

Content-based Image Retrieval for Connected Mobile Devices

Moncef Gabbouj, Iftikhar Ahmad, Malik Yasir Amin and Serkan Kiranyaz

Abstract—Mobile device usage is increasing nowadays as the processing power and the memory of mobile devices are increasing along with new emerging network technologies promising a higher service quality, larger and cheaper bandwidth. Recent multimedia mobile devices are equipped with camera and higher storage capabilities; this urges the need of efficient multimedia content management over various networks. In this paper a client-server architecture for content-based image retrieval framework over connected mobile devices is proposed. The server application performs the query operation and generates hyper text markup language pages. Internet browser is used on the mobile device to view these pages. Experiments show that such a scheme is feasible and provides a generic solution for the content-based image retrieval from connected mobile devices.

I. INTRODUCTION

MODERN multimedia mobile devices with integrated support of camera provide new multimedia services, which are becoming an essential part of our daily life. 3G [5] networks are already in the market and offers new services. With new multimedia mobile devices users can generate and consume multimedia (image/audio/video) items on the fly and hence this causes a rapid growth for the personal multimedia collections. Consequently it brings the needs and ways for the efficient management and accessibility of such personal archives. Scientists and researchers have undertaken the challenge of meeting user requirements for content management over mobile devices.

The multimedia item name that is automatically assigned by the capturing software on the mobile device is not so useful for media content recognition. Annotation of digital media at the time of capture is a cumbersome process for a mobile user, which is the basis of metadata creation process [10]. Systems such as IDEixis [11] is content-based image retrieval for location based services but the users usually experience a large latency because of an inadequate mechanism for image transport used in IDEixis, which is

based on Multimedia Message Service (MMS). Therefore, without proper content-based management methodologies applied, it is not quite feasible to find a particular media item whenever needed. Furthermore, due to ever-growing today's digital multimedia era on the mobile devices with limited storage, users have to move their multimedia collection into some other storage medium, most probably to their computers or peripheral hard-drives. Accessing such displaced multimedia items from a mobile device further contributes to the challenge of multimedia management for the users of mobile devices.

Content-Based Image Retrieval (CBIR) is an active area of research for past decade due to its promising results. In CBIR various low-level features color, texture, etc. are used for dis-similarity distance calculation. Low-level visual features (descriptors) are extracted from the images and stored in a database. Using such features, query by example (QBE) based retrieval performs relatively well for images. Systems such as "Multimedia Video Indexing and Retrieval System" (*MUVIS*) [16], VisualSEEk [9], Photobook [8], Virage [21] have common feature of having a framework for indexing and retrieving over multimedia databases. Particularly the contemporary *MUVIS* is developed as a framework for content-based multimedia indexing and retrieval on a PC-based environment. *MUVIS* provides a unified and global framework and consists of robust set of applications for capturing, recording, content-based indexing and retrieval, combined with browsing and various other visual and semantic capabilities.

With the help of camera in mobile device user can capture an image and perform content-based query operation virtually from anywhere. However CBIR for mobile devices adds new challenges beside a content-based query operation. For instance different mobile devices come in different design and capabilities; moreover, they have different operating systems and input/output limitations. So it is hard to provide a generic content-based multimedia indexing and retrieval solution that suites all devices. However, with the help of Java Server Pages (JSP) [6] we can generate Hyper Text Markup Language (HTML) pages, which are dynamically adaptable to different user interfaces for different devices.

Recently the capabilities of mobile devices have improved significantly (faster input/output, memory capacity and processing) but comparatively they are still lacking far behind the desktop computers. With the existing mobile operating systems such as Symbian OS [19], Window CE [13], Linux, etc. it is possible to index a multimedia database

Professor Moncef Gabbouj is the Head of the Institute of Signal Processing in Tampere University of Technology, P.O.Box 553, FIN-33101, Tampere 33720, Finland, (e-mail: moncef.gabbouj@tut.fi)

Iftikhar Ahmad is with Nokia Corporation, P.O.Box 88, FIN-33721, (Tieteenkatu 1, Tampere 33720, Finland); e-mail: iftikhar.ahmad@nokia.com.

