Multimedia Database Server Implementing Content based Retrieval

Cosmin Stoica Spahiu, Liana Stanescu, Dumitru Dan Burdescu and Marius Brezovan

University of Craiova, Faculty of Automation
Computers and Electronics, Craiova, Romania
{stoica.cosmin, mihaescu, stanescu, burdescu,
marius.brezovan}@software.ucv.ro

Abstract. The article presents a software tool implemented in C++ that implements a multimedia database server. An element of originality is that along with the classical functions of a server it has a specialized module for content based retrieval. The users can execute both simple text based queries and complex visual queries, based on a query image. The server processes the images and extracts the color and texture characteristics and stores them in a new data type called IMAGE. The image color information is represented by means of color histograms resulting from the transformation of the RGB color space to HSV color space and the quantization to 166 colors. In order to represent the texture it is considered the co-occurrence matrices .To compute the dissimilitude between the images, the histogram intersection has been used for the color and the Euclidian distance for the texture. It is also presented the client-server communication based on SQL language.

Keywords: Multimedia database server, Content based retrieval, Image data type.

1 Introduction

The success of the digital revolution and the growth of the Internet have ensured that huge volumes of high-dimensional multimedia data are available all around us. The medical system represents an important area were large amount of digital information are produced every day due to medical devices (echograph, endoscope, MRI). As the quantity of medical images is increasing, the problem of storing medical image collections in digital format along with the associated information (patient name, diagnosis, consulting date and treatment), managing the database and executing efficient queries, are aspects that are intensely studied for finding new and more efficient solutions. This information is often mixed, involving different data types such as text, image, audio, graphics, and video components interspersed with each other.

In order to manage content based retrieval for medical image collections a series of applications that use traditional DBMS, have been implemented. Most of them use servers like MS SQL Server, My SQL or PostgreSQL. The complete solution is provided by Oracle - the Oracle 10g database server and Intermedia tool that can

Stoica C., Stanescu L., Burdescu D. and Brezovan M.
Multimedia Database Server Implementing Content based Retrieval.
DOI: 10.5220/0004463400780084
In *Proceedings of the 3rd International Workshop on Enterprise Systems and Technology (I-WEST 2009)*, pages 78-84
ISBN: 978-989-674-015-3

manage all kind of multimedia data. This kind of solution involves high costs for buying the database server and for designing and implementing complex applications for content based query [6].

This paper presents an application implemented in C++ that includes a Multimedia Database Management Server (MMDBMS) based on the SQL3 standard that is less expensive than a commercial database server. This server is designed to manage medium sized image collections.

This MMDBMS has the following elements of originality and advantages:

- images are stored directly in the database in a new data type called image
- a specialized module is used to process images and extract visual characteristics from images
- possibility to execute visual content based queries
- possibility to add on the client side a visual interface for content-based image query using color and texture characteristics, in order to visually select the query image.

The paper has the following structure: section 2 presents the server description, section 3 presents the content based retrieval functionality, section 4 presents the client server Image data type, and section 5 presents the conclusions and future work.

2 Server Description

The MMDBMS allows database creation, table and constraints adding (primary key, foreign keys), inserting images and alphanumerical information, simple text based query and content-based query using color and texture characteristics. The software tool is easy to be used because it respects the SQL standard. It does not need advanced informatics knowledge and has the advantage of low cost. It is a good alternative for a classical database management system (MS Access, MS SQL Server, Oracle10g Server and Intermedia), which would need higher costs for database server and for designing applications for content-based retrieval.

It is designed in a modular way, where each module executes a certain function. There is a module specialized to client communication, database management, executing text based queries, executing content based retrieval and collecting the responses to queries.

A simplified schema of the server is presented in the next figure.

In the first step, the client applications connect to the server using TCP sockets. This way it will be created a communication channel between them. All commands and responses will use this channel to send queries requests and receive answers.

