

A generic content-based image retrieval framework for mobile devices

Iftikhar Ahmad · Moncef Gabbouj

© Springer Science+Business Media, LLC 2010

Abstract Multimedia mobile devices have created new possibilities for developing and accessing a variety of multimedia items such as images, audio and video clips. Personal multimedia items are, nowadays, being consumed at an enormous rate. Therefore, the management of these media items has become a pressing problem. In this paper, a client-server content-based image retrieval framework for mobile platforms is developed, which provides the capability of content-based query and browsing from mobile devices. The proposed framework provides an adaptive user interface and a generic structure, which supports a wide range of mobile devices. In this framework, a client requests the server for retrieval of particular images with a particular content. The server performs a content-based retrieval of images from a selected database and streams the retrieved results back to the client in an efficient way. The query results are transmitted over a wireless network and a certain number of similar images are rendered on the mobile device screen using thumbnail sizes. The proposed framework serves as a basis of content-based image retrieval on mobile devices. It addresses several important challenges such as hardware and software limitations as well as efficient use of the available network bandwidth.

Keywords Content-based · Retrieval · Mobile

1 Introduction

During the last two decades, visual content creation has been revolutionized from (analog) silver films to digital sensors that have changed the psychology of photography. Now every moment is important, users of multimedia devices can capture, edit, save and share lots of images. However, finding an image of interest from a heap of few gigabytes is an ongoing

I. Ahmad (✉)

Nokia Corporation, Hermiankatu 12, 33720 Tampere, PL 1000, 33721 Tampere, Finland
e-mail: iftikhar.ahmad@nokia.com

M. Gabbouj

Institute of Signal Processing, Tampere University of Technology, Tampere, Finland
e-mail: moncef.gabbouj@tut.fi

challenge for scientists and engineers. New multimedia devices (mobile phones, Personal Digital Assistants (PDAs), etc.) combined with web technologies provide the users with new forms of multimedia sharing such as story-telling and social networks. With different technologies such as Email, Multimedia Messaging Service (MMS) and Web; people share multimedia content with family members, friends and the general public. There are various web sites [2, 9, 16, 37, 42, 44] where users can share images and videos with other people. People can annotate their multimedia contents that can later be used for text-based search to find a specific media item. Parts of this metadata may be automatically generated during image capture, such as date, time, device information, location (Global Positioning System enabled devices) along with the user activity information (which can be found from e.g. device's calendar) would be useful. The use of annotated metadata helps in organizing multimedia items [33]. Flickr [9], Facebook [8], Photobucket [31] and YouTube [44] have shown that, manually assigning text to images and videos can be helpful. Although annotated metadata can play an important role for image retrieval in this paper we focus on content-based image retrieval from mobile devices.

Mobile phone industry has gone through a phenomenal change over the past few years with significant advances in the areas of communications and multimedia. 3G [45] services are already in the market and offer a great bandwidth to meet rising demands of users to deliver high quality multimedia services. Currently, state-of-the-art multimedia compliant mobile phones, equipped with digital cameras, have inherent support for network connection. This enables a user to access large amounts of digital media. The processing power and memory capacity of mobile phones are increasing all the time. The current generation rate of digital media by capture and store facilities in multimedia mobile phones calls for an ever-increasing demand of an efficient content management [27]. It requires a generic framework structure assuring a fast, reliable and low-cost retrieval of digital media items with a particular content from large archives [3].

The tag or filename of a multimedia item that is automatically assigned by the camera application on a mobile device is not so useful for media content identification and recognition. Moreover, annotation of digital media at the time of capture is cumbersome due to the limited device capabilities [10, 33, 36]. Systems such as IDEixis [39], is a content-based image retrieval system for location based services but users usually experience a large latency because of an inadequate mechanism for image transport, which is based on MMS. Kim et al. [13] proposed Visual-Content recommender for the mobile Web, where the main focus is image retrieval. Roth [32] proposed a content-based image indexing and retrieval by "mobile agent", which mainly addresses the security related issues. FAÇADE proposed by Kurz et al. [17] provides a context aware application for Web content adaptation on mobile devices. Sarvas et al. [34] have developed a client-server architecture for controlled and immediate online sharing of digital images over mobile platforms. Chandrasekaran and Joshi [5] developed a system, which provides access to the World Wide Web across wireless devices by a client-server architecture. Sorvari et al. [36] present a system for content management on mobile devices with the help of metadata. They mainly focused on usability issues such as usefulness, meaningfulness and privacy of the metadata context. Pham and Wong [30] discuss different features and requirements set in a mobile device for the purpose of realizing mobile multimedia applications. Boll and Westermann [4] present a peer-to-peer system for distributing multimedia items in an event, such as sports (Olympics, World Cup) to mobile devices. Due to the limited capabilities of mobile devices most systems discussed above are based on client-server architecture.

Among many research challenges towards which the multimedia mobile devices entail including media storage, presentation, adaptation to different platforms for exchange, etc.,

the area of multimedia retrieval plays an important role. Due to limited input capabilities and limitations imposed by only text-based queries, Content-Based Image Retrieval (CBIR) promises a potential solution to image retrieval from mobile devices. CBIR addresses the problem of accessing an image item that bears certain content and it usually relies on the characterization of the low-level features such as color, shape, texture, etc.