Malik Y. Amin is a master degree student in University of Tampere, Department of Computer Sciences, Kanslerinrinne 1, FIN-33014, Tampere, Finland. (e-mail: malik.amin@uta.fi).

Dr. Serkan Kiranyaz is a Senior Researcher at the Signal Processing Laboratory, Tampere University of Technology, P.O.Box 553, FIN-33101, Tampere 33720, Finland, (e-mail: serkan@cs.tut.fi).

entirely in a mobile device [3] but there are many limitations. For example mobile devices have propriety Application Programming Interfaces (APIs) for handling (i.e. accessing, processing, editing, streaming, etc.) of multimedia items. Applications using such proprietary APIs will be limited to certain set of devices or certain platforms (operating systems). Another limitation is battery power consumption and lack of system resources such as processing power and memory capacity. Above all with the ever-increasing number of multimedia items on a mobile device, it might take an infeasible amount of time to perform content-based indexing and retrieval operations due to lack of speed and memory. Furthermore, such a system would consume significant battery power that eventually reduces the mobile device talk and standby time. Therefore, a feasible solution to this problem could be performing all the resource-taking indexing tasks on powerful machines (i.e. server in a computer) whilst the mobile devices act as their clients. In our earlier work *M-MUVIS* [2] is proposed as a client-server architecture with Java Servlets [6] on the server side and a stand-alone Java [15] application on the client side. It consists of Servlets which are deployed over Tom-Cat [20] web server active on the PC and the Java application, so called midlet [17], which can be deployed on any Java enabled device.

In this paper, we present an efficient CBIR framework, which uses combination of low-level features for image retrieval as compared to previous retrieval schemes, which only work over a single feature for content-based retrieval [1]. The proposed framework, Java Server Pages on *MUVIS* (*JspMuvvis*) is also a client-server architecture, where server is active on Personal Computer (PC) and an Internet browser application acts as client on the mobile devices. It is based on contemporary Mobile-MUVIS (*M-MUVIS*) framework [2] but *JspMuvvis* differs from *M-MUVIS* in a way that users can now use the Internet browser on a mobile device for initiating a content-based query operation along with displaying the retrieval results, and Java is not required on the client side at all. Therefore, such a system provides a more generic solution, particularly for the mobile phones without Java enabled.

Rest of the paper is organized as follows. Section II provides an overview of *JspMuvvis*. Experimental results are presented in section III and finally conclusions are drawn in section IV.

II. JSPMUVIS OVERVIEW

The access and use of web browser is quite habitual and periodic these days. In the proposed framework *JspMuvvis* server generate HTML pages dynamically for the mobile devices. Mobile devices can use a Web or WAP browser to display those HTML pages.

As stated earlier *JspMuvvis* has client-server architecture. *JspMuvvis* server (Tom Cat web server) is kept active on a powerful computer. The client and the server communicate

over Hypertext Transfer Protocol HTTP [12]. HTTP is a stateless protocol so a session [6] is created on the server side to keep track the client information (query type, mobile device screen size, etc.). The overview of *JspMuvvis* framework is shown in Figure 1.

In *JspMuvvis*, session is an object created and maintained by Tom-Cat web server on the server side. Session persists across all connections from client to server till the user terminates the Internet browser (i.e. the client application) on the mobile device. Server assigns a unique identification number to each session and passes that number to the client. The client uses that number for each connection established with the server. The server can therefore identify a particular client and uses its session information for any requested operation. In *JspMuvvis* one client can have only one session, after the completion of a content-based query operation, the retrieval results are also integrated into the session stream in case any client can request these results later.