The communication module manages all these messages. It waits to receive commands from users (queries, updates, system commands, etc.) and pass them to a main module which manages them all. After the command is executed, the response will be returned by the Query Response Module to the Communication Module in order to be sent to the client. The Query Response Module will compact the result using a standard format and then return it to the client. The client will receive it on the same communication channel used to send the request.

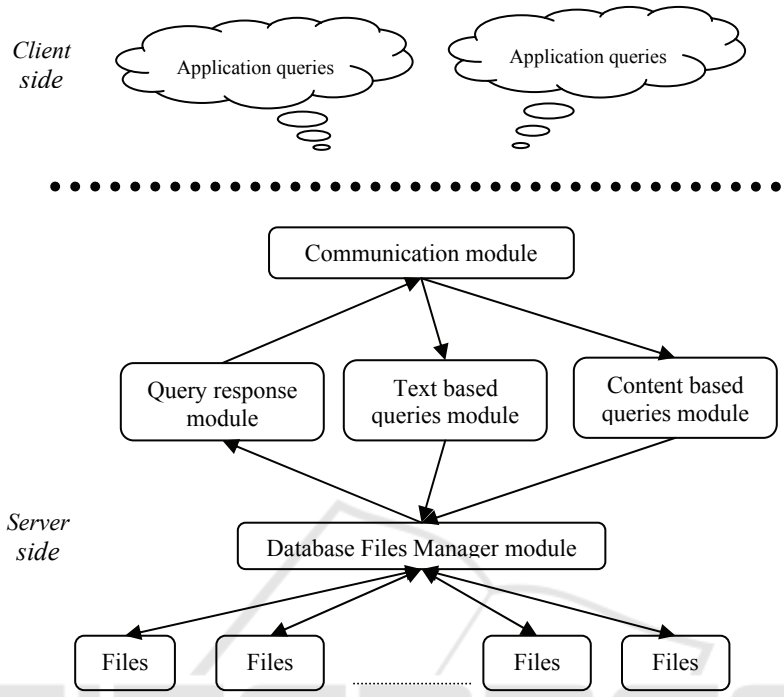

Fig. 1. Multimedia Database Server design.

There can be two kinds of queries:

- Classical text based queries
- Visual content based queries

When the command is received, it is checked what type of query is, it is extracted the parameters of the query and then calls the specific module to execute it. Each of them is computed by a different module: if the command refers to a text based query, it will be computed by the Text Based Queries Module. If it is a content based retrieval query it will be sent for computation to the specialized Content Based Retrieval Module.

This module extracts the parameters from the query and then search in the database files for specific information. If the query is a SELECT IMAGE query, it will use for comparison the similitude of characteristics instead equality of parameters. The parameters used by this module are color histogram and texture characteristics. When the image used as a query image is not already in the database it is needed first to be processed. Before executing the query there are called the functions specialized to extract the color and texture characteristics of the image. The obtained data will be used to initialize an attribute of a special data type called IMAGE.

This type can be used to store all the information regarding the image: color characteristics, texture characteristics, width, height, etc.

Other specialized functions included in this module are:

- Delete Function. It is called when the user executes a DELETE command. The kernel executes only logic deletes. It never executes physical deletes. The physical deletes are executed only when a "Compact Database" command is sent by the user.
- Update functions. If the query received from the user is an UPDATE command, it will be called the Update functions to execute it.

Another important module is the Database Files Manager Module. It is the only module that has access for reads and writes to the files in the database. It is his job to search for information in the files, to read and write into files and to manage locks over databases. It receives read/write requests from the Text Based Query and Content Based Retrieval modules and sends responses to the Query Response module.

When a request to read form a file arrives, it is enabled a read lock to the specific file (that represents a table in the database). All other read requests will be permitted but no writes will be allowed. If the client module request a write to file, it will be enabled a write lock. No other requests will be allowed until the lock is canceled.

The files data read and write operations are not structured in any way. This module does not modify the structure of the result in any way. All the results will be raw data, as they are read from the files or received from client modules. The results will be structured in the Query Response module in a standard manner that can be understood by the client. Only after this operation is finished, the result is sent to the Communication Module to be returned to the client.

