an browsing, where user can access the database through queries based on text and visual examples or browse it through interaction with displays of meaningful icons.

The video stream can be segmented into segments, shots, and keyframes. The key issue here is to make use of the meaningful properties embedded in the video media to effectively parse the video stream [SSZ94]. Once the video has been well parsed and indexed, the video query could be reduced to image query combined with temporal query.

The temporal relationship is an important characteristic of video streams. Many existing digital video systems rely on the traditional view of video as a linear temporal medium. They do not take full advantage of either the logical structure of the video or the hierarchical relationships between video segments. In [WED94], Ron Weiss and David Gifford introduced an approach called *algebraic video* , in which they use a set of basic operations on video segments to create a desired video stream. The goal of designing video algebra is to provide a high-level abstraction that models complex information associated with digital video data and supports content-based access. The video algebra consists of operations for combining and expressing temporal relations, for defining the output characteristics of video expressions, and for associating descriptive information with these expressions. The video algebra operations are divided into following categories: (1) Creation, which defines the construction of video expressions from raw video; (2) Composition, which defines temporal relationships between component video expressions; (3) Output, which defines spatial layout and audio output for component video expressions; (4) Description, which associates content attributes with a video expression. Here, video expression is used to describe video segment. The most primitive video expression involves the creation of a single-window presentation from a raw video segment. Compound video expressions can be constructed from single ones using video algebra. The user can query the desired video streams by describing its attributes.

3.4.2.4 Audio Database

Audio is another important component of multimedia applications. Music, speech, sound and other types of voice signals could be included in the audio domain. Audio can exist either separately or associating with other media such as video. Audio as a track within the video

signal can provide a very rich source of information to supplement the understanding of the content of the video. For example, the sound of a bomb explosion may indicate the scene of war. This kind of information can be used as an auxiliary tool in video segmentation and indexing. Furthermore, we can decompose the auditory perceptions into sequence of objects, just as we can do on visual perceptions. In this way, we can characterize these objects, track them across an auditory stream, and use this important information for segmentation and indexing. Users may formulate their queries by the content of the audio, either in textual form or just speaking a sentence. The later one needs the help of speech recognition system to translate the voice signal into text form, and then match it with the sound-index within the database.

Automatic speech recognition has gained a significant progress in recent years. All speech recognition systems use either time domain or frequency domain approach. They can be classified along a number of standard dimensions, for example, speaker dependent vs. independent, discrete vs. continuous (whether or not the user needs to separate individual words by short silence), vocabulary size, and so on. The availability of speech recognition technology has encouraged the development of prototypes that merge speech capability into a general-purpose computer interface [KAC94]. It can make the user interface more intuitive and user-friendly in such a way that it makes such query specifications much easier that are difficult to realize on a graphical interface.

3.5 Related work

Similar to mono-medium database retrieval, multimedia retrieval rests on the exhaustive search of document contents to retrieve those documents that satisfy the specification of a query.

It has not received as much attention as single information retrieval in the past. However many attempts have been made concerning query on multimedia database. Some of them are extended from the standard SQL language, while some others are developed completely from new ideas.

Generally speaking, there are two approaches of query specifications on multimedia databases: the static query and the dynamic query. The former approach takes into account the static attributes of media, such as color, shape features of image, keywords of text. The latter approach not only considers the static attributes, but also puts emphasize on the temporal relationships on media types, such as the beginning time of a video stream, and the relative time difference of a video segment and an audio segment. Thus, it gives the user more capability to query on the basis of the content of media. Most of the multimedia database systems base their queries on static features of media types.

The IBM's QBIC (Query By Image Content) [NIW93] might be the most well-know multimedia database. In this project, they use the image content, such as color, texture, sketch, and shapes of image objects, as the basis of the queries. A major challenge in the queries is to determine a set of attributes to describe the content of an image and formulate an index into the image collections. It benefits from the great deal of work done in machine vision on feature extraction and similarity measures. QBIC supports two ways of specifying a query: "direct query", where the user specifies the desired ***color/shape/texture/sketch*** directly, for example, by picking colors from a color palette on the screen to specify the query; "query by example", closely related to the concept of relevance feedback, which means that the user can choose one of the displayed image and ask for images similar to the selected one.

[ROB94] introduces a Video-on-Demand system that has the ability to access the video of interest in a large video database. The database is mainly composed of a relational database and video file servers (VFS). The database contains metadata and indexes about the video stored in the system, and tracks the location of video material on VFSs. It makes use of four types of indexes to create the meta-database:

- **Bibliographic**: such as title and author of the document
- **Structural**: referring to visual and outline of video clips, such as start and end time, text description of shot and scene, geographic location where the scene takes place, etc.
- **Object**: such as actor and actress, and objects important to the scene
- **Keyword**: referring to the unique identifier of a document

The query language of the VOD system is called POSTQUEL, which is the query language of POSTGRES database and very similar to SQL, developed at Berkeley. The specification of the query "movies produced by Universal in the 1960's in which Elvis did appear" can be expressed in the syntax of POSTQUEL as follows:

```
retrieve (movie.all) from movie in DOCS, bib in VIDEO_BIB,
   dv in DOC_VIDEO

where dv.docid=movie.oid and dv.base_reference=bib.oid
```

```
and bib.ref_org="Universal" and bib.ref_date between ["Jan1
   1960", "Jan 1 1970"]

and bib.vbib_cast contains "Elvis Presley"
```

It is obvious that the system is limited by the domain-specific attributes. Another drawback is that no quantitative presentation control (e.g., playing back some video stream for 60 seconds) is supported.

In [KAT94], a multimedia query language, called MQL is proposed, for specifying and manipulating multimedia information was introduced. The data model is established on the basis of object-oriented concepts. The MQL language is also divided into two parts as in SQL: data definition language and data manipulation language. The data definition language (DDL) is a C++-like language that achieves extendibility and flexibility by employing an object-oriented framework for multimedia information management. The data manipulation language (DML) is an SQL-like language that is used to retrieve multimedia elements from database in several ways. MQL takes into account the integration of different media, and users can use it, whose syntax is SQL-like, to query on multimedia documents to retrieve the interested materials. But it only deals with the conceptual structure of multimedia documents. However, MQL is not concerned with such important relationships as spatial and temporal relationships within multimedia data. Therefore, it can be considered at most as a query language that has the ability

to integrate the static attributes of multimedia data within a multimedia database, not a real multimedia query language.

Only recently, we have seen introduction of dynamic queries on multimedia database systems, which gives more power to users to access the multimedia data.

[ABL95] introduced a video retrieval system that provides flexible ways of formulating the query. In addition to making use of the attributes embedded in image, such as objects and spatial relationships between two objects, it puts more emphasis on the motion features unique to video, including inter-frame temporal relationships, sequential composition, camera and object motions. The queries can be formulated by specifying any combination of camera motions, objects and their motion paths, either domain dependent or independent. An example of query may consist of an object moving parallel to the camera while the camera is panning, followed by a zooming into the object. Therefore, the query can contain various levels of motion complexities in terms of motion query specifications. It provides a user-friendly query builder, which is named MovEase (Motion Video Attributes Selector), to the end user. The user can easily use visual method, such as 3D icons, to interactively formulate the query on the basis of motion features of video data, e.g. camera operation, trajectory of moving object. The query can also be viewed by animating it, such that the user can get instant feedback and can make modifications to the query.

Hibino and Rundensteiner introduced a temporal visual query language (TVQL) [HSR95]. With the help of the language, the user is able to identify temporal trends in video data by querying the relationships between video annotations. Relative temporal queries are composed of two components: relative duration and relative temporal position. Relative temporal position refers to starting points and ending points of temporal relations of events or subsets of

events. The relative duration describes the long (or short) active time of some events compared to others, e.g., student S1 spends at least one minute more on a task than student S2. The primitive temporal operators are visualized graphically with some visual primitives, which are shown in Figure 2.2.

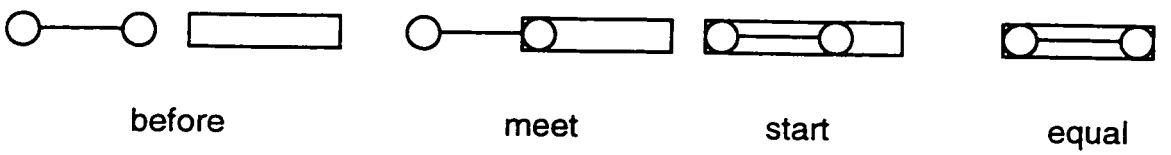


Figure 3.2 Sample of Individual temporal relationships

The user can manipulate these temporal primitives on the graphical query window to specify relative temporal queries, which is more intuitive to specify it in the form of text language.

3.6 Summary

In this chapter, we have discussed the user interface issues and query languages for database systems. The user interface is the front end of any application, and is an important component of the system. To a large degree, it determines the usability of the whole system. A good and user-friendly interface might make users feel comfortably and interacting with the application in an efficient way. On the other hand, a badly-designed interface might let users feel the application to be awkward and hard to grasp. The evolution of user interface underwent stages from the simple command-line driven to highly graphical interface, which occupies the dominant place among the current computer applications.

The query language is the front-end of database system. The purpose of any query language is to facilitate users to communicate with an associate data/knowledge base. An important criterion for evaluating a particular query language is its expressive power. However, the semantic power of query language is limited by the semantic depth of the underlying data model. With the advent of such complicated databases as pictorial, multimedia, much more expressive query languages are needed as tools of interaction between users and databases. The visual interaction is one of the efficient ways to meet the complexity of the database. One reason for developing visual interaction for complex database is that pictorial or iconic communication increases information content. An single icon can replace a great deal of textual descriptions. It also significantly reduces the cognitive demands on the database user.

In the next chapter, we will introduce the interactive query interface for the multimedia database system, called MEDIAQUINT (multimedia query interface), which is under development in our laboratory.

Chapter 4

The Graphical Query Interface for Multimedia Database

4.1 Introduction

The purpose of a database management system is to maintain information and to make the information available on user's demand. A quite important step to a multimedia database system, which contains video, image, audio, text, and so on, is the data indexing. Taking the video as an example, the video stream has to be segmented into structure primitives (scenes, shots, and keyframes) on the basis of the video properties such as camera operation, histogram distribution, and motion vector analysis, which are considered to be the low level video indexing [DEL94]. Another approach, known as high-level or semantic indexing, for instance, aiming at

detecting the semantic meanings contained in the video, relies heavily on object recognition techniques. In [HNK95], an algorithm, which can successfully detect the cuts and some basic camera operations such as zoom in, zoom out, pan left, pan right, tilt up and tilt down, was proposed. Such information is unique and very important contents of video. In this paper, we suppose that all the data in the multimedia database are well indexed and organized, either recognized automatically by computer or entered by hand.

The behavior of an end user of a database system may involve three possible operations: query the database to retrieve user's desired information, control the presentation of the results and browse them, or just briefly browse the database to find out what information might be included in it. Therefore, a friendly graphical user interface with all these functionalities should be equipped so that the user could effectively utilize the resources provided by the database.

4.2 Architecture of the User Interface

Figure 4.1 shows the basic structure of the user interface. As we can see, there are two main branches, query interface and browse interface, connected to the multimedia database user interface. The user specifies queries to the database on the query interface. The translation module is responsible for interpreting the query specifications generated from the graphical query interface into the format of the specific query language of the database, while the query algorithm is utilized to search for the matching information in the database. There is a presentation control module to control the layout of the query results. There might be many relevant materials retrieved as the results of the same query specification. The user makes use of the result browser to select most favorite data and plays them back. The final query result will

be played back by the MediaPlayer [RJK95]. In some cases, the user might just want to browse the database randomly in an attempt to find something interesting. Therefore, a good database browser should also be equipped to the graphical user interface.

