Proceedings of the 7th International Conference on Applied Informatics
Eger, Hungary, January 28–31, 2007. Vol. 1. pp. 343–351.

Enhancing content-based image retrieval

Lajos Kollár, Krisztián Veréb

Department of Information Technology
Faculty of Informatics
University of Debrecen
e-mail: {kollarl,sparrow}@inf.unideb.hu

Abstract

Content-based image retrieval is a relatively recently developed field of computer science. Its main goal is to provide methods for finding similar images to a given input image. In the literature, there exist such methods which operate on images by automatically extracting and comparing the values of some predefined features, such as colour, shape and texture.

However, in real world, an image might have features which cannot be predefined in general but they can also automatically extracted by using appropriate algorithms. For example, some images contain text, others contain faces (or both), etc. These are pieces of information which can easily be extracted by a human being who is asked to describe the picture. That is not the case when we replace humans with computers.

In this paper, we propose a method which is flexible and easy-to-extend to capture image semantics using image processing algorithms defined in an image database built upon Oracle Database 10*g*. Extracted pieces of this semantical information are represented in XML documents and serve as a starting point for enhancing the content-based retrieval of images.

Keywords: Image databases, feature vectors, content-based image retrieval, XML, Oracle

1. Introduction

Nowadays, as digital image, video and sound recording devices has become cheaper and cheaper, more and more people could afford them which caused the amount of digital multimedia data be multiplied. Many domain-specific applications requiring an archive of multimedia data have come into existence, e.g. an editorial archive of a news portal should contain lots of images and videos about specific events and may also contain recordings of interviews. Besides, as the result of the success of the World Wide Web, a huge amount of such data can be accessed by public, as well. One can think of the Web as a very big multimedia archive.

The need for searching those multimedia data by content has been arisen. For example, the author of an article on informing readers about an upcoming international conference on applied informatics may want to illustrate the article by including some photos made on the previous conferences. In this case, she or he either needs the names of files containing images to be descriptive enough or a mechanism is required for defining a search expression serving as a basis for (semi-)automatic retrieving of the appropriate images. Information sources natively supporting the storage and retrieval of multimedia data are called multimedia databases.

For the sake of simplicity, we only consider a subset of multimedia databases, namely, image databases. Nevertheless, our basic idea may be applied to other types of multimedia databases, such as video or sound databases, as well.

This paper is organised as follows. In Section 2, we introduce the main concepts of content-based image retrieval (CBIR) known from the literature and show the lack of flexibility regarding existing CBIR systems. Our proposal of enhancing content-based image retrieval using XML for storing image metadata is described in Section 3. Finally, Section 4 presents the current state of development and our future plans.

2. Content-based image retrieval

Content-based image retrieval is used for finding similar images (in some manner) to a given image. Keyword-based search and information extracted from the image can serve as a basis of content-based image retrieval.

Users may attach some metadata to images when uploading them to the database. In case of a CD cover image it may mean information about the performer, genre, title and length of tracks, year of publication, etc. Keyword-based search has nothing to do with the images themselves, it means only searching among metadata.

Feature vectors can be extracted from images using the well-known algorithms from the field of digital image processing [11]. These may include the global histogram of the image, or determining colours, shapes, structure and texture of objects in the image which can later be used during a comparision. The process of content-based image retrieval based on feature extraction consists of three steps:

1. The values of some predefined features should be extracted from images, which define the properties that are used for comparing images. For instance, if we are looking for images containing similar colours to the given one, colour should be extracted. Other examples of features well-known in the literature are shape and texture.

2. A similarity measure needs to be defined, which is used to determine the distance between two features.

3. Based on this measure, an ordering should be done which tells us which image is the most similar to the given image in the sense of the given features and

measure.

It is also very common to define a threshold, as well, which is interpreted as if the distance of an arbitrary image and the given image exceeds this threshold then they are treated as being not similar at all.

In [10], such a content-based image retrieval system using a hierarchical indexing strategy and applying fuzzy connectives for defining the measure [1] has been introduced. This system uses Oracle Database as a back-end for storing images and Oracle's *inter*Media cartridge [2] for the basic image processing operations. Our work is mainly based on this legacy system.