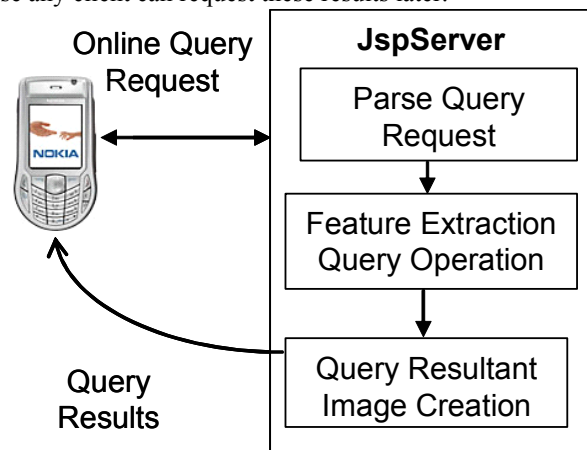


Figure 1: The Overview of *JspMuvvis* Framework

Client-server communication needed for a query operation is shown in Figure 2.

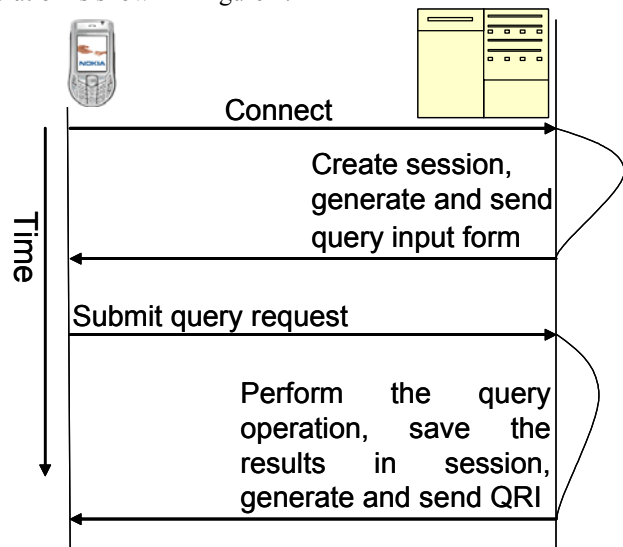


Figure 2: JspMUVIS client-server communication.

There are basically three layers in *JspMuvvis*:

A. Presentation Layer

This is the user interface layer of *JspMuvis*. HTML pages that are generated by the *JspMuvis* server contain the user input form and the optional settings for a content-based query operation described below. Presentation layer is responsible for interacting with the user. The user has the following options in this layer as one particular example displayed in Figure 3.

- **Database:** A server side option to select a particular *JspMuvis* database among the list of available databases.
- **Query Type:** Selection for either “Random Query”, or “Query By Image”.
- **Search:** Selection of the query type.
- **Query Media:** User can provide a query image in case “Query by Image” option is chosen.

In a “Random Query” operation, query is performed after picking an image randomly among the images in the active database on the server side. “Query by Image” is a typical Query by Example (QBE) process where the user selects a particular image (via “Query Media” edit box in Figure 3) and initiate a query operation. The query image is then uploaded to the server, the query process is performed and the retrieval results are streamed to client and shown to the user in an Internet browser as one particular example is shown in Figure 4.

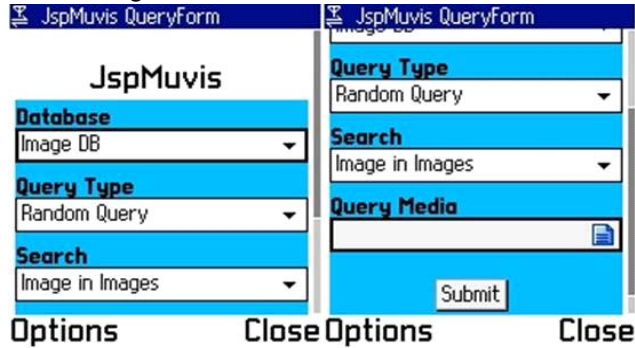


Figure 3: User settings for content-based query



Figure 4: A sample QRI in an Internet browser on a mobile device

The presentation layer is also responsible for displaying

the query results. Considering the small screen size of mobile devices query results are displayed as thumbnails. A “Query Resultant Image” (*QRI*) is created on the server side where image thumbnails of the retrieval results are drawn. An example *QRI* is shown in Figure 4.