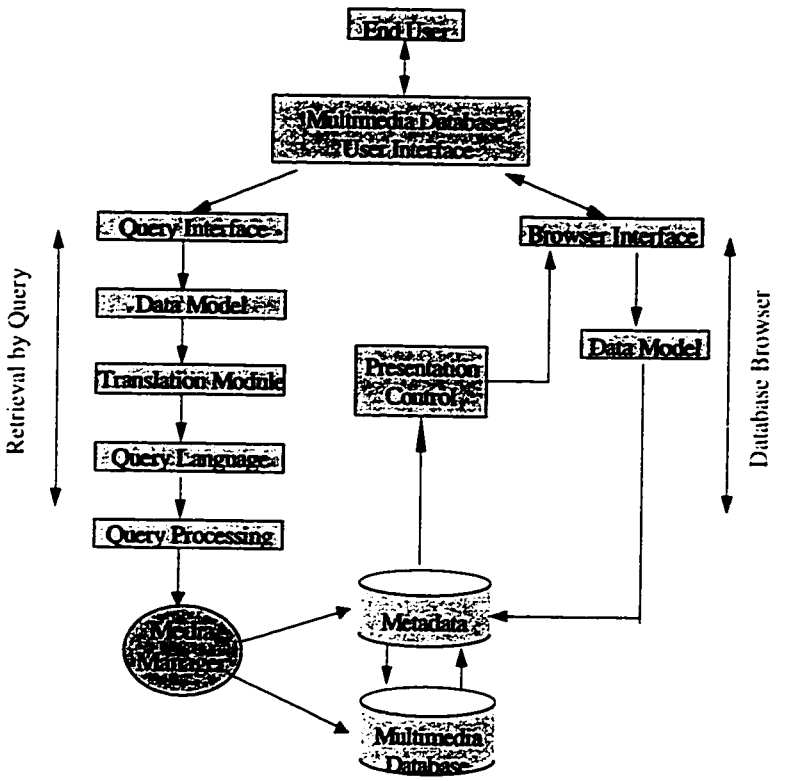


Figure 4.1. The functional architecture of the user interface

4.3 Multimedia Query Languages

In our laboratory, [HIK95] proposed a multimedia query specification language that could be applied to retrieve different types of multimedia documents from a multimedia database. The proposed query language takes into account the spatial-temporal specifications as well as the media specifications contained in a multimedia document segment. By having the corresponding multimedia image in mind, users can construct the query specifications step by step with the query language, just like issuing queries with SQL language on the relational database. If the user has previously seen the multimedia scenario he/she wants to retrieve from the database, it will definitely help the user to better recall and write down the description.

The syntax of a query specification is defined as follows:

```
FIND        X      IN    Doc_List
WHERE       BEGIN       Multisegment-Specification
      [Temporal_Specification]        and/or
      [Spatial_Specification]         and/or
      [Media_Content_Specifications]and/or
      [External_Specifications]
END
```

It is a FIND-WHERE query in which X is a variable referring to the retrieved multimedia documents. Doc-list is a list of documents selected from the database, and Multisegment-

Specification is a description of the multimedia scenario of interest written in the format multimedia specification language.

The multimedia information perceived by viewers can be decomposed into three different types:

- *Temporal information* that specifies the relationships in time between the various media objects, such as those resulting from camera operations (e.g. zoom-in) in a video clip.

- *Spatial information* which specifies the locations of the media objects in space, such as describing a 'person' to be in front of a 'table' in an image or in a video frame.

- *Media information* describing the individual media elements composing the multisegment, such as the key words of a text, and the objects in an image or a video frame.

In addition, interactive multimedia documents may also include unpredictable time events, such as user interaction, reaching a particular program or document state, and programs with unpredictable execution time. These behaviors are usually remembered by the viewers and could be used to enrich the description of a query specification. We name this type of information as external information to differentiate it from the specific media information.

Therefore, a multimedia query is formulated by specifying the various information on the multimedia scenario of interest, as illustrated on Figure 4.2.

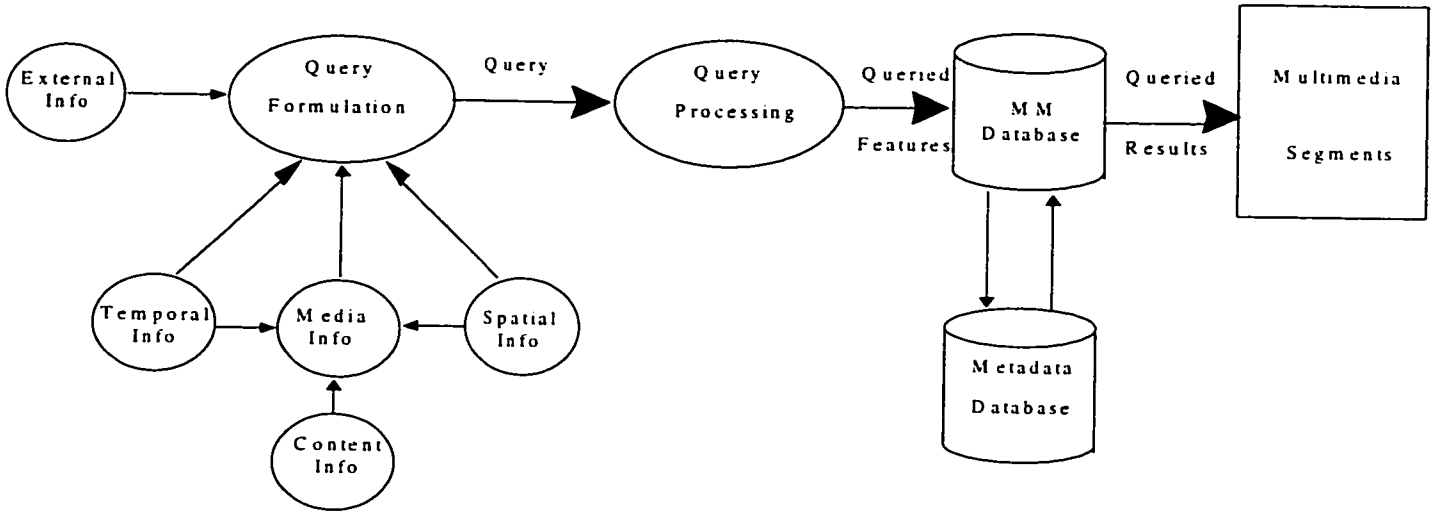


Figure 4.2 Multimedia Query Formulation

4.4 Graphical Query Interface

The multimedia database contains different types of information, which can be grouped into two basic types, either dynamic, such as video, audio, or static, such as image, graphics, and text, on the basis of the properties of the media. There exist complex temporal and spatial relationships as well as other characteristics among and within the various media types. In order to achieve the goal of information retrieval, the system must possess a high level query language and an easy-to-use query interface with highly expressive power, with which the user could efficiently formulate the queries based on the content of data stored in the database.

By examining the human behavior of querying on databases, it could be found that people always decompose complex queries into smaller and simpler steps. Therefore, on our graphical query interface, which is built on top of the multimedia query language, we provide the user with some sub query windows, in an attempt to represent and emulate the querying procedure going on in people's minds. The users could navigate among these windows to construct the queries on different query windows respectively on the basis of different properties of media, and then logically assemble the sub-queries together to formulate the final query. Our approach to the user interface is to use the visual methods as much as possible to specify the complex information embedded in the media, such as temporal, spatial and content information.

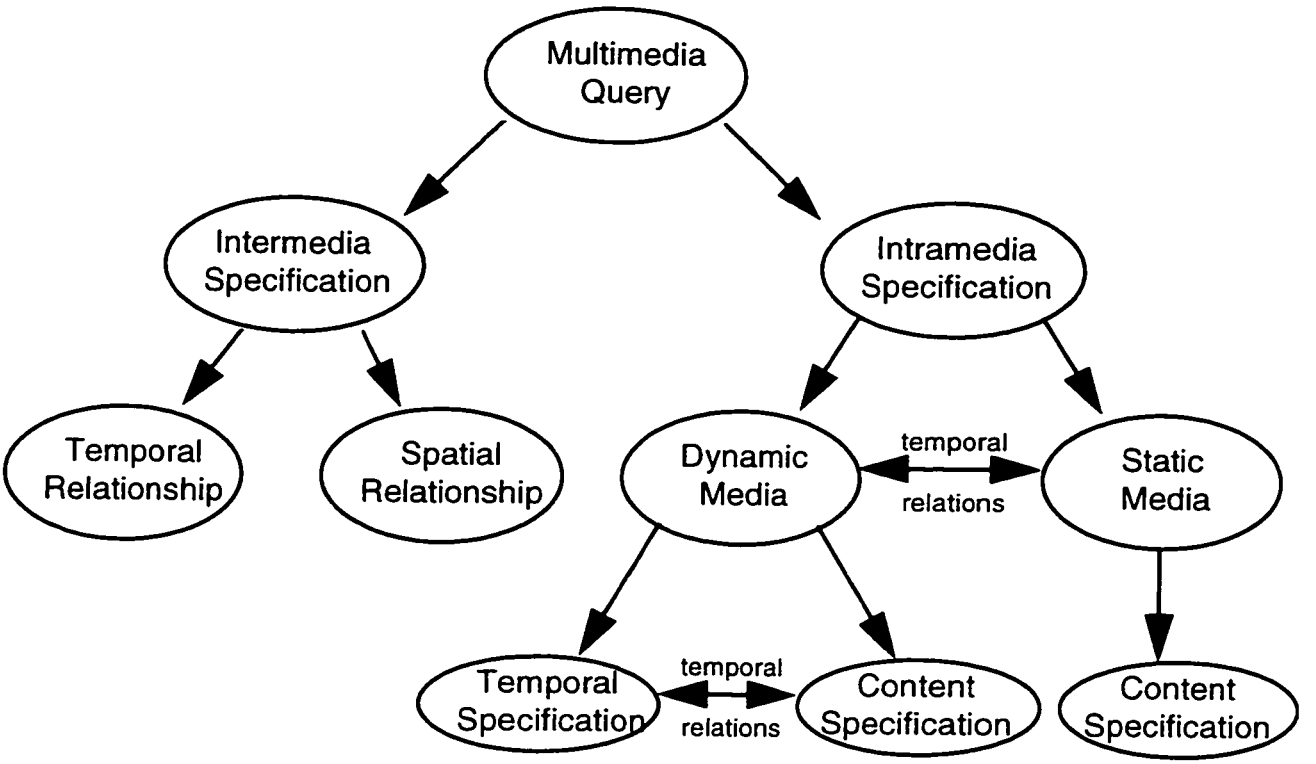


Figure 4.3. Query decomposition

Figure 4.3 illustrates briefly how the query is decomposed on our interface. We separate the query specifications into two major steps: intermedia specification and intramedia specification. The intermedia specification deals with the temporal and spatial relationships among different media types. For instance, how long does a video stream last in terms of seconds, how long is the interval between an image and a video stream, what is the spatial location of an image. The intramedia specification provides the means to specify queries based on the contents of a specific media. Based on the properties of the media, the media could also be divided into two categories: dynamic media and static media. The dynamic media refers to those media whose characteristics are changing with the time. The media falling under this category include video and audio. However, the properties of the static media are not the variants of the time axis. Image, text, and graphics belong to this category.

4.4.1 The Need for Graphical Query

Graphical interface is the trend of today's user interface development. Command line or form-based user interface is not user-friendly, especially for novice users. Before interacting with the system, they have to spend strengths on learning the might-be-complex commands. This might lower users' interests on the application. Taking file and directory manipulation on a computer as an example, there are a file manager and a set of file manipulation commands respectively. Suppose your are a novice computer user, which means do you select to manipulate files? It is safe to say that most novice users will choose file manager, because it is straightforward and easy to use. The same theory also applies to user interface for database system. Even though the system has a highly developed query language to access the data space, it might still lose the potential to become a widely accepted application due to lack of user-friendly interface. Therefore, a graphical user interface is highly needed.

4.4.2 Intermedia Query

The intermedia specification on our interface allows the user to construct the query in terms of the temporal relationships and spatial relationships among different media objects or separate media objects of the same type. We discuss the temporal and spatial specifications respectively in the following sections.