Content-based multimedia retrieval for personal computers is an active area of research for the last two decades. Systems such as “Multimedia Video Indexing and Retrieval System” (MUVIS) [24], VisualSEEK [35], Photobook [29], Virage [40] have common feature of having a framework for indexing and retrieving images and audio–video files. Content-based multimedia retrieval on mobile devices is a newly emerging area. Most of the studies available in the literature are focused on some specific domains or use-cases [17, 32]. Therefore, without proper content-based management methodologies, it is not feasible to find a media item-of-interest based on its content. Furthermore, due to the current limitation in storage capabilities of mobile devices, users have to store and manage their multimedia collection using a more convenient storage medium such as personal computers or peripheral hard-drives. Accessing such displaced multimedia items from a mobile device further contributes toward the challenge of multimedia management for mobile device users.

Recently, various low-level descriptors in CBIR such as color, texture and shape, have been developed for content-based retrieval operations in various image databases. Using such features, content based retrieval can perform relatively well for images [1]. Efficient retrieval of desired content (e.g. the images with certain content) also requires a dedicated query technique along with certain database organization and indexing structure [24].

With the help of cameras in mobile devices, users can capture an image and perform a content-based query operation [13], virtually from anywhere. However, CBIR in mobile devices presents new challenges [1], besides the content-based query operation. For instance different mobile devices are available in different designs and capabilities, different operating systems and input/output limitations. So, it is a challenging problem to provide a generic solution that suits all devices [3]. Nowadays the capabilities of mobile devices have been significantly improved (faster input/output, memory capacity, processing and battery power) but comparatively they are still far behind personal computers. With existing mobile operating systems such as Symbian OS [38], MS Windows Mobile edition [23], Linux [19] etc., it is possible to extract content-based features of a media item on mobile devices [11]. However, this approach presents several limitations. For examples, 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 a certain set of devices or certain platforms (operating systems). Another limitation is the battery power consumption and the lack of system resources such as processor power and memory capacity. Though a standalone CBIR system can be implemented on a mobile device [11], the ever-increasing number of multimedia items means that it would take an infeasible amount of time to perform content-based indexing and retrieval operations. Furthermore, such a system would eventually reduce the mobile device talk and stand by time.

The following research aims to bring CBIR beyond the standalone environment (such as personal computer) into the realm of mobile devices. In this paper, we present a CBIR framework for Mobile Devices (*CMD*). It is built upon *MUVIS* engine [24] and basically consists of two parts: Mobile MUVIS (M-MUVIS), which is used for Java enabled mobile devices and Java Server Pages. MUVIS (JspMUVIS) is used for non-Java mobile devices. Java [7] or Web browser on mobile devices helps to address some of the challenges

mention above. *M-MUVIS* has a client-server architecture. The *M-MUVIS* server is made of two Java servlets [18] running inside a Tomcat [6] web server which, in fact, performs the content-based media retrieval on the server side. The first one, *M-MUVIS* Query Servlet (*MQS*), performs efficient image query operations. The second servlet, *MUVIS* Media Retrieval Servlet (*MMRS*), is used for media format conversion and streaming. *JspMUVIS* creates a markup language [22] such as XHTML or HTML pages dynamically according to the look and feel of the device. On a mobile device (the client side) a Web browser displays the generated (XHTML) pages. The objective is to show that such a framework can conveniently be used for sharing or re-use of digital images, content management for networked photo albums, art galleries, etc.

The rest of the paper is organized as follows: Section 2 gives an overview of *CMD*. Section 3 presents several CBIR experiments over *CMD*. Finally in Section 4 we draw conclusions and provide ideas for future research work.

2 *CMD* architecture

As mentioned in the introduction, the proposed *CMD* is a client-server framework for image retrieval over wireless networks taking into account the processing power, memory (random access memory (RAM)) and screen size limitations of mobile devices (the clients). The *CMD* Framework is illustrated in Fig. 1, where the server side consists of four blocks: *M-MUVIS*, *JspMUVIS*, *Query Engine* and multimedia database. *M-MUVIS* and *JspMUVIS* use *Query Engine* on the server side to perform a query operation. The Query Engine underneath uses *MUVIS* for content-based media retrieval. The Event Manager in the Query Engine manages events between the Java side and the native side such as start the query, the results are ready etc. It also notifies the Query Result Manger when the results are ready.

When the *M-MUVIS*, *JspMUVIS* or PC client initiates a content-based query operation, the *CMD* server performs the query operation on the server side and generates the query results. The *CMD* server uses the generated results to create the Query Resultant Image (*QRI*), according to the screen size and resolution of the client (i.e. the mobile device or PC) where the first eleven similar images along with the query image are displayed as thumbnails. The similarity score (distance) of each retrieved image is shown on top of it. The client receives the *QRI* and presents it to the user. Database shown in Fig. 1 is a collection of indexed media items (e.g. images, video and audio clips) and features (extracted from media items) files. For efficient retrieval the images in *CMD* databases are indexed using a recently developed similarity-based indexing scheme called Hierarchical Cellular Tree (HCT) [15].