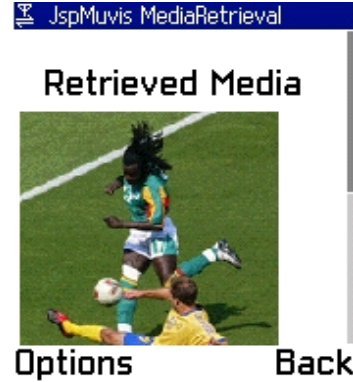


Figure 5: Retrieved image shown in an internet browser on a mobile device

The user can select any thumbnail among the images in a *QRI* and the client can fetch the original image (with real dimensions) from the server as shown in the Figure 5.

B. Application Layer

In this layer the necessary coordination between presentation layer and data layer is organized. Application layer is responsible for using the data layer according to the user selections in presentation layer. It captures the query information from user input form via presentation layer, uploads it with the selected image to the *JspMuvis* server and uses the data layer to perform the query operation. As mentioned earlier, a session is created between client and server. Application layer is also responsible for session tracking of a particular *JspMuvis* client. It uses cookies [6] to store the session information on the mobile device, and Internet browser picks the session information during the beginning of a client session.

C. Data Layer

Data layer is responsible mainly for database operations. *JspMuvis* uses the APIs developed in *M-MUVIS* project to access the database through native (C/C++) code. Native libraries are used for efficient content-based image query related operations. *JspMuvis* contains Java beans [15], which handle the operations such as activating the selected database within application, performing the content-based query operation, retrieving the 12 best results embedded into *QRI*.

III. EXPERIMENTAL RESULTS

The sample database used in our experiments contains 476 images in different formats. *JspMuvis* server is activated on a PC equipped with P4 2.99GHz and 1.9GB of RAM. Basic visual features such as YUV, HSV, RGB color histograms, Gray Level Co-Occurrence Matrix (GLCM) as a texture feature [7] are extracted for the sample image database

operated by *JspMuvis* server.

Table 1: Mean and SD of CQT and SQT for different network

Connection Type	CQT		SQT	
	Mean (ms)	SD (ms)	Mean (ms)	SD (ms)
3G	7100	567	2118	11
EDGE	7628	479	2118	7
WLAN	3462	158	2122	6

A content-based query operation is initiated from a client device to retrieve the similar images with respect to a query image. Server Query Time (*SQT*) is measured as the time spent to perform a query operation on the server side whereas Client Query Time (*CQT*) is the entire time passed from sending a query request, performing the query on the server side, formation of the *QRI* on server side and the to the reception of *QRI* to the client. Statistics (mean and Standard Deviation (*SD*)) of *CQT* and *SQT* are presented in Table 1. Each mean and SD is calculated over 12 query operations. *CQT* is significantly high as shown in Table 1 but a direct comparison with *SQT* approves that this is due to the network conditions. The advances in mobile network technologies in the near future might close the gap between *CQT* and *SQT*. Furthermore, some reduction methods on network traffic such as session tracking [6] can be used to minimize the flow of data between mobile device and the server.

As stated earlier *QRI* as a JPEG (Joint Photographic Experts Group) image is created on the server side to embed the query results and display all in once in the browser GUI on a mobile device. Since today's mobile devices usually do not support high-resolution screens; a low quality factor in JPEG format can be used conveniently for the *QRI* creation in order to reduce the *QRI* size and thus in effect to reduce the transmission period. During our experiments we observed that JPEG quality factor 0.20 to 0.25 is enough to display the *QRI* without a significant degradation in visual perception for most of the mobile devices.