3 Content based Retrieval and Image Data Type

An element of originality of this server is the existence of an image processing module and a special data type called Image.

The image processing module will process the images before being used, extracting the color and texture characteristics. The results along with the image in binary will be stored in an attribute of Image data type.

The color is the visual feature immediately perceived on an image [1,2]. In content-based visual query on color feature is important the used color space and the level of quantization, meaning the maximum number of colors. This implementation uses the representation of images in the HSV color space that has the properties of being complete, compact, natural and uniform and its quantization to 166 colors [1].

The color histograms represent the traditional method of describing the color properties of the images. They have the advantages of easy computation and up to certain point are insensitive to camera rotating, zooming, and changes in image resolution. The quantization algorithm generates a characteristics vector of maximum 166 values and has the complexity O(width*height) where width and height represent the image dimensions. For computing the distance between the color histograms of the query image and the target image, the intersection of the histograms is used [1].

Together with color, texture is a powerful characteristic of an image, present in nature and medical images, where a disease can be indicated by changes in the color and texture of a tissue. A series of methods have been studied to extract texture

feature [3,2]. Among the most representatives methods of texture detection are the co-occurrence matrices, a method that have been also implemented in this server.

In this case, one matrix was computed for each of the three channels R, G, B. For an image f(x, y), the co-occurrence matrix $h_{d\phi}$ (i, j) is defined so that each entry (i, j) is equal to the number of times for that $f(x_1,y_1) = i$ and $f(x_2,y_2) = j$, where $(x_2,y_2) = (x_1,y_1) + (d\cos\phi, d\sin\phi)$. This leads to three quadratic matrices of dimension equal to the number of the color levels presented in an image for each distance d and orientation ϕ. The classification of texture is based on the characteristics extracted from the co-occurrence matrix: energy, entropy, maximum probability, contrast, inverse difference moment and correlation. The three vectors of texture characteristics extracted from the three occurrence matrices are created using the 6 characteristics computed for d=1 and ϕ=0. The texture representation in this case is done using a characteristics vector with 18 values stored in the database.

The results of these computations are stored in a variable of Image Data type. The color information is stored as a vector with 166 values and it is used furthermore in the content-based image query and content-based region query.

After the algorithm's execution, the resulted image texture feature will be an 18-dimension vector that is also added to the IMAGE variable.

The results of the content-based visual query process, using the color histograms with 166 values in HSV color space and Gabor Filters were presented in [4,5].

4 Client-Server Working Example

The first step is to create a connection with the server using TCP sockets communication. Once a channel has been open, all commands will be sent using this connection. Now, the server has accepted the client but cannot accept any command from it, yet. Before that, the client should be authenticated using a username and a password.

The command that should be sent first is an array with the text:

```
login username="username" password="password"
```

After receiving this message the server respond either with an acknowledge message, either with a reject message. In case of a reject message, the user has the possibility to try again two more times. If it doesn't get an acknowledge message after these tries, it will be banned for an hour. During this time, even if the client sends a connection message containing a valid username and password it will be rejected.

Once the client has been recognized by the server it will received a unique identification number that will be de-allocated only when the client closes the connection or send a logout message.

There are 2 kinds of commands that it can send: system commands and query commands.

The system commands refer to create new users, set/change users rights or get tables structure (attributes and the associated data types).