CMD supports Normal Query (*NQ*) and Interactive Query (*IQ*) operations [14]. *NQ* is a query-by-example (QBE) scheme, which is based on an exhaustive search over the entire database. *IQ* is developed in *MUVIS* and can provide intermediate retrieval results whenever they are requested by the user. In *CMD* a user can specify a particular time interval after which the server provides intermediate query results. Later the user can retrieve the final results when the query is completed on the server side. The use of *IQ* in *CMD* is described in Section 3.4.

The client-server framework raises and addresses privacy issues related to content since mobile devices usually contain personal or private contents. Mobile device users may want to share their images with family members and friends but not necessarily with others. To this end, the client-server framework can use authenticated and secure communication channel for the media transfer and the query operations.

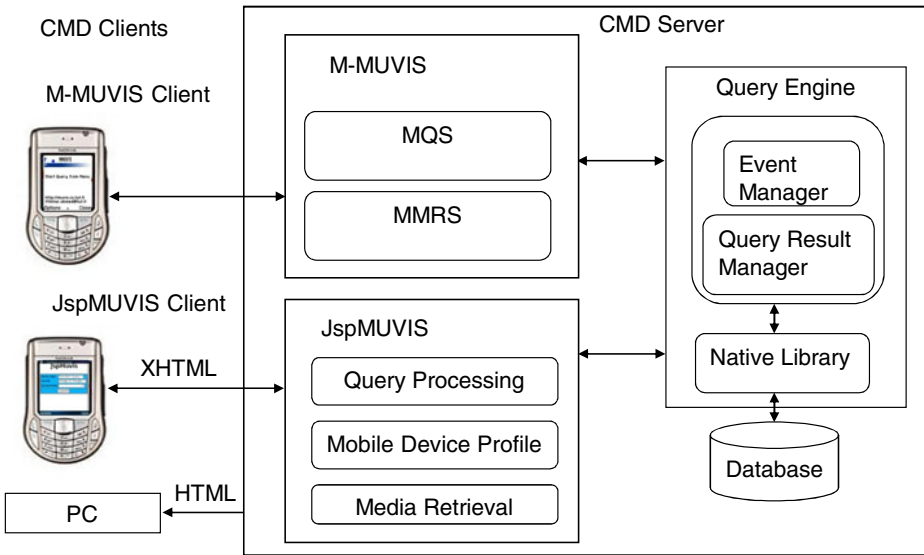


Fig. 1 CMD framework architecture

In the following sub-sections we shall discuss *M-MUVIS* and *JspMUVIS* architectures in Sections 2.1 and 2.2, respectively.

2.1 M-MUVIS architecture

As stated earlier, *M-MUVIS* has a client-server architecture, where the server is running on a PC and the client on a mobile device. The communication protocol between a client and the server has been designed to reduce network traffic for image retrieval across a wide range of mobile devices. An overview of *M-MUVIS* is shown in Fig. 2.

2.1.1 *M-MUVIS* client architecture

To take advantage of the flexibility and portability of Java, *M-MUVIS* client has been developed using Java 2 Platform, Micro Edition (J2ME) [12]. *M-MUVIS* clients can use the command types from a Mobile Information Device Profile (MIDP), to map the client commands to the device buttons, in such a way that it has the look and feel of the device applications [12].

The limited dimensions and weight of mobile devices impose certain constraints on their hardware and user interface capabilities. Additionally, mobile devices have different screen sizes, resolutions and limited input capabilities. These constraints challenge the effective access and presentation of the information on such devices. In order to address such limitations, the User Interface (*UI*) display capability of the *M-MUVIS* client can support different screens and image formats. When the client application is initialized, its *UI* is adapted according to the device screen size and therefore, a single *M-MUVIS* client application can be used on different devices with different screen sizes.

The client application consists of three parts: *Engine*, *UI* and *Utility*. The *Engine* is responsible for the activities going on behind the scene such as communicating with the server for a query operation, etc., and then pass *QRI* to the *UI* for display. To cope with the