2.1. Problems

When building an image database, a user interface should be presented to the user who is loading the images into the database. If the database is intended to use in a specific application domain, e.g. it should contain CD covers only, then users may be supplied with an input form, as well, in order to be able to enter some metadata. In this case, one may think of a form containing text fields for the performer, title of album and its songs, genre, year of publication and so on. Actually, these are not metadata about the CD cover (since it does not say anything about the cover image) but metadata about the underlying CD itself. However, if we need to answer such questions that "how does the cover of the album containing a song entitled «Californication» look like", it is easily understandable that we can consider these pieces of information as a metadata of the cover, in a broader sense. Unfortunately, in the most of the cases, song titles cannot be extracted from the cover image even by applying the best image processing algorithms (since it does not contain song titles) so this example illustrates that we also need to handle user-supplied metadata. Application domain specific data can easily represented in a relational or object-relational database table since they have a well-defined structure.

On the other hand, the result of the execution of an optical character recognition (OCR) algorithm on the cover image is some textual information contained by the image. (Let us disregard the possible errors of the OCR algorithm since its precision and/or performance may be increased by changing it to a better one.) Using this text, such questions can be answered as "which are those images that contain the given text" and "which are those images that contain/does not contain any text at all". By applying other algorithms, different kind of information may be extracted even from the same image so this is a semi-structured information which is hard to represent in a relational environment.

If the usage of the image database is not bound to a specific application domain (e.g. in the case of a general-purpose system), it is very difficult to build a user interface for user-supplied metadata since its structure is not known. The most commom solution in such cases is to provide the user with a text area where arbitrary textual data can be written. Unfortunately, this hides the structure of the information and only allows free-text search which decreases efficiency.

This CD cover database is a very special one since from the viewpoint of the information of a CD it seems to have a well-defined data structure representing data about the album and its songs but from the cover images' point of view it is a general-purpose system as the content of the image can be arbitrary. That is not the case when having a database of photos on identity cards since both the underlying data structure (name, address, etc.) and the content of the image (a face) is of a specific application domain.

3. Our proposal

Our goal was to propose a method for describing image metadata which is flexible enough for capturing both application-specific (structured) and general-purpose (semi-structured) data. Since it is impossible to create such a system which can produce all metadata about an arbitrary image, we addressed to allow the model easy to extend. Extensibility should be present by the means of adding algorithms to the system since new algorithms produce new pieces of metadata (typically from another aspect). Theoretically, this property ensures that all relevant metadata can be extracted from a given image since if it is not the case then one only has to add a new algorithm to the system which is doing the work.

Another requirement was to develop such a flexible data structure which can collaborate with built-in image representations (in Oracle 10*g*, there are two of them: the ORDImage and the SI_StillImage object types) as well as user-defined representations (e.g. a quad-tree representation of an image). If this data structure is flexible enough, it will ensure both backward and forward compatibility with different Oracle versions.

3.1. Possible solutions

A possible solution for the above mentioned problems is to map each image to an object describing its set of metadata. Such an object should be an instance of a class representing a metadata set extracted by a given algorithm. Each newly added algorithm requires the creation of an underlying class. For example, if we have a jailhouse database of images, each containing the face and the identifier of a prisoner, we may need both a face recognition and an OCR algorithm. When adding them to the database, two classes (Face and Identifier) should be defined. Such a photo of a prisoner needs to be mapped to an object which is an instance of both classes. This requires multiple inheritence which might not be supported by some systems so this is not a general solution.

Another possible solution is to store image metadata in XML documents which are the most common way of representing semi-structured data. In this case, each newly added algorithm should define its own tag set which reflects the data structure of the algorithm domain. In order to avoid ambiguity, each tag set may belong to an algorithm-specific namespace, but using of namespaces is not mandatory. Due to the fact that nowadays practically all the systems can manage XML documents

Figure 1: An example of a CD cover

in a standardized way (e.g. in Oracle there is an implementation of the SQL/XML standard [13]), we propose using of this method.

3.2. Uploading process

The database can be extended by adding images or algorithms. Each time an image is inserted into the database, an XML document representing the metadata structure of the image (to be detailed in Section 3.3) is created along with information about the owner (who inserted the image), creation date, etc. This XML document is stored in the database as an XMLType column.