IV. CONCLUSIONS

A novel CBIR framework for the connected mobile devices is presented for the efficient content management of personal multimedia archives from connected mobile devices. *JspMuvis* is designed as a web application, which makes it easier to be used across a range of mobile devices. The reduction of network bandwidth usage and the consumption of processor resources are two of the major factors in the success of mobile device applications. *JspMuvis* reduces the latter via performing the operations requiring a significant CPU power and memory on the server side, which is activated on a powerful PC. With the help of session tracking we further reduced the excessive network traffic between client and server.

Currently, *JspMuvis* server acts only as a CBIR server.

The future plans are to upgrade it as a fully-functioning content-based multimedia indexing and retrieval server. Furthermore by using metric access method based indexing algorithms particularly designed for multimedia databases on the server side, *CQT* can be reduced especially for large multimedia databases.

REFERENCES

- [1] I. Ahmad, F. Alaya Cheikh, B. Cramariuc and M. Gabbouj, "Query by Image Content using NOKIA 9210 Communicator", Proc. of the Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS'01, pp.133-137, Tampere, Finland, May 2001.
- [2] I. Ahmad, Moncef Gabbouj, "Compression and Network Effect on Content-Based Image Retrieval on Java Enabled Mobile Devices" FINSIG'05, pp35-38, University of Kuopio, Finland, August 2005.
- [3] O. Guldogan, M. Gabbouj, "Content-based image indexing and retrieval framework on symbian based mobile platform", European Signal Processing Conference, EUSIPCO 2005, Antalya, Turkey, Sep. 2005.
- [4] S. Kiranyaz, "Advanced Techniques for Content-Based Management of Multimedia Databases", PhD Thesis, Publication 541, Tampere University of Technology. Tampere, Finland
- [5] J. Lempiäinen, M. Manninen, Radio Interface System Planning for GSM/GPRS/UMTS, Published by Kluwer Academic.
- [6] Sing Li, Paul Houle, Mark Wilcox, Ron Phillips, Piroz Mohseni, Stefan Zeiger, Hans Bergsten, Matthew Ferris, Danny Ayers, "Professional Java Server Programming", published by Peer Information Inc., ISBN: 1861002777.
- [7] M. Partio, B. Cramariuc, M. Gabbouj, A. Visa, "Rock Texture Retrieval Using Gray Level Co-occurrence Matrix", In Proc. of 5th Nordic Signal Processing Symposium, October 2002.
- [8] A. Pentland, R.W. Picard, S. Sclaroff, "Photobook: tools for content based manipulation of image databases", Proc SPIE (Storage and Retrieval for Image and Video Databases II) 2185:34-37, 1994.
- [9] J.R. Smith and Chang, "Visual SEEK: A fully automated content-based image query system", ACM Multimedia, Boston, Nov. 1996.
- [10] R. Sarvas, E.Herrarte, A.Wilhelm, and M.Davis, "Metadata Creation System for Mobile Images", Proc. of the 2nd international conference on Mobile systems, applications, and services, MobiSys, Boston USA, Pages: 36-48, June 2004.
- [11] K.Tollmar, T.Yeh and T.Darrell, "IDEixis: image-based Deixis for finding location-based information", Mobile HCI, Vienna, Austria, Pages: 781 - 782, 2004.
- [12] Clinton Wong, HTTP Pocket Reference, published by O'Reilly.
- [13] Douglas Boling, Programming Windows CE, 3rd Edition, by Microsoft Press; 3rd Bk&Cdr edition (June 25, 2003), ISBN: 0735618844.
- [14] "3rd Generation Partnership Project (3GPP)", <http://www.3gpp.org/About/about.htm>.
- [15] "Java" <http://java.sun.com/>
- [16] "MUVIS" <http://muvis.cs.tut.fi>
- [17] "Midlet" <http://java.sun.com/j2me>
- [18] "Nokia", <http://www.nokia.com>
- [19] "Symbian OS", <http://www.symbian.com>
- [20] "TomCat" <http://jakarta.apache.org/tomcat/>
- [21] "Virage", <http://www.virage.com>