4.4.2.1 Temporal Query

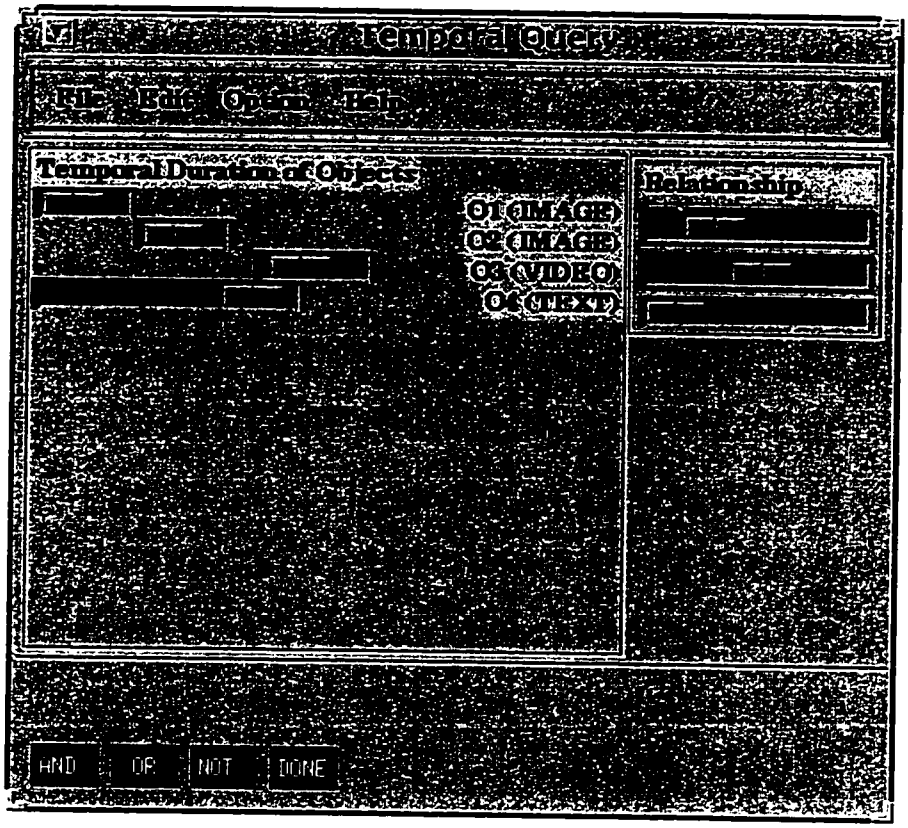


Figure 4.4 Temporal query window

In the temporal query window, as shown in Figure 4.4, it supports the user not only to specify qualitatively the temporal relationship of media objects or between pairs of media objects, but also to specify them quantitatively. For example, how long does a media object last in terms of minutes? Or what is the range of duration between a pair of media objects?

In this window, we use the basic concept of time line to indicate the temporal behavior of media objects. The traditional time line is straight forward, and has the characteristics of visual and intuitive representation of time relationships. It can as well represent the user's concept of

time in user's mind effectively. But one of its shortcomings is that it can only specify qualitatively the temporal relationship, for instance, one object meets before or after the other, but lacks the ability to tell the system the quantitative information. Here, we make some extension to the traditional time line to make it possess quantitative ability.

The horizontal axis in the temporal window represents the time-line. When the user clicks on "Object" button from the "Option" pull down menu on the menu bar, a scale will be created on the left sub-window to represent a media object. The length of the scale is proportional to the temporal duration of the media object. And then, the system will prompt the user to specify which kind of media object he has created, video, image, or text. The length of the scale is adjustable by moving the position of the slider in the scale. There is a number associated with each scale to tell the user the quantitative value of the duration of each media object, in the unit of millisecond, second, or minute. The unit of the scale can be chosen from millisecond, second, minute, or hour so as to meet the requirement of different length of media objects. In this way, the user can specify the temporal relationships more accurately. The scale can also support less accurate query, by just simply adding an operational symbol in front of the number. There are four possible candidates for the operational symbols: =, >, <, any. Combined with the number following the operators, they clearly tell the user the quantitative characteristics of the media objects. For example, "< 10 sec" means the duration of the media object could be any length between 0 and 10 second. Boolean operators such as OR, AND, and NOT are also provided for the user to specify relatively complicated temporal information. For instance, if the user wants to specify such information: 1 min < object_duration < 2 min, he/she could use the temporal window twice, object_duration > 1 min, object_duration < 2 min, and use AND operator to

combine them together. Following the same way, the user can specify more media objects on the temporal window.

What we described above is to specify the absolute duration of each media object. Meanwhile, the relative interval between objects is also very important factor of temporal information. The scales in the right sub-window have the same properties as that of left sub-window, except that they are provided for the user to specify the relative interval between each pair of media objects in the left sub-window. By adjusting the slider of the scale in the right window, the positions of the corresponding scales in the left window will be changed accordingly to reflect the relative time interval of each pair of media objects.

4.4.2.2 Spatial Query

Another important relationship among different media types is spatial. What we are concerned about here is the spatial layout of each type of media. In the spatial query window, we use rectangulars to indicate the spatial relationships. The user could create different rectangulars with different color backgrounds to represent different media types. By putting the rectangulars on desired positions on the query window and adjusting the size of each rectangular, the user could clearly express the spatial layout in such a visual means. The mechanism in the system to calculate the spatial relationships between each pair of objects is based on the projections of the rectangulars to the horizontal and vertical axis. Figure 4.5 shows the spatial query window.

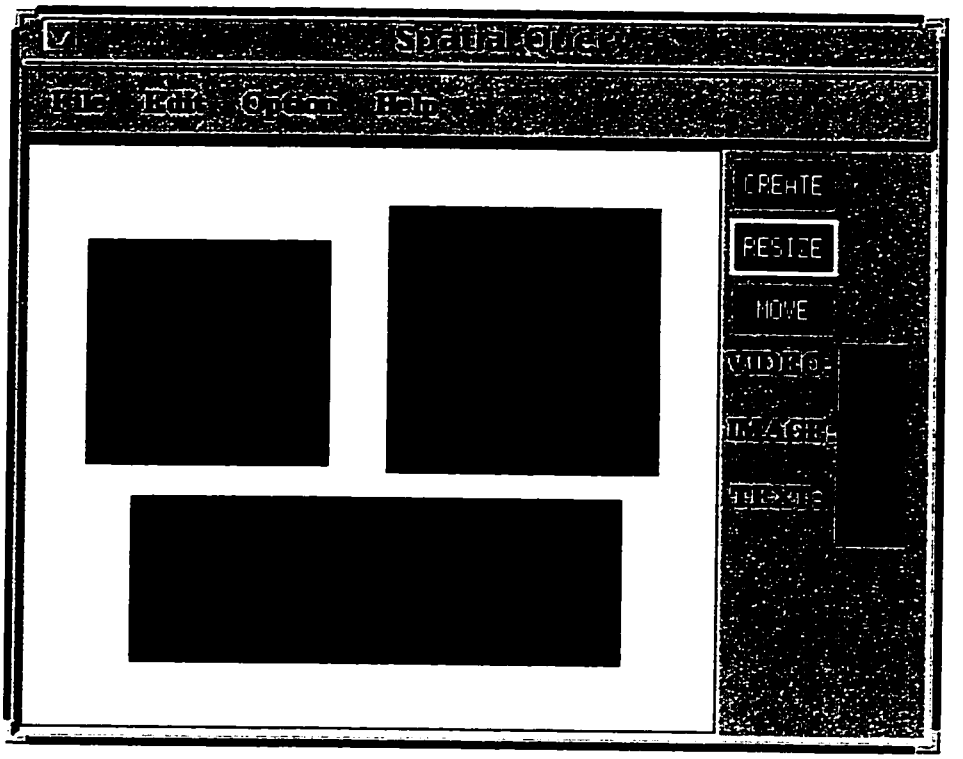


Figure 4.5 Spatial Query Window

4.4.3 Intramedia Query

As we have discussed before, the media can be categorized into static and dynamic media. The static media, such as text and image, contain relatively simple and less contents compared to their counterparts, such as video and audio. Let's take a look at the simpler one first.

4.4.3.1 Static media

The text deals with alphanumeric data. It is the most commonly used data type by the traditional database systems. There has been a suite of well-developed query mechanism related to it. We use the conventional keyword searching for querying texts. Though there are some drawbacks for keyword searching, such as keyword mismatching, it is still the most straight

forward, effective, and widely used method. We will not discuss it in detail here. Instead, we focus on querying the image.

Although image could be thought of as a static medium, there are lots of visual features associated with it, either in low-level (color, texture, shape, etc.) or in semantic level (visual object in a picture), which makes the query specifications on image complicated. The meanings contained in a picture might be more plentiful than those of a hundred words might. It is almost impossible to use the text query to specify the visual features effectively. In other words, if the user is in an attempt to describe the content of a picture in text, the user has to spend great strength to achieve the goal, which is a very tedious work. Although the content of the picture could be successfully parsed into the text format, such a parsing is not unique, and tends to be infinite, resulting from the different points of view of different people. This makes the query algorithm much difficult to handle the various possibilities, and directly influences the performance of the system.

Our approach for the image query is to use visual query methods as much as possible to retrieve the information the user is interested in. The contents of image involved in the querying refer to the features obtained by image analysis and processing, either automatically or manually. Such features include color, texture, visual objects, and subjects of domains. The system allows the user to construct the queries on the basis of these features.

4.4.3.1.1 Outline query

The image stored in the database may involve a large number of application areas. When the pictures were entered into the database, they were classified into different domains on the

basis of their contents. The outline query is concerning the keyword description of the domain. The domains are classified into hierarchical structure, similar to the class classification in the object-oriented programming. When the user selects the outline query option, a hierarchical structure pertaining to the subjects of image is displayed. The user can navigate up and down among the hierarchy and select his/her favorite subject. It sorts out the uninterested domains, and thus reduces the searching range of the system.

4.4.3.1.2 Object Query

We use the icon-based method in the object query window for the user to construct queries. The iconic query method is considered to be intuitive and user-friendly, since it gives the user the sense as if he/she is somehow directly interacting with the reality. The user constructs the query specifications according to his/her memory representations of the real world. Only those salient objects with prominent visual effect in the image give deep impressions to the user.

To construct the query, the user unnecessarily has to see the real world scene in advance, but having watched them will certainly be helpful for better constructing the query. By using these important objects, the user can semantically specify queries without any difficulties.

In the object query window, we provide a hierarchy of icons, which are classified by using the object-oriented paradigm. Each icon maintains a symbolic relationship with the corresponding object in the real world. The user can navigate freely through the icon hierarchy. When a category of icons in the hierarchy is selected, the icon instances belonging to the category will be automatically displayed to the user. The user can pick up interested icon and put

it to the appropriate location on the query space. The size of the icon on the query space is also adjustable easily with the operation of the mouse, so that the user can clearly specify the size information of the object as well. By arranging the size and relative location of the selected icons on the object query window, the user could clearly express the content of the picture to be retrieved from the database in his/her memory representation.

The user could also choose an option to pop up a property window, with the name of "Additional Information", associated with each selected icon. It is offered for the user to supply supplementary information to the icon, since sometimes the visual icon itself can not convey enough meaning the user wants to express. For example, the user is interested in the scene containing Parliament Hill, but there is not the exact icon in the icon hierarchy to represent it. What the user can do is to select an icon representing a building, and then write down "Parliament Hill" in the property box. In the mean time, through this way, we can add more semantic information to icons, and let them have new meanings.

But the icon hierarchy does not and cannot have the ability to represent all the objects in the real world. The window also provides users with drawing tools to sketch some shapes or objects that cannot be found in the icon hierarchy, and they can also be saved in the system for the future query.

Figure 4.6 shows the object query window. The left part of the window is the querying space on which the user could put icons. The meaning of the querying space showing on the window is that a tree is beside a house, and a car is in front of the house. The upper-right part of the window displays a portion of the icons under the category of transportation. The lower-right part of the window is a collection of operators provided for the user to directly manipulate the

icons on the querying space, such as move, resize and delete, and construct more complex queries.

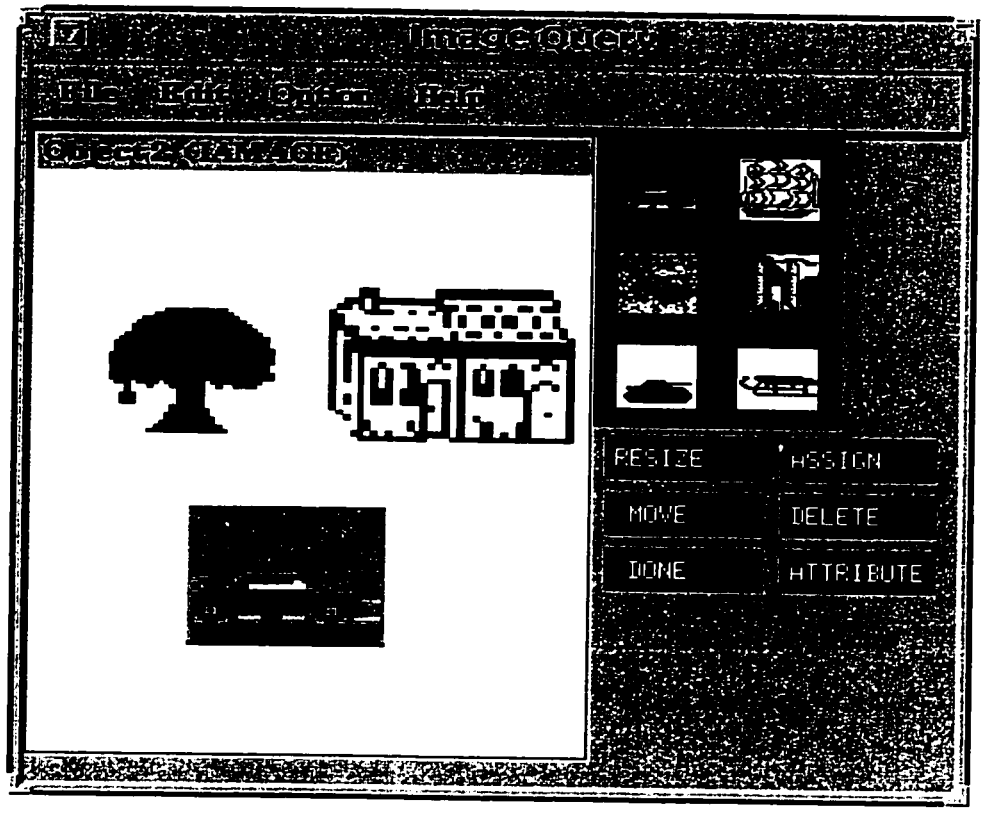


Figure 4.6 Object Query Example

Boolean operators, such as and, or, not, can be used in the query window to formulate more complicated query from two separated queries. For example, the user wants to see the pictures with a car or a truck located at the center. The query process is like this: first, selects the car icon, putting it at the center of the query window; then, selects "OR" boolean operator; finally, puts the truck icon at the same place as the car.