Algorithms may be written in either PL/SQL or Java since Oracle Database supports stored subprograms of both languages. When algorithms are loaded into the database, they need to be registered at the controller. Controller is a PL/SQL package containing methods for registration and invocation of algorithms. It maintains another XML document (the repository of the registered algorithms). Invocation of an algorithm is performed via controller which provides some infrastructural activities such as managing the corresponding XML documents.

An algorithm is not necessarily applicable to an image since there could be a lack of the required representation. In this case, a warning message should be arisen by the controller. Furthermore, if an algorithm is applicable to a specific representation of an image, it does not follow that the execution will give us a result. For example, application an OCR algorithm to an image containing no textual information should result in a failure meaning that the given image does not have the specified metadata. In such cases, the corresponding XML document remains unchanged.

3.3. Document structure

Figure 2 shows the basic structure of an XML document used by our system. Element `<imagexml>` is the document element having several atributes such as time of document creation or last modification, username, etc.

Recall that each image inserted to the database may have several representations (for special reasons it is allowed to create both built-in representations

```
<imagexml generated="2006-11-17 17:36:04" id="95"
          last-updated="2007-05-10 13:48:52" user="IMGSYS">
  <si-content>
    <si_stillimage><ref>00004A0...00000000</ref>...</si_stillimage>
    <si_featurelist><ref>000028...F098006E</ref>...</si_featurelist>
  </si-content>

  <ord-content>
    <ordimage><ref>00004A038A0046A827...00000000</ref>...</ordimage>
    <ordimagesignature><ref>00002...06E</ref>...</ordimagesignature>
  </ord-content>

  <user-defined>
    <metadata name="text" algorithm="my_ocr">
      a day without rain
    </metadata>
    <metadata name="face" algorithm="my_face_counter">1 </metadata>
  </user-defined>

  <description>
    <artist>Enya</artist>
    <title>A Day Without Rain</title>
    <genre>New Age</genre>
    <tracks>
      <track no="1">
        <title>A Day Without Rain</title><time>4:18</time>
      </track>
       ...
      <track no="11">
        <title>Lazy Days</title><time>3:43</time>
      </track>
    </tracks>
  </description>
</imagexml>
```

Figure 2: XML document representing the image of Figure 1

along with some user-defined ones at the same time). All of these representations are instances of object types, and feature vectors are extracted by invoking a method (algorithm) of the object (the given representation). These objects are represented in the XML document using their serialized object reference by which object can be accessed but here (due to the lack of space) we would not go into details. Naturally, each representation (either built-in or user-defined) has a corresponding serialized object reference. For built-in representations, there are to elements (<si-content> and <ord-content>). Under <si-content>, one can find a sequence of elements corresponding to the SI_StillImage and SI_FeatureList objects (the latter is a composition of other objects representing four basic image features, namely, SI_AverageColor, SI_ColorHistogram, SI_PositionalColor and SI_Texture) as defined in the SQL/MM standard [12]. All of these elements have a child element called <ref> containing the serialized reference. The structure of the element <ord-content> is similar, only names of children differ: <ordimage> and <ordimagesignature>.

All user-defined metadata which are extracted by an algorithm written by the user are enclosed by the <user-defined> element which is a direct child of the document element. Under this element, there is a list of <metadata> elements, each of which contains the name of algorithm, the name of metadata (defined by the algorithm) and the value extracted as the result of the invocation of the algorithm. Each invocation of an algorithm may cause this part of the XML document to be dinamically changed. In order to hide the technical details of handling XML documents from algorithm developers, controller will do this work. An execution might be initiated by calling the method `invoke` of the controller API (with proper parametrization). If the operation succeeds then it is the controller's responsibility to add a new <metadata> element to the XML document.

Element <description> might be used to handle user-supplied metadata. As the content model of that element is mixed, structure of data might be captured, as well. Users may need tool support for authoring such XML document fragments as we discuss it in Section 4. This solution clearly separates system-managed and user-managed data.