Among the most useful query commands used it can be enumerated:

- Create table. The available data types are: int, double, varchar and image. The command returns “ok” if it was successfully executed, or “fail” in case the table could not be created (eg: table already exists)
  `create table <table_name> (first_attribute type1, second_ attribute type2,…)`
- Adding constraints (primary keys, foreign keys). The command returns “ok” if it was successfully executed, or “fail” in case of an error.
  `alter table <table_name> add primary key (<column_ name>)`
  `alter table <table_name> add foreign key (<column_ name>)references <refered_table> ( <refered_column_name >)`
- Inset data into tables. The command returns “ok” if it was successfully executed or “fail” in case the table does not exist, one of the attributes has a wrong data type or a constraint didn’t allow the insertion.
  `insert into table <nume_tabel>values(val1,val2, val3,…)`

An element of originality is the possibility to add images directly in the database. If one of the columns has the image type, the insertion in the database it is a little more complicated because the client has to send the image and the server has to process it. After sending this command the server expects to receive from the client a “send image” command, followed by the image itself. Then the server calls the image processing functions to extract the image’s characteristics: color and texture.

The extracted values along with the image name, size and image in binary are stored in an attribute of a special type, called image. On the disk, the server stores all the information about image in the table data file. The image is stored in a separate file used only for images. In the table data file will be stored only a pointer with the position of the image in the images file.

- Delete records. It returns true if succeeded or “fail” if not. The server executes only logical deletes. That means if the server executes multiple insert/delete operations the size of the database will increase continuously. The server offers a solution for this problem by compacting the database. This operation needs a significant amount of time. That is why it should be executed offline, only when the server is less used.
  `Delete from <table_name> where <column_name>=<value>`
- Select records. It returns a recordset containing the records that are according to the condition specified in the where clause, or an empty pointer if there are no records that fulfill the condition.
  `Select * from <table_name> where <column_name>=<value>`
- Select images from database. In case that the where clause refer to an image attribute, the server will execute a content based retrieval. In this case we cannot discuss about equality between images, but similarities. The server will compare all the images in the table that has characteristics appropriate with the characteristics of the query image.
  `Select image from <table_name>.<column_name> where like <query_image>`

5 Conclusions

The paper presented a multimedia database management server used to manage medium sized multimedia databases. An element of originality is the possibility to store images directly in the database in a special data type called image. It stores information about image name, color characteristics, texture characteristics, image size and image name.

The software tool allows creating and deleting databases, creating and deleting tables in databases, updating data in tables and querying.

The client has the possibility to execute both simple text based queries and visual content based retrieval queries. In the second case the user selects one image from the database as a query image and the server returns all the images similar with it.

The color characteristics are represented by extracting color histogram from images. The texture characteristics are computed using co-occurrence matrices. Using these characteristics, the similitude between images is computed using the histogram intersection and the Euclidian distance. The complexity of both distances are equally O(m*n) where m is the number of values in the characteristics vector, and n is the number of images in the database.

This software can be extended in the following directions:

- Adding new types of traditional and multimedia data types
- Studying and implementing indexing algorithms for data inserted in the tables
- Also, a parallel computation of the two distances can be proposed in order to make the execution time for a query shorter.

Acknowledgements

This research was partially supported by the Romanian National University Research Council under the PCE Grant No. 597.

References

1. Del Bimbo, A.: Visual Information Retrieval. San Francisco: Morgan Kaufmann Publishers (2001)
2. Smith, J.: Integrated Spatial and Feature Image Systems: Retrieval, Compression and Analysis. Ph.D. thesis, Graduate School of Arts and Sciences, Columbia University (1997)
3. Zhang, D., Wong, A., Infrawan, M., & Lu, G.: Content-Based Image Retrieval Using Gabor Texture Features. The First IEEE Pacific-Rim Conference on Multimedia, pp. 392-395, Sydney (2000)
4. Gevers, T.: Image Search Engines: An Overview. Emerging Topics in Computer Vision. Prentice Hall (2004)
5. Stanescu, L., Burdescu, D., Brezovan, M., & Stoica Spahiu, C.: A New Software Tool For Managing and Querying the Personal Medical Digital Imagery. *Proceedings of the International Conference on Health Informatics*, (pp. 199-204). Porto – Portugal (2009)
6. Chigrik, A. (2007), SQL Server 2000 vs Oracle 9i, http://www.mssqlcity.com/Articles/Compare/sql_server_vs_oracle.htm