In addition to the boolean operators, we also introduce two more operators: combination and assignment, to make the icon query method have the capabilities of conveying more

semantic meanings. The semantics of the combination operator is to assemble some icons that have some kind of relationships among themselves together. For example, the relationship between a person and a T-shirt is "wearing". By associating a person icon and a T-shirt icon together with the use of combination operator, the user could construct the query of "a person with a T-shirt on". The purpose of the assignment operator is trying to assign more visual features, such as color and texture, to an icon to generate more meaningful icon. For instance, in order to retrieve an image of a red car, what a user might do to specify the query is to select the car icon, the assignment operator, and the red color from the color query window, which will be mentioned in the next section. In this way, the car has been assigned with color attribute. It is no doubt that the introduction of the two operators greatly improved the expressive power of the iconic query method.

4.4.3.1.3 Color query

In the color query window, there is a color picker provided to the user to select the desired color. Once a color is selected, it is displayed in a patch, and the user can further tune in the color in the patch with the help of a set of R, G, B sliders, until he/she is content with it. The user can divide the query space into different areas in order to assign different colors to each of them to formulate more complex queries in terms of color distribution of the image. For example, in order to specify the meaning of an image with grass land under a sunny sky, what the user needs to do is to divide the query space into two parts. The upper area is assigned to blue color, and the lower part is assigned to green. In this approach, the query space can be divided into sub-areas on the fly in the process of constructing the query on the user's demand. It

provides a means to specify the spatial relations in terms of color distribution of the image. Figure 4.7 is the color query window.

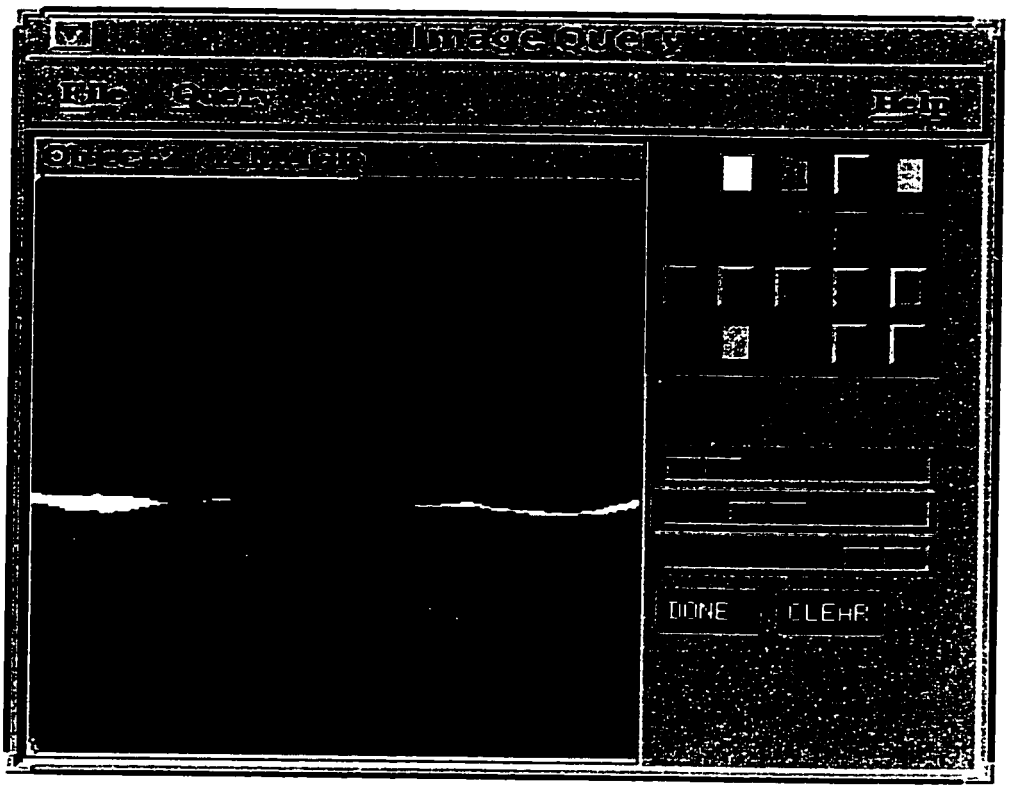


Figure 4.7 Color query window

4.4.3.2 Dynamic media

The dynamic media distinguishes itself from the static media with their temporal property and some other unique features. It is owing to its time-variant characteristics that make it difficult to construct the query on contents. Video and audio belong to this category. We emphasize on querying on the video.

Video is considered to be the most complicated medium type in terms of its content and properties. It is an "information rich" medium, possessing all the properties contained in image as well as temporal relationships and hierarchical structure. A video stream can be parsed into, from top to bottom level hierarchically, segments, scenes, and shots. A shot can further be represented by a keyframe or a number of keyframes, depending on the complexity of the shot. The keyframe could be thought of as a static image. Thus, we can use the query methods for image to query the keyframe. Perhaps the most important feature of video is the temporal information, including the object movements, camera operations, and the duration of the video, etc. The camera operations refer to the movements of camera, such as zoom in (out), pan left (right), and tilt up (down). Based on the information embedded in video, we divide the video query into two major steps: video temporal query and video content query.

4.4.3.2.1 Video temporal specification

The purpose of video temporal query is to create video objects, and specify the duration of each video object and the temporal relations among them. The video object here refers to any level of granularity of video part, such as segment, scene, shot or keyframe.

We use the querying space similar to that in the intermedia temporal query window to construct the video temporal query, as showed in Figure 4.4. The user can choose the "Video Object" button under the "Option" pull-down menu on the menu bar to create any number of video objects, which are visually represented by scales in the left part of the query window. By moving the sliders within the scales in both the left and right parts of the querying space, the user can specify the temporal relationships among the video objects. Combined with video content query windows, the semantic meanings could be added to the video objects.

The video temporal query window also supports nested temporal query, which means that the user could create sub-objects under any created video objects so as to support the hierarchical property of the video. The user could as well manipulate the temporal relationships of the sub-objects in the same way as what can be done to the video objects. In this way, temporal relation hierarchy is established to represent more complicated query. A supplementary window can be recalled to show the hierarchical relationships of created video objects and sub-objects.

4.4.3.2.2 Video content specification

The video content query deals with the content information of video, either dynamic or static. The user could use the query window mentioned in the image query to specify the static contents of the video, such as color, texture, visual object, and domain information. The query methods are almost the same as those used in image query. Figure 4.8 illustrates the general video query flow.

In addition to static properties, there are also dynamic features, such as camera operations and object movements, associated to video streams. We named it structure information, referring to the features of camera operations such as zoom, pan, tilt and types of transitions such as fade in, abrupt cut, etc. In [HNK95], an algorithm, which can successfully detect the cuts and some basic camera operations such as zoom in, zoom out, pan left, pan right, tilt up and tilt down, was introduced. Such information is considered to be unique and quite important contents of video. Therefore, we group them together to formulate the structure query window.

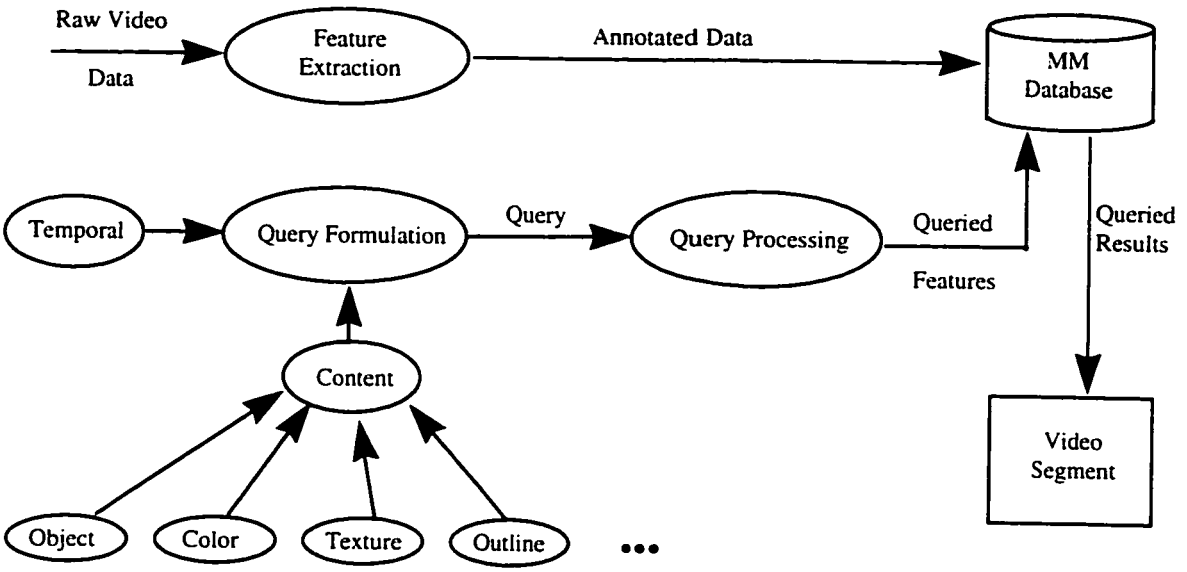


Figure 4.8 Video Query Flow

In the structure query window, a set of buttons representing the structure information is listed for the user to choose from. Once the user selects a button and clicks on it, an animation clip about the corresponding structure information would be displayed in the window to remind him/her of the behavior of the structure he/she has chosen. If the user considers that it is the right camera operation he/she is interested in, he/she can assign such a structure with the "assign" button in the window to a video object that has already been created in the video temporal query window. By manipulating the scales representing the video objects, the user can quantitatively control the length of the structure as well as the temporal relations with other video objects.

So far, we discussed the various query windows provided by the system to formulate multimedia query specifications. These windows help users to construct queries on the basis of contents of media.

4.4.4 Database Browser

Users sometimes just want to briefly browse through the database such that they can know it quickly if there exist their interested information in the database. Therefore, a simple and useful browser is provided to the end users. Figure 4.9 shows the appearance of the browsing interface.