Let us suppose we have already inserted Figures 1 and 3 into the database along with two algorithms, `my_ocr` (doing an optical character recognition) and `my_face_counter` (counting the number of faces in a picture). After registering these algorithms at the controller, they might be invoked using controller's `invoke` method. After executing `my_face_counter` on both of the images, controller will update the corresponding XML documents to reflect that the algorithm has found faces in both images and counted them. When execution of `my_ocr` is requested for both images, due to the lack of textual information in Figure 3, its XML representation (Figure 4) remains untouched while controller updates the document of Figure 1 to include the second metadata element as shown in Figure 2.

Figure 3: An image without textual information

```
<imagexml generated="2006-12-22 10:25:12" id="102"
          last-updated="2007-05-10 13:50:34" user="IMGSYS">
  <si-content>...</si-content>

  <ord-content>...</ord-content>

  <user-defined>
    <metadata name="face" algorithm="my_face_counter">1</metadata>
  </user-defined>

  <description>John Travolta</description>
</imagexml>
```


Figure 4: XML document representing the image of Figure 3

4. Future work

The system is currently in development phase but our opinion is that automatic extraction of image semantics could be a step towards building more efficient image retrieval systems [7]. Naturally, a lot of work should be done in order to ensure that our method is flexible and efficient enough. First of all, we need a mechanism for defining which algorithms must be executed automatically when an image is inserted into th database. Having many and/or complex algorithms may result in a loss of performance. Since our system is mainly built up using stored procedures, a user interface is also needed for accessing system functionality in a non-programmatic manner [8, 9]. Users should be supplied with an XML authoring tool, as well, in order to define structured domain-specific information. As our implementation is bound to Oracle Database, it can only improve its capabilities for enhanced image management but our idea can be applied to other image databases, as well.

References

[1] Veréb, K., On a Hierarchical Indexing Fuzzy Content-Based Image Retrieval Approach, CEUR Vol.: 76, *Proc. of the VLDB 2003*, PhD Workshop, Berlin, Germany, (2003).

[2] Oracle *inter*Media User's Guide, 10*g* Release 2 (10.2), Part Number: B14302-01, Oracle Co., http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14302.pdf

[3] Oracle *inter*Media Reference, 10*g* Release 2 (10.2), Part Number: B14297-01, Oracle Co., http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14297.pdf

[4] VERÉB, K., Tartalomalapú képkinyerés képarchívumokból – egy lehetséges megoldás (in Hungarian), *NetworkShop 2004,* https://nws.niif.hu/ncd2004/docs/ehu/037.pdf

[5] HAJDU, A., TÓTH, T., VERÉB, K., ZÖRGŐ, Z., Distance functions in multidimensional image processing applications, *Proc. of KÉPAF4,* (2004), 106–111.

[6] Oracle XML DB Developer's Guide 10*g* Release 2 (10.2), Part Number: B14259-02, Oracle Co., http://download-uk.oracle.com/docs/cd/B19306_01/appdev.102/b14259.pdf

[7] SANTINI, S., Exploratory Image Databases, Content-Based Retrieval, *Academic Press,* (2001).

[8] CASTELLI, V., BERGMAN, L. D. (editors), Image Databases–Search and Retrieval of Digital Imagery, *John Wiley and Sons,* (2002).

[9] LEW, M. S. (editor), Principles of Visual Information Retrieval, *Springer,* (2001).

[10] VERÉB, K., Vizuális információkinyerés és tartalomalapú képkinyerési technikák képadatbázisokban, PhD Thesis (in Hungarian), University of Debrecen, (2005).

[11] SONKA, M., HLAVAC, V., BOYLE, R., Image Processing, Analysis and Machine Vision, Second Edition, *PWS Publishing,* (1998).

[12] EISENBERG, A., MELTON, J., SQL Multimedia and Application Packages (SQL/MM), *ACM Sigmod Record,* 30 (4), December, (2001), 97–102.

[13] EISENBERG, A., MELTON, J., SQL/XML is Making Good Progress, *ACM Sigmod Record,* 31 (2), (June, 2002), 101–108.

Lajos Kollár, Krisztián Veréb
H–4032 Debrecen, Egyetem tér 1.
Hungary