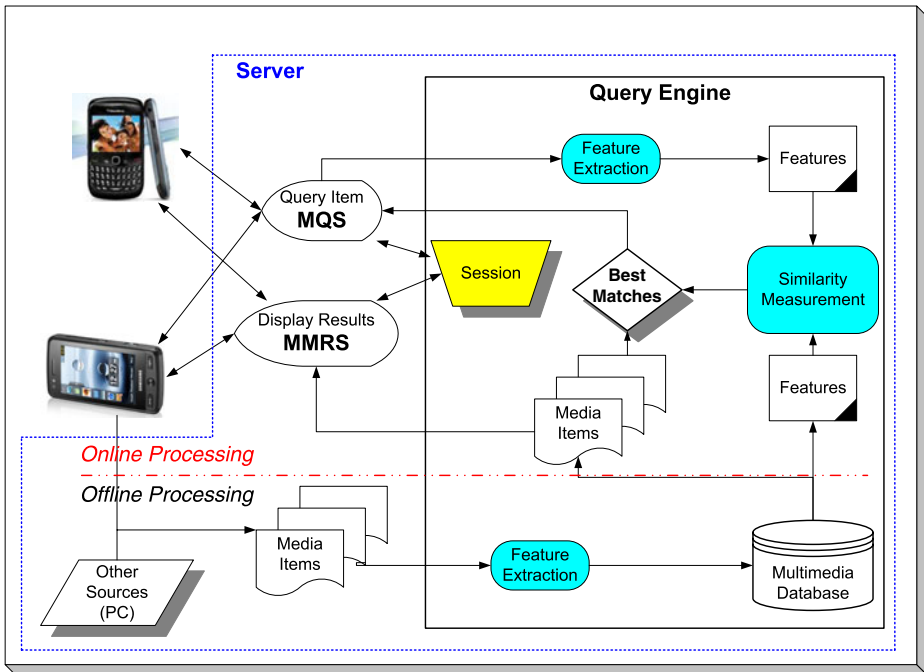


Fig. 2 *M-MUVIS* framework

difference of size and shape of mobile devices, the *UI* module adapts to the look and feel of the devices. It is also responsible for the client command placements, as different devices have different keys for the same command operation. The command placement enables the same look and feel to the user as with the native (device) applications. *M-MUVIS* client diagram is shown in Fig. 3.

Utility is used to parse and assemble the query string that is used as a communication protocol between the client and the server. Whereas mobile devices are for personal use of general public, the software of mobile devices are usually localized in different languages to make the device usable for every one. The utility part is responsible for the localization of *M-MUVIS* client application. The query result on Nokia 6630 [25] is shown in Fig. 4.

A user can change the settings of the *M-MUVIS* client such as query type, number of thumbnail in *QRI* etc. from the setting dialogs as shown in Fig. 5. *IQ* setting in the dialog is used to specify a time interval after which the user wants to see the query results. As mobile devices do not have high resolution displays and *QRI* is a JPG image, a low JPG quality value can be specified in the setting dialog for *QRI* creation to compress it on the server side. As a result, network traffic between the client and the server is reduced when streaming a smaller size *QRI* to an *M-MUVIS* client.

2.1.2 *M-MUVIS* server

With the help of 3G and Wireless local area network (WLAN), network access is improved significantly in the current mobile devices. However, compared to desktop computers, the user is not only waiting longer but might also be paying for the data exchange in mobile network. To overcome this problem, session management [43] on the server side is

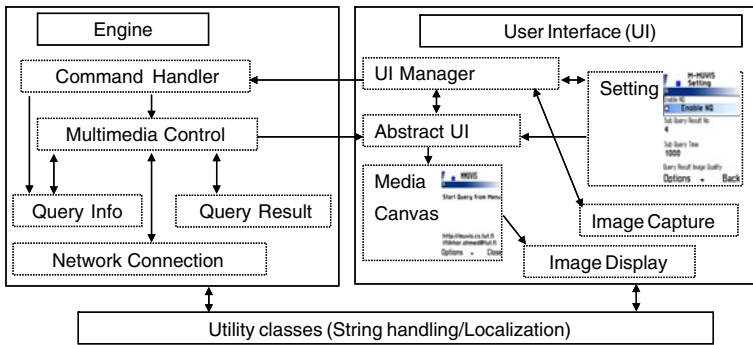


Fig. 3 M-MUVIS client diagram

proposed to reduce the network traffic and retrieve the *IQ* results. The session manages the *M-MUVIS* client information (status and setting). The status includes the query results while the setting includes the *IQ* time interval, the image thumbnail size, the device screen size and the *JPG* quality factor for the *QRI*. The session is created when the client connects to the server for the first query operation and then the server maintains the session during all transactions between the client and the server. If the user changes some client settings, the new settings are ultimately passed to the server and saved in the client's session.

Due to privacy and security reasons, the Uniform Resource Locators (URLs) of the retrieved image items are not passed to *M-MUVIS* client. The client only receives the thumbnails of the images. Later, the client can request the full-resolution images of some selected thumbnails and the server provides the images upon authentication. Furthermore, images may be watermarked, by re-encoding the image with additional metadata information in the *JPG* EXIF header, for later identification and copyright issues.