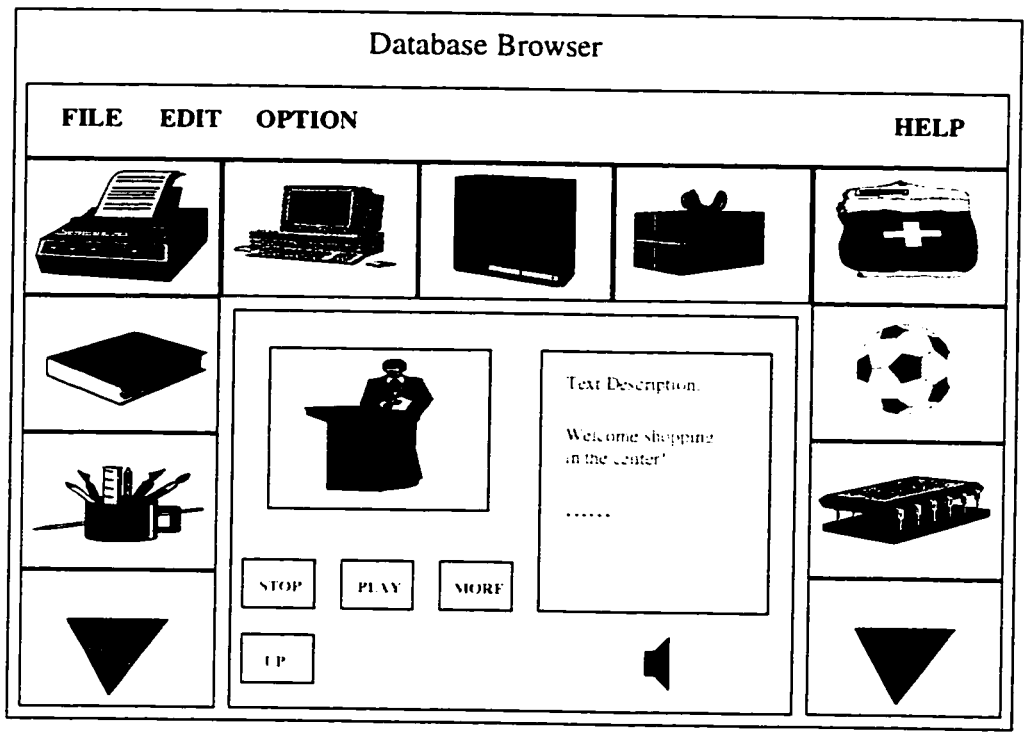


Figure 4.9 Database browser

The multimedia documents in the database are organized into a multi-tier structure based on their categories, which is similar to a shopping catalog of a department store. The basic functionality of the browser is to allow hierarchical browsing through the various levels of

granularity. The central part of the browsing window is the presentation window, which can present video, audio, image, and text. Surrounding the central presentation window is a U-shaped set of frames, each of which is a thumbnail picture of an image or a key-frame of a video stream. The thumbnail pictures shown at the same time on the window belong to same level of hierarchy. By clicking on the arrow buttons at the two ends, users can browse through all the thumbnails belonging to the same level.

The user can double click on a thumbnail frame to go one level deeper. The thumbnail pictures belongs to this deeper level will replace those of the upper level to show on the window. If the user find something interesting, the user can select the particular thumbnail frame and click the "PLAY" button on the presentation window. The video clip, accompanying by audio and text illustration, will be played back on the presentation window simultaneously.

This database browser is just a suggestion for the multimedia database system. It is not implemented as a part of the prototype graphical user interface

4.6 Summary

In this chapter, we introduced the architecture of the query interface for multimedia database that is under development in our laboratory. The system provides the user with a graphical query interface. Our main concern for the query interface is to use visual query methods as much as possible. Users can formulate queries based on their impressions, without the necessity of knowing the query language. Different types of media have their own characteristics, so we provide different query windows for query formulation. Since temporal relationship is the most important property of multimedia data, we paid much attention on how to

specify temporal queries effectively. The final query is the logical combination of the different query windows. A simple but practical browsing interface is also provided for accessing the database. In the next chapter, the implementation of the user interface (MEDIAQUINT) will be discussed.

Chapter 5

Implementation

5.1 Introduction

This chapter presents the implementation of the graphical query interface for the multimedia database (MEDIAQUINT) system. There are two main components of the interface: Query Specification Module and Query Translation Module. The query specification module is the front-end of the user interface, which is used by the user to construct queries. The query translation module is designed to interpret the user's queries specified on the graphical interface into the query language of the multimedia database such that the database system can understand the meaning of the query specifications. Figure 5.1 shows the flow chart of the user interface.

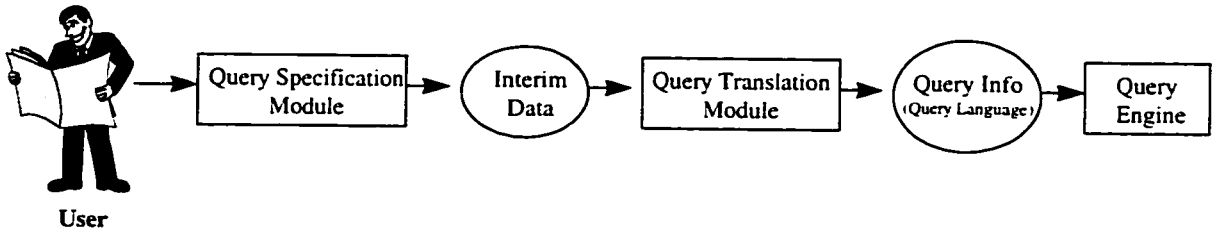


Figure 5.1 The flow chart of the user interface

The prototype of the graphical query interface is implemented on SPARC 10 workstation of SUN Microsystems. The operating system on the workstation is Solaris 2.4 from SUN Microsystems. The query specification module, or the GUI component of the interface, is implemented with Motif 1.2 [FPM93] and X Toolkit Intrinsics [NAO93]. The query translation module is implemented with C programming language.

The chapter starts with the description of the query specification module, including the query interface and the steps of constructing multimedia queries on the query interface. Next, the query translation module, including the data structure for storing the query information temporally, is described. Finally, a query example will be given to illustrate how to construct the query on the interface.

5.2 Query Specification Module

As we discussed earlier, querying on multimedia database is a relatively more complex process than its relational counterpart. In an attempt to simplify a complex process, people normally break it down into several simple sub-processes. Upon completing each sub-process, they put them back together to formulate the whole process.

On the query specification interface, users can construct queries on the fly without knowing the grammar of the query language, since the approach for query specification we are taking is highly graphical, which brings convenience to ordinary user as well as experienced user.

5.2.1 Steps of Query Construction

People might think about the same issue from very different angles. This happens on query specification as well. Ten persons might come up with ten different queries for the same multimedia scenario. This will make the system difficult to handle the different situations resulting from the on-the-fly query specifications by different users. Though our query interface allows users to construct queries based on what they have in their minds regarding the multimedia scenario, we design it to follow a sequential order while user's specifying queries in order to make the prototype of our interface simpler. Figure 5.2 shows us the sequence.

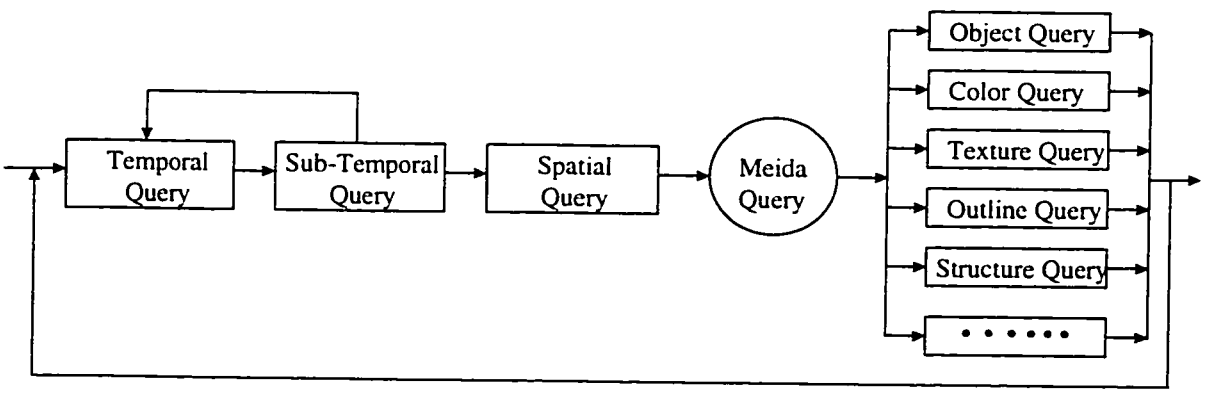


Figure 5.2 Query specification sequence

When the graphical user interface is launched, the startup window (Figure 5.3) is displayed. On startup window, users can launch all the query windows provided by the system. As shown on Figure 5.3, the window provides a set of push buttons and choosers. The three choosers that are arranged vertically on the right side of the window are used to pop up other query windows, such as temporal query window and spatial query window. The third chooser is a dropdown list, from which the user can choose query windows for different types of media, for example color, or visual object query windows for image media, and text query windows for text media. The three boolean operator buttons arranged horizontally can be chosen to logically cascade two query windows. In other words, the query specification on one window can be "ANDed" or "ORed to other windows " with the help of the boolean operators in order to construct more complicated queries. The File menu is used to save a well-constructed query into a file for reuse in the future. For example, a user wants to retrieve the similar multimedia documents from the database sometime later, the user can reload the query file to the query interface instead of constructing the query from the scratch. The lower part of the window is the query result window. It is a scrolled window, which is used to display the query results in the format of thumb nail images. The user can choose interested thumb nail candidate and play it back in full by double clicking on the corresponding thumb nail image.

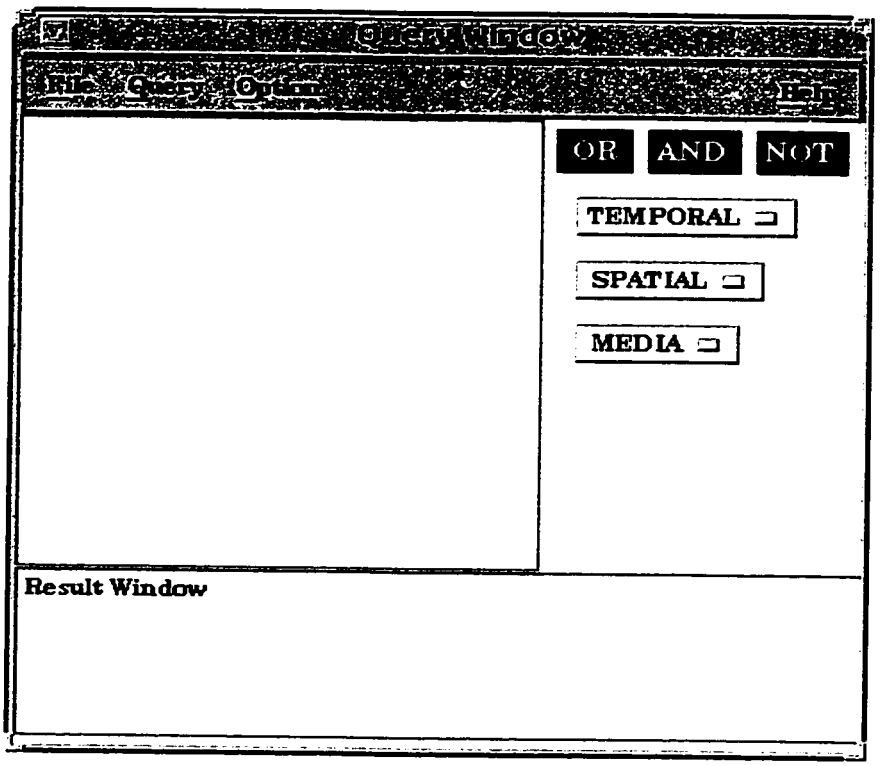


Figure 5.3 Startup window for the user interface

As we can notice on Figure 5.2, the very first step of query specification is temporal query. Clicking on the TEMPORAL button in the startup window pops up the temporal query window. On the temporal query window, users can create media objects in terms of their temporal relationships, including absolute duration of each media object and relative duration of each pair of media objects.

Since each media object may further consist of more granular sub-object, for example, a scene of video might be composed of several relevant shots, so the next step of query

specification is sub-temporal query. On the sub-temporal query window, based on the media objects already created on the temporal query window, users can further create sub media objects and specify their temporal relationships. The sub-temporal query can be a recursive process, which means a sub-temporal query may recursively contain other sub-temporal queries until an atom component is reached. The atom component here refers to the component that can not be further temporally decomposed.

The third step is spatial query: specifying the spatial layout of each medium type. The final step is media query. The query methods for media query are media dependent. For example, we use outline query method for text media, and use color, texture, and object query methods for image.

The query steps we described above are only one circle of query specification. When one circle is finished, the user can go back to the very first step to modify what has been done until the user is pleased with each step and finally sends the query specifications to the system.

5.2.2 Data Structure

The query specification information constructed by the user on the GUI has to be stored in some interim format for the further use. For example, it will be used by query processing engine to do exhaustive search for the matched multimedia documents in the database. We adopt a set of data structure to contain such query information. Figure 5.4 lists the key components of the data structures.

As we can see on Figure 5.4, the ***temporal_object*** data structure is used to hold the query information specified in the temporal query window. Each instance of the ***temporal_object*** data structure is corresponding to a temporal object created on the temporal query window. The *object_no* element of the data structure is the unique key to distinguish a temporal object from one another. The *media_type* element tells what kind of medium this temporal object is. For example, text is represented by 1, image is represented by 2, and so on. The third element is a float number, used to store the start point of the temporal object in terms of second, minute, or hour. The next element is also a float number, storing the information that how long the object lasts. The *time_unit* element holds the information about the unit of time, which can be one of the several choices, such as millisecond, second, minute and hour, of the previous two elements. This parameter can be set on the query interface by the user to fit for different lengths of media segments. For example, some segments may last only several milliseconds, which can be a camera operation within a video stream, such as an abrupt cut; while some others may persist a few minutes. The *opt* element is used to carry the accuracy information about the *duration* element mentioned above. It is helpful when user wants to specify less accurate temporal queries, says a video segment less than 10 seconds long or a video segment with any seconds of length. We have four candidate operators here: =, <, >, and any. The default operator is "=", which means that if no operator is explicitly specified on the GUI, the length of the scale on the GUI represents the exact length of the temporal object. The *sub_obj* field points to a list of sub temporal objects that the parent object might contain. This field is useful for those media that have hierarchical properties, such as video and audio. For example, a video stream may recursively consist of scenes, segments, and shots. For the non-hierarchical media, this element is a NULL pointer, which means it has no sub objects. The next two

elements contain the spatial information of the temporal object: the *left_upper_corner* holds the coordinate of the left upper corner of the physical location of the temporal object on the window, while the *right_lower_corner* holds the coordinate of the opposite corner. The last element is the content specification of the temporal object. Since the temporal object may represent different types of media, this element is media dependent. It might be one of the several pointers, such as text_content, image_content, and video_content, referring to the content specification of a specific type of medium.

The other data structures are similar and self-explanatory, and we will not discuss them in detail.

```
temporal_object ::
        object_no           :   integer;
        media_type          :   integer:  (Text=1. Image=2. Video=3. ...)
        start               :   float:    /*staring time of the object*/
        duration            :   float:    /*time length of the object*/
        time_unit           :   integer. /* unit to represent time. Sec.=1.
                                    Min.=2.Hour=3 */
        opt                 :   inteter. (=:1. <:2. > 3. any:4 )
        sub_obj             :   ^sub_obj_list:   /*if the object contains sub objects. this points
                                to the sub obj list; otherwise. NULL*/
        left_upper_corner   :   ( int. int ):    /*spatial specification*/
        right_lower_corner  :   ( int. int ):    /*spatial specification*/
        content_spec        :   ^text_content          or
                                ^image_content   or
                                ^video_content;
        END

text_content::
        keyword             :   string;
        boolean             :   Boolean: /*AND. OR. ... */
        next                :   ^text_content;
        END

image_content::
        color_spec          :   ^color_info;
        visual_spec         :   ^viaual_info:    /*prominent visual object in
                                the   image*/
        texture_spec        :   ^texture_info:   /*similar to color information*/
        END

color_info::
        color               :   integer;
        left_upper          :   ( int. int ):    /*spatial info of the color*/
        right_lower         :   ( int. int ):    /*spatial info of the color*/
        boolean             :   Boolean: /*AND. OR. ...*/
        next_color          :   ^color_info:     /*pointing to the next color_info
                                associating      with it by the boolean operatior*/
        END

visual_info::
        serial_no           :   integer:  /*sequential number of the visual object
                                in this image */
```

```
        object_name          :    ^character:
        left_upper           :    ( int, int ):        /*spatial info of the object*/
        lower_right          :    ( int, int ):        /*spatial info of the object*/
        boolean              :    Boolean:
        next_obj             :    ^visual_info:        /*pointing to the next visual_info
                                  associating with it by the boolean operation*/
        add_info             :    ^character:  /*additional info for the visual object*/
        END

video_content::
        color_spec           :    ^color_info:
        texture_spec         :    ^texture_info:
        visual_spec          :    ^visual_info:
        structure_spec       :    ^structure_info:
        END

structure_info::
        structure_name       :    ^character:        /*zoom_in, fade_out, pan_left, ...*/
        boolean              :    Boolean:
        next_structure       :    ^structure_info:
        END
```

Figure 5.4 Major components of data structures

5.3 Query Translation Module

So far, the query specification information is in the format of interim data stored in the memory of the system. It cannot be recognized by the database system for query processing, since it is not in the format of the query language of the system. What the query translation module can do is to interpret the interim data into the format of the proprietary multimedia query language, which will be further processed by the query engine in searching of the matched multimedia documents.

Figure 5.5 gives us the abbreviated algorithm of the query translation module. The query translation happens when the user has completed the query specifications on all the query windows provided by the system. At this time, there are a fixed number of temporal objects created on the temporal query window, and the query information specified on other query windows has been assigned to the corresponding temporal objects. The algorithm will start a loop to traverse the temporal objects to parse the query specification, such as temporal, spatial and media information, one object after another.

Taking the temporal specification as an example, it first determines what type of media this object is; then it calculates the starting point of the object in the unit of minute, second, or millisecond; next it determines the duration of the object; if the *opt* element of data structure is not empty, it will add accuracy operator, such as =, <, or >, to the object; if the object has sub-objects, it will further translate the query specifications associated with them. The translation module ends when it has successfully traversed all of the temporal objects created on the temporal query window.

```
FOR k=1          TO       no_of_obj
        {
         /* TemporalSpec */
         {
         determine media type of object k:
          calculate starting time of object k:
          calculate duration of object k:
          IF opt THEN modify duration of object k:
          IF       (exist sub_object)  THEN
                  {
                   j=1       TO       no_of_subobj
                        {
                          process sub_obj j:
                        }
                        Temporal translation
                  }
         /* SpatialSpec */
         {
           left_upper_corner of object k:
           right_lower_corner of object k:
         }
          Spatial Translation
         /* ContentSpec */
         {
           IF      (media_type==Text)          THEN
                   {
                     capture keyword:
                     while  (exists boolean operator)
                          {
                            capture boolean operator:
                            move pointer to next keyword:
                          }
             Text translation
           }
           IF      (media_type==Image)         THEN
                   {
                     CASE  ColorSpec
                          {
                            capture color name:
                            capture color position:
                            while     (exists boolean)
                                  {
                                    capture boolean opterator:
                                    capture color name:
                                    capture color position:
                                  }
                          }
                      CASE  ObjectSpec
                          {
                            m=1       TO       no_of_visualobj
                                  {
                                    capture visual object name:
                                    capture visual object postion:
                                    while     (exists boolean)
                                          {
                                            capture boolean opterator:
                                            capture color name:
                                            capture color position:
                                          }
                                  }
                          }

                      CASE  TextureSpec
                          {
                            similar to ColorSpec
                          }
             Image translation
                  }
```

```
        IF    (media_type==video)
              {
                Color_spec:
                Texture_spec:
                Visual_obj_spec.:
                Structure_spec:
                while  (exist subobj)
                       {
                         Color_spec:
                         Texture_spec:
                         Visual_obj_spec:
                         Structure_spec:
                       }
          Video translation
              }
    }
}
```

Figure 5.5 Abbreviated Algorithm of query translation module

The query specification might be more complicated in some case. For example, a temporal object created on the temporal specification window may recursively contain sub temporal objects. The user might be confused about how many objects and sub-objects have been created and what are the hierarchical relationships among them. Therefore, we provide a reminder window to the user to keep track of the hierarchy information. Figure 5.6 gives us an example: four temporal objects and two sub-objects within the 4^{th} object.

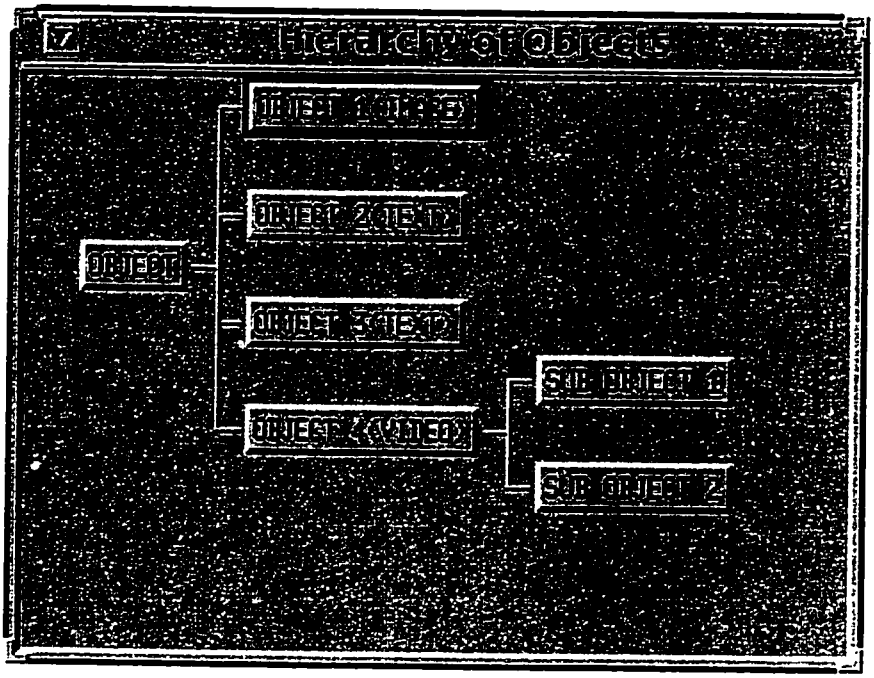


Figure 5.6 Hierarchy of temporal objects

5.4 Query Example

We have discussed the basic functions of the query user interface, and how they are implemented. What follows is a simple example on how the user can specify queries on the interface.

Example: Retrieving a video clip on which a red car is moving from the left side to the right side, and there is a house and blue sky as the background.

The query steps of specifying such a query is like this: (1) On the startup window, select and popup the temporal query window first; on the temporal query window, create two temporal objects; adjust the length of the two temporal objects to <= 2/15 seconds; (The reason to let them less equal than 2/15 seconds is that a user-acceptable video stream consists of 15 frames per seconds, and 2/15 seconds is equivalent to the length of two frames.) adjust the relative duration of the tow objects to, says < 1 second. (2) Select and popup the visual object query window; pickup the house icon from the icon hierarchy and put it on the query window; pickup the car icon and put it on the left side of the query window and assign red color to it. (3) Click on AND button and popup the color query window in an attempt to do "AND" operation of the two windows (object query window and color query window); on the color query window, select the blue color from the color picker and paint it on the whole color query window. (The result of the "AND" operation of the two windows portraits such a picture: a red car on the left side of the screen with a house and blue sky as the background.) And then assign the two windows to the first temporal object created on the temporal query window. (4) Repeat step 2 and 3 except putting the car icon on the right side of the query window, and assign the result to the second temporal object on the temporal query window. Figure 5.7 shows us the example.