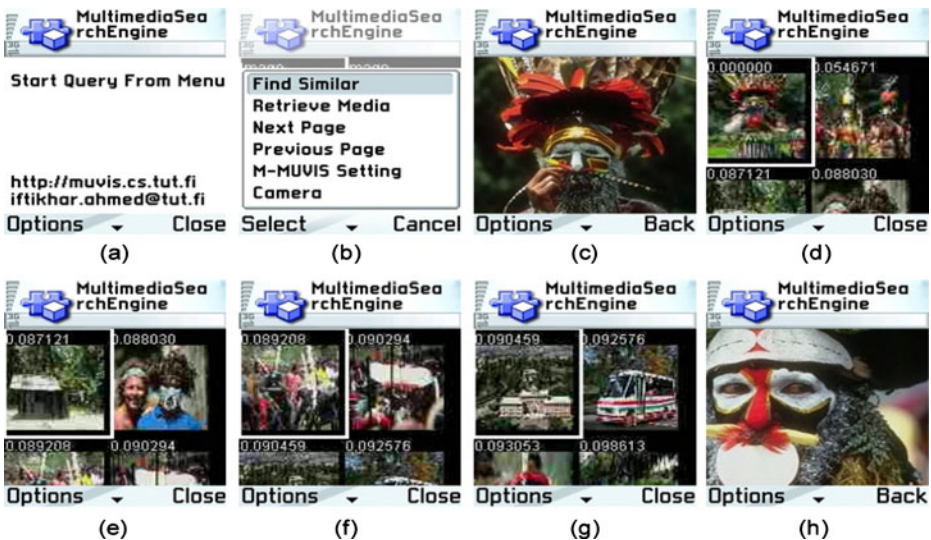


Fig. 4 a Main view of *M-MUVIS* client on Nokia 6630. b Initiating a query from menu c The query image d, e, f, g Query results on *QRI*. h An image selected within *QRI*

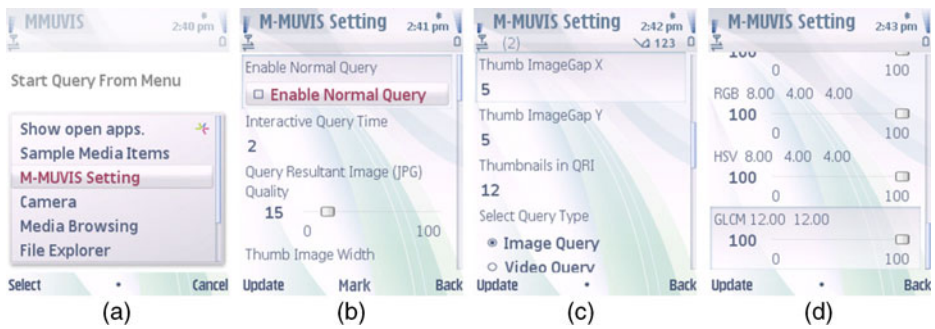


Fig. 5 M-MUVIS client setting view on Nokia 6630. **a** Setting menu. **b** Interactive Query setting and QRI, JPG quality factor. **c** Query type as image search. **d** Available feature in the database

MQS is responsible for performing the query operation on the server side. It uses the *query engine* to perform the query operation. When *MQS* receives the query request from an *M-MUVIS* client, it first parses the query string and then passes the query request to the *query engine*. The query operation uses a combination of low-level features (color/texture etc). With the help of *HCT*, a query path is created in the selected database where relevant (to the query) images are retrieved first and similarity distances are calculated. Ranking is performed after each comparison and the retrieval results are re-arranged accordingly. After *IQ* interval, the query result is saved to the client's session on the server side for the *QRI* retrieval and the query operation continues until all the images in the database are compared and ranked. The client is notified to retrieve the *QRI* from *MMRS*. After completion of the query operation, the final result is saved in the client's session.

For efficient media retrieval, a separate servlet is used due to the fact that CBIR can be time consuming in a larger database. *MMRS* receives the query result retrieval request from the client. It retrieves the query results from the client's session and requests the query engine to create the *QRI* for the mobile device according to the screen's size of the mobile device.

In order to specify the settings of a particular query operation described above, the *M-MUVIS* clients, *MQS* and *MMRS* use stream messages in an encoded string format. The encoded string between a client and the server uses HTTP [43] protocol underneath. HTTP is a stateless protocol, so a session [43] is created in *MQS* as soon as a query request is received from an *M-MUVIS* client as shown in Fig. 2. Session tracking [43] allows the *M-MUVIS* client to retrieve the query result from the session. Table 1 shows a sample string for a query operation. The query type defines the type such as "query by

Table 1 A sample content-based image query request in textual format between *M-MUVIS* client and server

<i>M-MUVIS</i> client query string	
"qt=601,in=6,sqn=4,sqt=500,qmt=490"	
Field	Description
qt	Query type
in	Image index in the client session
iqt	Interactive query interval
qmt	Query media type

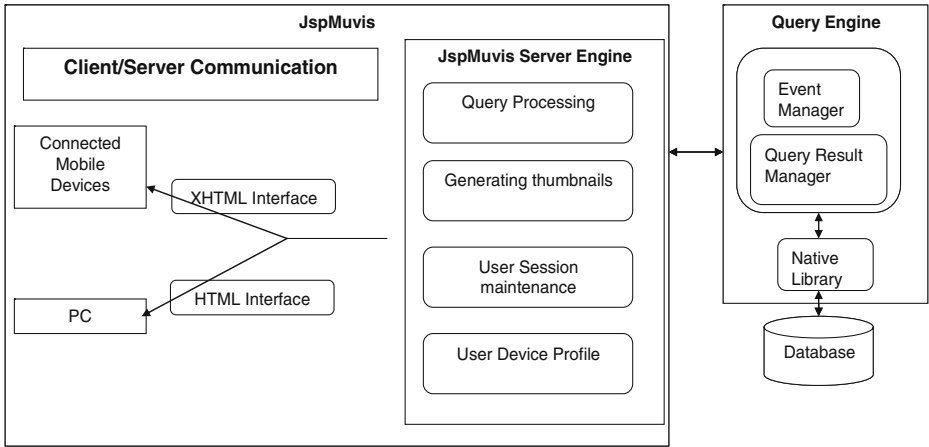


Fig. 6 *JspMUVIS* architecture

image name”, “query by image data” “image retrieval”, “randomly selected images from an image database”, etc. whereas query media type is either “visual query” in “images” or in “video”.