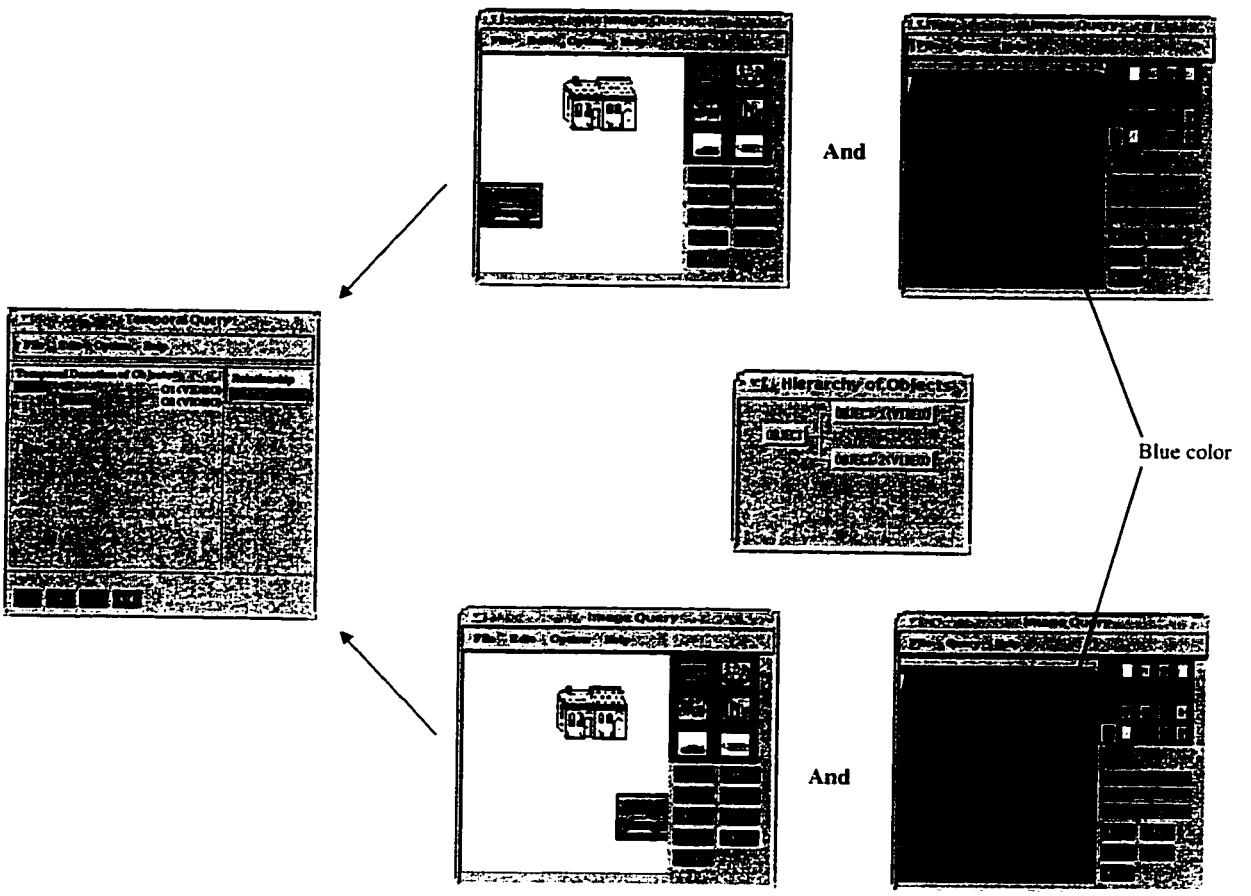
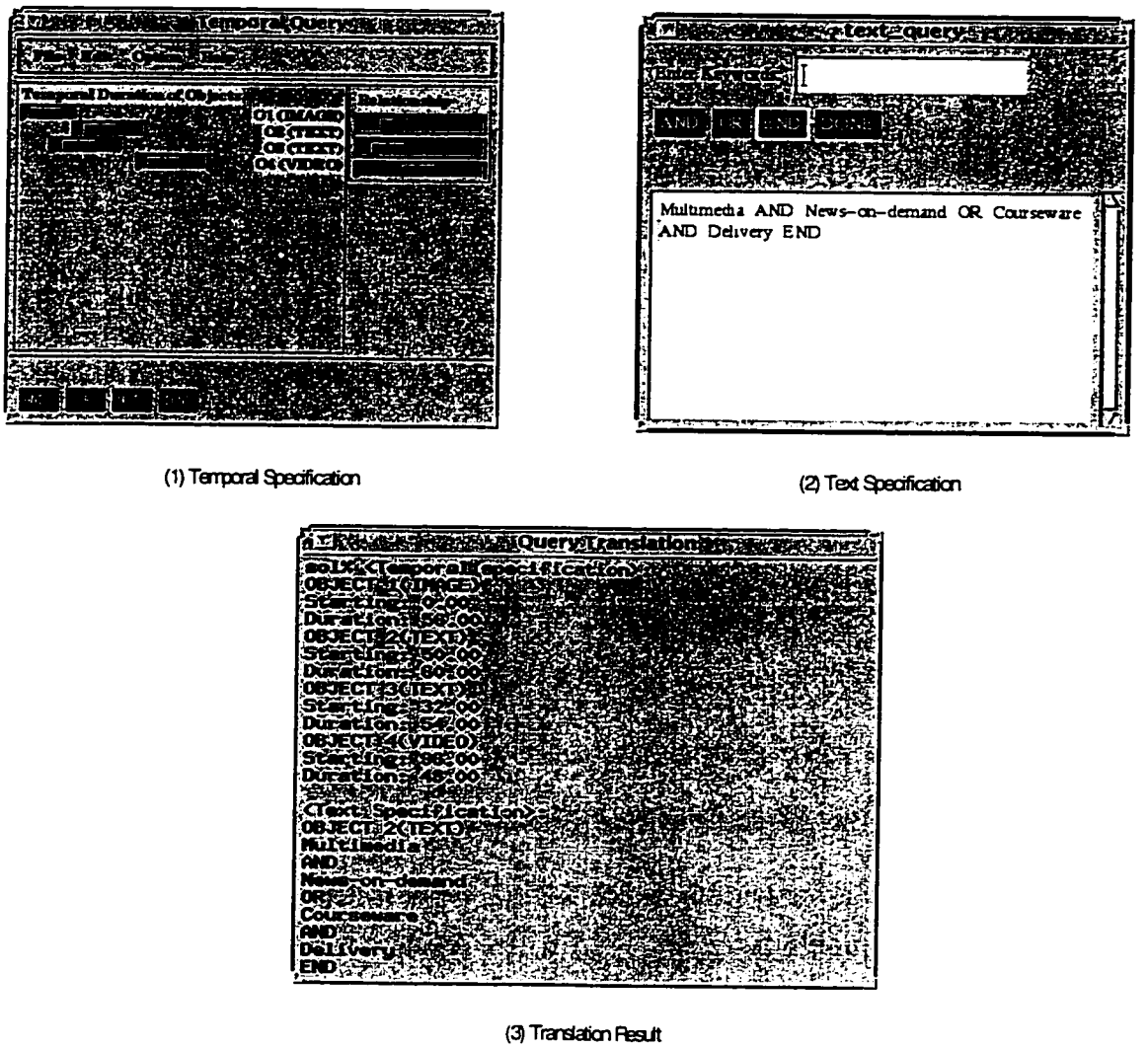


Figure 5.7 Query example

5.5 Evaluation

On the query interface, the user has to follow a sequential order when constructing queries on the different query specification windows. This is because that what has been implemented is a prototype of the design. It has been tested that the query interface is able to fulfill the task of query specification and query translation under normal conditions. But we didn't take into account the error handling issue of the user interface. In other words, if users

didn't follow the query steps, the user interface might crash due to the lack of exceptional handling capability.



(1) Temporal Specification

(2) Text Specification

(3) Translation Result

Figure 5.8 Query translation example

Figure 5.8 shows an example of query specification and query translation. As we can see, Figure 5.8(1) is a temporal query window, on which four temporal objects are created: an image object, two text objects, and a video object, with the object number from 1 through 4

respectively. The temporal relationships of the four objects are clearly specified on the window. Figure 5.8(2) is a text query window, on which the user can entering keywords combined with boolean operator to specify queries on text. When the keywords input is done, the user can assign such text specification to either of the two text objects created on the temporal window. In this example, we assign it to object 2. Figure 5.8(3) is the translation result interpreted by the query translation module according to the query specification information on the two query windows.

5.6 Summary

In this chapter, the implementation of the graphical query interface for multimedia database (MEDIAQUINT) is presented. It shows that the graphical query interface is an integral component of the database system, with the help of which the user interacts with the system. Normally, a complex query is divided into several steps. We discussed the steps of constructing queries on our query interface, and introduced the different query windows provided by the system. Next, the implementation of the query interface was detailed. The front-end of the interface, says the GUI, is responsible for dealing with the user's requirements. At the back-end, a set of data structures used by the system to store the query information were introduced. After a brief description of the data structures, the translation module, which translates the query information into the format of the proprietary query language, was discussed, and a translation example was given to interpret how the translation module works. Finally, a simple query scenarios were given to show how query can be constructed on our query interface.

The prototype of our interface has a number of limitations, which result from the fact that a simplified version of the graphical query interface was adopted for implementation. The

implementation tools, the Motif and X Toolkit, don't seem to fit the needs of implementing such a query interface very well, though they are popular GUI design tools on UNIX platform. At the time of implementation, we didn't have other alternatives, such as Motif++, Java, Visual C++, which are totally object-oriented methodologies and will make GUI design much simpler and more efficient.

Chapter 6

Conclusions

6.1 Summary

In this thesis, a user query interface for multimedia database, called MEDIAQUINT (QUery INTerface for MutiMEDIA database) was presented. The MEDIAQUINT is built on top of the proprietary multimedia query language, which was introduced in [HNK95]. It focuses on providing a visual facility to the end users of multimedia database to allow them to directly and efficiently retrieve interested information from the database based on the contents of the information in their minds. It also provides some sort of browsing capabilities to help the user to navigate through the database.

The basic architecture of the user interface for the multimedia database system was discussed. The front-end of the user interface should have both query and browsing interfaces to support retrieval by querying as well as retrieval by browsing of the multimedia information. One of the main issues existing in current database system development is how to facilitate users to construct complex queries in order to retrieve information from the database. Some database systems lack the easy ways for the user to interact with them. This inconvenience results from the gap lying between the query language of the database system and the users' ways of thinking. For some ordinary end users, it is not an easy step to translate what in their minds into the matching query language expressions. The ultimate goal of MEDIAQUINT is to provide a straight forward and easy-to-use interface to the users to achieve efficient multimedia information retrieval.

The MEDIAQUINT consists of two main components: query specification module, including the browsing facility, and query translation module.

The query specification module is the graphical front end of the interface, on which the users formulate their query specifications. We separate the query specifications into two major steps: intermedia specification and intramedia specification. The intermedia specification deals with the temporal and spatial relationships among different media types, while the intramedia specification provides the means to specify queries based on the contents of a specific media. Concerning the intramedia specification, we further divide the media into two categories: dynamic media and static media. The user can navigate through the different windows and follow some steps to construct a complex query. The final query will be formulated by logically combining the different query steps together.

The query translation module is the back end of the user interface. It is responsible for interpreting the query information specified on the GUI into the proprietary query language of the database system, which will be further used by the query engine in searching of the matched multimedia segments in the database.

The MEDIAQUINT was implemented in C programming language and Motif on SUN Sparc workstation running Solaris 2.3 operating system.

6.2 Suggestions for Future Work

Multimedia database system can be applied to a wide range of areas, such as news-on-demand, education, commercial, and financial, etc. The query interface and the corresponding query language should not be restricted to a specific domain. On the contrary, they should be designed to have the ability of facing challenges under different circumstances. It is safe to say that whether or not supporting a wide range of domains is one of the criteria to evaluate the query interface and the underlying system.

The multimedia query language is the foundation of the underlying query interface. So far, there isn't a standard for multimedia query language. Is it possible to adopt a wide-accepted standardized multimedia query language, just like SQL that is the standard query language for relational database?

The amalgamation of different types of media, especially the inclusion of dynamic media such as video and audio, makes multimedia database system difficult to handle. Among these media, video is the most complicated one, in terms of not only its structure but also the abundant

information it might include. The dynamic properties of video are worth our further attention. On our interface, users can use the camera operations, such as zoom, pan, and tilt to specify queries on video. There also exists other important dynamic properties, for example, the motion of objects, the trajectory of object movement, etc. As far as it is concerned, the future work lies in feature extraction and indexing of the useful dynamic properties which can be used in constructing more complicated queries. For instance, features that are considered to be more useful than others for a particular multimedia application should be paid more attentions.

Audio is thought of as another important dynamic media. It could be either as a stand-alone medium, or as an important component of video. Therefore, how to utilize the audio properties to formulate its own query, or incorporate it to enhance the video query is another issue worthy of concern. For example, we can use a type of, say "audioIcon", to represent a particular kind of sound, such as a bird singing and thunderstorm. Users can click on the audioIcon to play the sound back, and query the system to retrieve audio documents with the similar sound.

Bibliography

[ABL95] Ahanger, G., Benson, Dan, and Little, T.D.C., "Video Query Formulation", Storage and Retrieval for Image and Video Databases II, IS&T/SPIE Symposium on Electronic Imaging Science & Technology, San Jose, CA. Feb. 1995.

[ASE92] Ahuja, S.R., and Ensor, J.R., "Coordination and Control of Multimedia Conferencing", IEEE Comm., Vol. 30, No. 5, May 1992, pp.38-43.

[BAD93] Bimbo, A.D., "A Three-Dimensional Iconic Environment for Image Database Querying", IEEE Transaction on Software Engineering, Vol. 19, No. 10, October 1993, pp.997-1011.

[BMD92] Blattner, M.M., and Dannenberg, R.B., "Multimedia Interface Design", ACM press, 1992.

[BMM94] Blattner, Meera M., "In Our Image: Interface Design in the 1990s", IEEE Multimedia, Spring 1994, pp. 25-36.

[BOF94] Borko Furht, "Multimedia Systems: An Overview", IEEE Multimedia, Spring 1994, pp.47-59.

[CFU90] Chang, N.S., and Fu, K.S, "Query-By-Pictorial-Example", IEEE Transactions on Software Engineering, Vol. SE-6, No. 6, 1980, pp.519-524.

[CIT93] Cardenal, A.F., Ieong, I.T., and Taira, R.K., "The Knowledge-based Object-oriented PICQUERY+ Language", IEEE Transaction on Knowledge and Data Engineerin, Vol. 5, No. 4, August 1993, pp.644-657.

[COB93] Cole, B., "Interactive Multimdia: The Technology Framework", IEEE Spectrumk Vol. 30, No. 3, March 1993, pp.32-39.

[CWJ92] Clark, W.J., "Multiport Multimedia Conferencing", IEEE Comm., Vol. 30, No. 5, May 1992, PP.44-50.

[DAM93] Davis, M., "Media Stream: An Iconic Visual Language for Video Anotation", Proc. of 1993 IEEE Symposium on Visual Languages, Bergen, Norway, IEEE Computer Society Press, 1993, pp.196-202.

[DCJ91] Date, C.J., "An Introduction to Database Systems", Vol. 1, 5th edition, Addision-Wesley Publishing Company, 1991, pp.3-102.

[DEL94] Deardorff, E. and Little, T.D.C "Video Scene Decomposition with the Motion Picture Parser", Proc. of IS&T/SPIC Symposium on Electronic Science and Technology, San Jose, Feb. 1994.

[DNN91] Davies, N.A. and Nicol, J.R., "Technological Perspective on Multimedia Computing", Computer Communications, Vol. 14, No.5, June 1991, pp.260-272.

[ERN94] Elmasri, Ramez and Navathe S.B., "Fundamentals of Database Systems", second edition, The Benjamin/Cummings Publishing Company, 1994.

[ESL89] Emerson, S.L., "The Practical SQL Handbook: Using Structured Query Language", Addison Wesley Publishing Company, 1989.

[FBK95] Falchuk, B., Karmouch, A., "A Multimedia News Delivery System Over an ATM Network", Proc. Of International Conference on Multimedia Computing and Systems, pp.56-63, Washington D.C., May 1995.

[FEA91] Fox, E.A., "Advances in Interactive Digital Multimedia Systems", Computer, Vol. 24, No. 10, 1991, PP.9-21.

[FKA96] Falchum, B., and Karmouch, A., "Feature Set and Architecture of a Multimedia News System Over an ATM Network", Multimedia Tools and Applications, Kluwer Academic Publishers, Vol. 2, No. 1, PP.11-33.

[FPM93] Ferguson, Paula M., "Motif Reference Manual", O'Reilly & Associates, Inc., 1993

[GDL91] Gall, D. Le, "MPEG, A Video Compression Standard for Multimedia Applications", ACM Communications, Vol. 34, No. 4, April 1991, pp.46-58.

[GYZ94] Gong, Y., Zhang, H., and Chuan, H.C., "An Image Database System with Content Capturing and Fast Image Indexing Abilities", Proc. of Intl. Conference on Multimedia Computing and Systems, Boston, USA, May 1994, pp.121-131.

[HIK95] Hirzalla N., and Karmouch A., "A Multimedia Query Specification Language", International Workshop on Multimedia Database Management Systems, New York, August 1995.

[HNK95] Hirzalla, N and Karmouch, A. "Automatic Cut and Camera Operation Detection for Video", International Conference on Consumer Electronics, June 1995.

[HSR95] Hibino, S., and Rundensteiner, E. A., "A Graphical Query Language for Identifying Temporal Trends in Video Data", Proceedings of International Workshop on Multimedia Database Management Systems, Blue Mountain Lake, New York, August 1995.

[JRA94] Jain, Ramesh, "Metadata in Video Database", SIGMOD RECORD, Vol. 23, No. 4, pp.27-33, Dec. 1994.

[JTC88] Joseph, T., and Cardenas, A.F., "PICQUERY: A High Level Query Language for Pictorial Database Management System", IEEE Transaction on Software Engineering, Vol. 14, No. 5, May 1988, pp.630-637.

[KAA9] Karmouch, A., "A Multimedia Information and Communication System: MEDIABASE", Proc. of Multimedia Communication'93, Banff, April 1993.

[KAC94] Kamm, Candace, "User Interface for Voice Applications", Voice Communication Between Humans and Machines, Natioan Academay Press, Washington D.C., 1994.

[KAT94] Kau, S.C. and Tseng, J.C.R. "MQL—A Query Language for Multimedia Database", Proceedings of Multimedia 94, Japan, pp5-1-1--pp5-1-6.

[KDA93] Keim, Daniel A., "GRADI: A Graphical Database Interface for a Multimedia DBMS", Proc. of Intl. Workshop on Interface to Database Systems, Glasglow, U.K., 1993, pp.95-112.

[KWK95] Kwan, W. and Karmouch, A., "An Intelligent Agent for Multimedia Newspaper", Proc. of the Canadian Conference for Electrical and Computing Engineering, Montreal, pp.594-597, Sept. 1995.

[LDP93] Lucarella, D., and Parisotto, S., "More: Multimedia Object Retrieval Environment", Proceedings of Hypertext'93, pp.39-50, Nov. 1993.

[LRW94] Lee, D., Barber, R., and Niblack, W., "Query by Image Content Using Multiple Objects and Multiple Features: User interface Issues", Intl. Conference on Image Processing, Vol. 2, Nov. 1994, Austin, TX, pp.76-80.

[NAO93] Nye, Adrian, and O'Reilly, Tim, "X Toolkit Intrinsics Programming Manual", O'Reilly & Associates, Inc., 1993.

[NIW93] Niblack, W., and Barber, R., "The QBIC Project: Querying Image By Content Using Color, Texture and Shape", Proceedings of SPIE, San Jose, CA, Feb. 1993.

[NWG94] Neil Williams and Gordon S. Blair, "Distributed Multimedia Applications: A Review", Computer Communications, Vol.17, No.2, Feb. 1994.

[OET93] Oomoto E., and Tanaka, K., "OVID: Design and Implementation of a Video-object Database System", IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 4, August 1993, pp.629-643.

[OVM95] Ogel, V.E., and Stonebraker, M., "Chabot: Retrieval from a Relational Database of Images", IEEE Computer, Sept. 1995, pp. 40-48.

[PAJ94] Lynch J. Patrick, "Visual Design for the User interface", Journal of Biocommunications, Vol. 21, 1994, pp.22-30.

[PDT95] Papadias, D., and Sellis, T., "A Pictorial Query-By-Example Language", Journal of Visual Languages and Computing (Special Issue on Visual Query Systems), March 1995.

[RJK95] Rody, J.A. and Karmouch A. "A Remote Presentation Agent for Multimedia Database", International Conference on Multimedia Computing and Systems, Washington D.C. May, 1995.

[ROB94] Rowe, Lawrence, and Boreczky, J. S., "Indexes for User Access to Large Video Databases", Storage and Retrieval for Image and Video Databases II, IS&T/SPIE Symposium on Electronic Imaging Science & Technology, San Jose, CA, Feb. 1994.

[SCB92] Schneiderman, B., "Designing the User Interface: Effective strategies for Effective Human-Computer Interaction", Addison-Wesley, 2nd edition, 1992.

[SRW92] Stevens, R.W., "Advanced Programming in the Unix Environment". Addison Wesley Professional Computing Series, 1992.

[SSZ94] Smoliar, S.W., and Zhang, H., "Content-Based Video Indexing and Restrieval", IEEE Multimedia, Summer 1994, pp.62-71.

[VHM91] Vin, H.M, "Multimedia Conferencing in the Etherphone Environment", Computer, Vol. 24, No. 10, Oct. 1991, pp.69-79.

[WED94] Weiss,R. and Duda, A. "Content-Based Access to Algebraic Video", Proc. of International Conference on Multimedia Computing and Systems, 1994, pp.140-151.

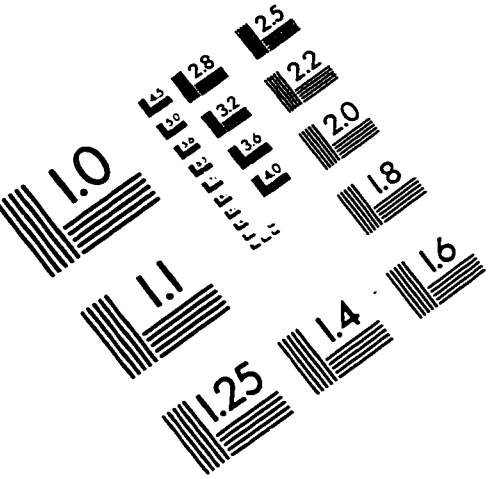
[WAK96] Wang, R., and Karmouch, A., "A Broadband Multimedia Telelearning System", Proc. of 5th International IEEE Symposium on High-Performance Distributed Computing – Multimedia and Collaborative Environment, Syracuse, Aug. 1996.

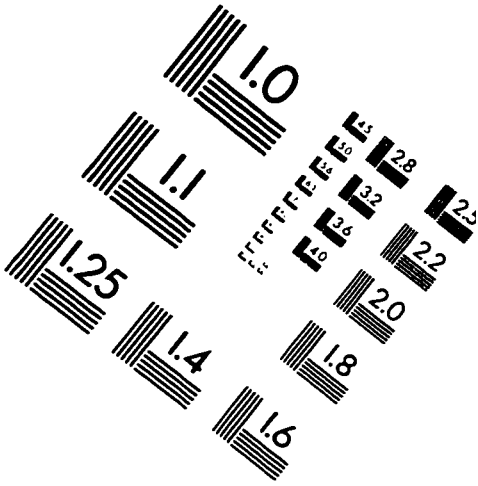
[YAK94] Yoshitaka, A., and Kishida, S., "Knowledge-assisted Content-based Retrieval for Multimedia Database", IEEE Multimedia, Winter 1994, pp.12-21.

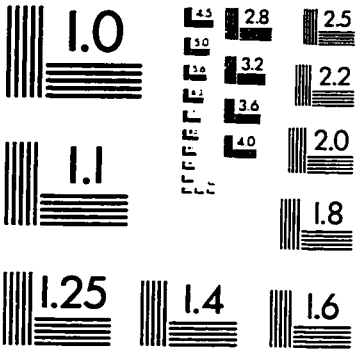
Publication

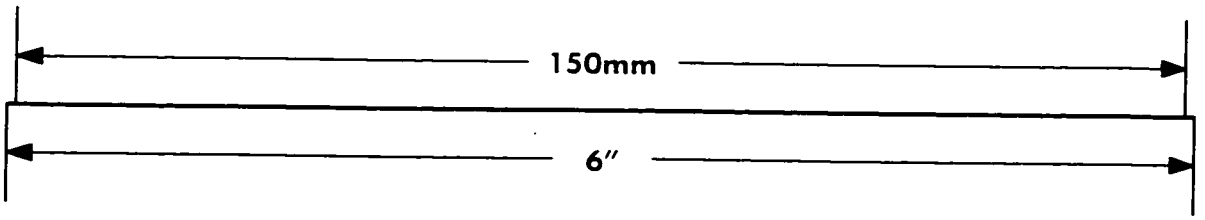
J. Cheng and A. Karmouch, "A Graphical Query Interface for Multimedia Database System", International Conference on Consumer Electronics, Chicago, June, 1996.

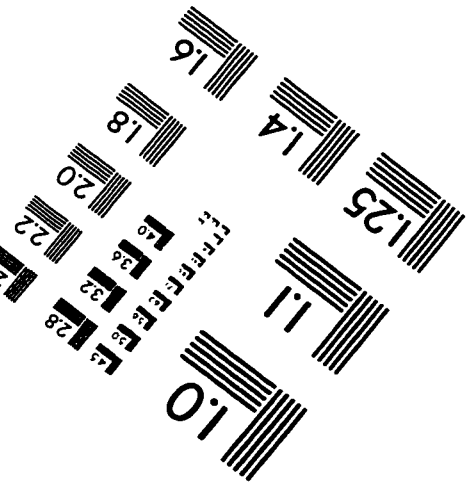
IMAGE EVALUATION TEST TARGET (QA-3)

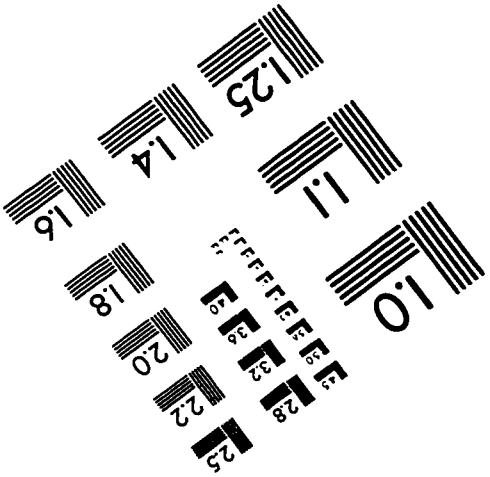












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989

© 1993, Applied Image, Inc., All Rights Reserved