2.2 *JspMUVIS* architecture

Nowadays, access and use of a web browser is quite common in many mobile devices; however, few devices may not support interpreted languages such as Java. Therefore, *JspMUVIS* uses markup language [22], e.g. HTML, XHTML with a device browser for

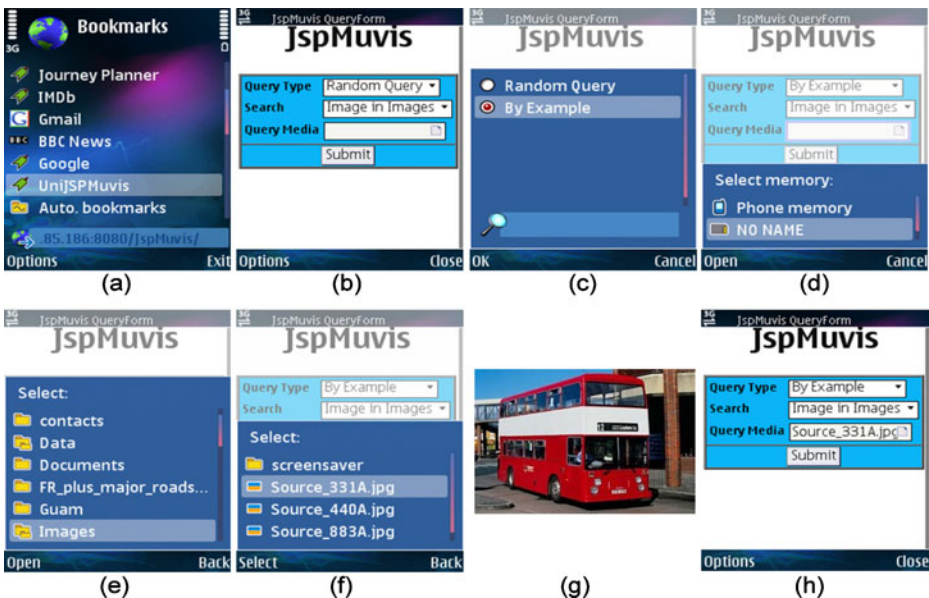


Fig. 7 *JspMUVIS* on Nokia N95



Fig. 8 *JspMUVIS* media retrieval on Nokia N95. **a, c** Retrieved *QRI* is presented. **b, e, g** *QRI* is selected by the user. **d, f, h** Original retrieved image item is shown

CBIR. The main layout of *JspMUVIS* is shown in Fig. 6. *JspMUVIS* has similar structure as *M-MUVIS*. There are separate Java beans [18] for the query operation and the media retrieval. As described in *M-MUVIS*, Section 2.1.2, a session is created and managed when a *JspMUVIS* client connects to the server. *JspMUVIS* uses *query engine* to perform the query operation and create *QRI* on the server side for the client.

In *JspMUVIS*, the server generates XHTML pages dynamically for mobile devices and HTML pages for PCs. Mobile devices can use a Web or WAP browser [41] to display those XHTML pages. *JspMUVIS* server (TomCat web server [6, 18]) also runs on a PC. Most of mobile device users are novice users. They are not computer programmers and their mobile devices are only for their personal usage; therefore, they are reluctant to install new applications on their devices. In *M-MUVIS*, the user has to install the client, a Java application, on the device to perform CBIR; whereas, in *JspMUVIS* there is no need to install any application on the devices. XHTML pages can be downloaded as in normal browsing and used for CBIR. Web or WAP browsers have either no or a limited access to device resources such as camera, Bluetooth etc., whereas the Java clients have easy access to different device resources (Bluetooth, camera, MMS, etc.). In *JspMUVIS*, a lot of extra data is transferred between the client and the server due to XHTML tags that increase the query and the media retrieval time. Although the data transfer between the client and the server is low in *M-MUVIS*, the perceived performance is better in *JspMUVIS* because Web browser can show partial information and keep on updating the page as more information is received.

JspMUVIS client running on Nokia N95 [25] is shown in Fig. 7. The web browser's main view is shown in Fig. 7(a), whereas *JspMUVIS* query form is shown in Fig. 7(b). In Fig. 7(c) to (f) the query images are selected from the file system on a mobile device. The Query Image is shown in Fig. 7(g) and in Fig. 7(h) the query is shown being submitted. Figure 8 shows the retrieved *QRI* and three selected images from the *QRI*. Four arrows on top of each thumbnail image in the *QRI* in Fig. 8 are for a content-based browsing of the

image database whereas, “similar” next to the arrows is to start CBIR query with the thumbnail image below it.

With the support of *JspMUVIS* and Java application, *CMD* is not only device independent but it is runtime (Java, Web etc.) independent CBIR framework for mobile devices.

2.3 Content based image browsing

Content based image browsing or a guided tour can be performed in an indexed database, where similar images are grouped together based on their features. HCT [15] is an efficient indexing scheme developed in *MUVIS* and used in *CMD*. In HCT similar media items are grouped together in cells based on their visual features and cells are further organized in a hierarchical cellular tree as levels. The user can perform the query operation and from the retrieved results, he can select one thumbnail image of his interest out of available thumbnail images on the device and can view all the other members of the same cell. The User can view next and previous cells or go up and down the tree. This provides a guided tour to the multimedia items where the user can view similar media items without performing any query operation.

3 Experimental results

A query-by-example is used to perform the query in a selected database where the features of the query image are first extracted then the query operation is performed. A set of experiments for image retrieval is carried out to evaluate the performance of the framework in different network technologies such as 3G and WLAN. Mobile phones used in the various experiments run Symbian OS. Nokia 5800 [25] mobile phone can use 3G networks in a query operation whereas Nokia N95 can use 3G and WLAN technologies. Three image databases are used in the experiments consisting of 10,000, 20,000 and 60,000 images in JPEG format. Basic visual features such as YUV, HSV, RGB color histograms, and Gray Level Co-Occurrence Matrix (GLCM) [24], [28] as a texture feature, are used on the server side to perform content-based query operations. The reader is referred to the cited references for details concerning the features used here.

A query image feature extraction time (T_q) is negligible when compared to a server query time (T_s) or a client server communication time (T_{cs}). In Eq. 1 the client query time (T_c) is directly proportional to T_{cs} , whereas, T_{cs} is directly proportional to the amount of data transferred between the client and the server. T_{cs} is decreased considerably by reducing the data exchange between the client and the server.

$$T_c = T_q + T_s + T_{cs} \quad (1)$$

For this purpose, session tracking is introduced on the server side (as described in Section 2.1.2) to reduce data transfer. As stated earlier, mobile devices have limitations on the upload and download channels [45]. There are fewer upload channels than download channels. In our framework, we have focused on reducing the data exchange (especially information uploading) with the help of session tracking. The client gets the intermediate query results and later can retrieve the complete query results.

Figure 9 shows *M-MUVIS* client running on Nokia 5800. In the Fig. 9(a) selected thumbnail image is the query image. Figure 9(b) and (c) shows the second and third row of the retrieved thumbnail images.

A query image along with content-based query results of *JspMUVIS* client on Nokia N95 is shown in Fig. 10.

3.1 Early access to most relevant images

The main goal of the HCT indexing structure is to partition a database in the feature domain into clusters (cell-based), where inter-image distances are minimized, so that in the selected features, similar images are grouped together. In an indexed database, a Query Path (*QP*) is formed over the clusters (a database subset) of the underlying indexing structure. The *QP* is a special sequence of database items where the most relevant items come first on the path. The advantage of *IQ* over indexed databases is that most relevant image items can be retrieved first; this reduces the query time on the server side. More details regarding *IQ* over HCT are provided in Section 3.4.

3.2 Query accessibility

A content-based image query, in a large scale image database from a mobile device, is different from querying it from a PC where the user has more interaction with the system (PC and software) and is watching the ongoing query operation. A mobile user has no idea how the query process is progressing on the server side and how long it will take to complete the query operation. Furthermore, wireless network delay, combined with long query time, may be frustrating for the mobile device user. So, the system must be responsive to the user. For this purpose, *IQ* is used to provide intermediate query results, so that the user can browse them and can retrieve the final query results. In this way, the user



Fig. 9 *M-MUVIS* client running on Nokia 5800 with an *IQ* interval of 130 ms

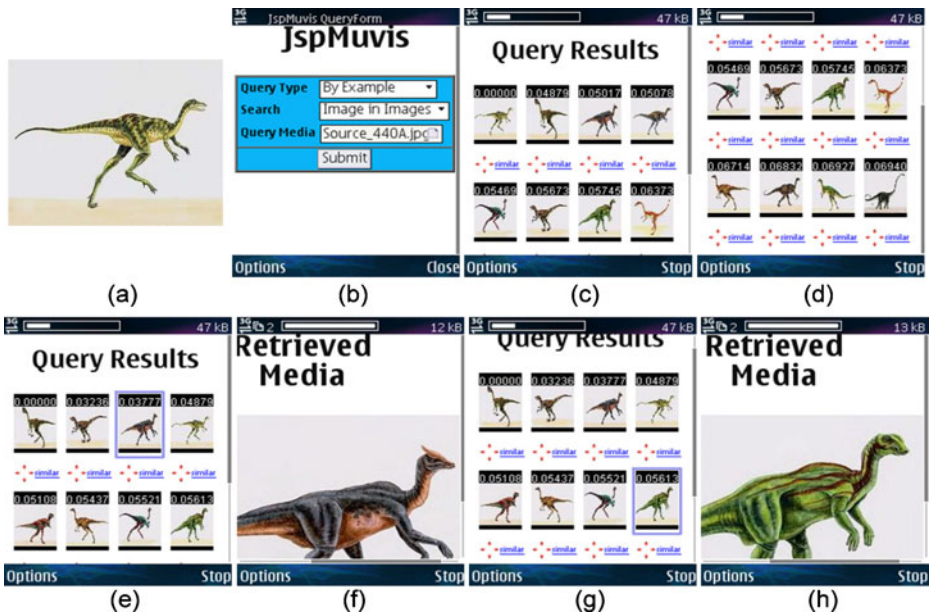


Fig. 10 a A query image is shown, b JspMUVIS main view is shown. c, d, e, g Retrieved *QRI* of image shown in (a) presented and f, h Images are shown from the query results

does not have to wait a long time for the query operation to be completed. The system will be responsive and interactive. If the user is satisfied with the intermediate results, he can retrieve the original media or start another query. The biggest advantage of *IQ* is that the user can adjust the query time and he can retrieve the results accordingly.

3.3 User specification for interactive query operation

The user can specify the *IQ* settings before initiating a content-based query operation. Since the mobile user does not know beforehand about the query completion time due to unknown size of the active database. Hence, the parameters of *IQ* have been selected accordingly. A fixed time (period = T_p) is selected for the first sub-query result. The *IQ* operation is designed to yield the result of the sub-query when the user requests a result update. When the query is completed, the final result will be sent back to the client. *IQ* provides interactive image query operation in a large scale image database so that the user can update the query results at any time and there is no limit on the numbers of intermediate query results. *IQ* in *M-MUVIS* is shown in Fig. 11. A similar approach is adopted in *JspMUVIS*.

3.4 Query retrieval statistics

Retrieval Rate (RR) [20] is used to evaluate the retrieval results where $N(q)$ is the size of the ground truth set for a query q in the database. $NF(\alpha, q)$ is the number of ground truth images found within the first α times $N(q)$ retrievals. $RR(q)$ is in the range of $[0, 1]$ where 0 means “no image found” and 1 means “all images are found”. If NQ queries are performed

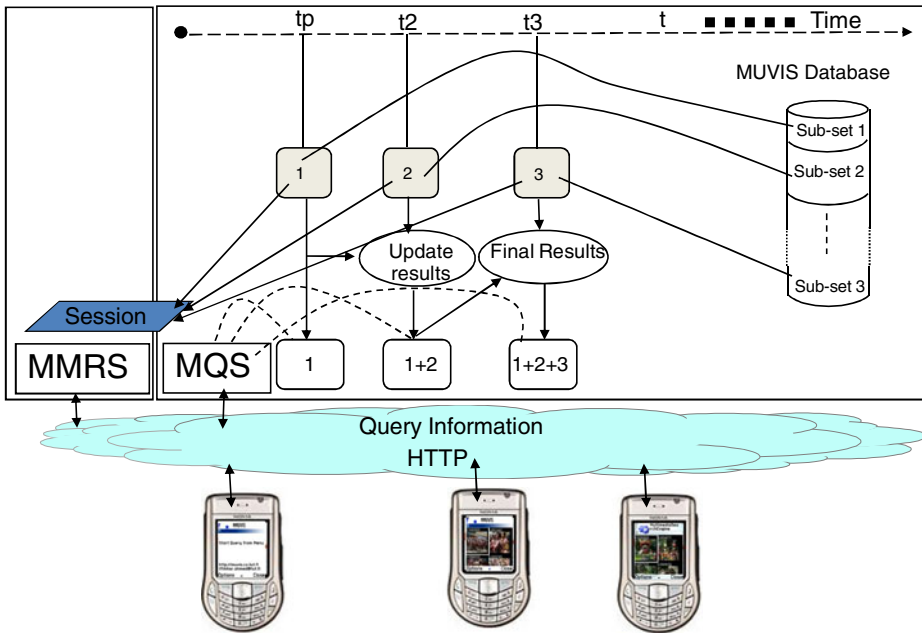


Fig. 11 IQ in M-MUVIS

then ARR is calculated according to the Eq. 2. Note that RR defined does not take the rank of retrieved images into consideration [20].

$$RR(q) = \frac{NF(\alpha, q)}{N(q)} \text{ and } \alpha \geq 1 \quad (2)$$

$$ARR = \frac{1}{NQ} \sum_{q=1}^{NQ} RR(q) \text{ and } 0 \leq ARR \leq 1$$

To overcome the limitation of RR, MPEG-7 Average Normalized Modified Retrieval Rank (ANMRR) [21] is used to measure the retrieval performance. It combines both the traditional hit-miss counters; Precision–Recall, and further takes the ranking information into account as given in the following expression:

$$AVR(q) = \frac{1}{N(q)} \sum_{k=1}^{N(q)} R(k) \text{ and } W = 2N(q)$$

$$NMRR(q) = \frac{2AVR(q) - N(q) - 1}{2W - N(q) + 1} \leq 1 \quad (3)$$

$$ANMRR = \frac{\sum_{q=1}^Q NMRR(q)}{Q} \leq 1$$

Here $N(q)$ is the minimum number of relevant (via ground-truth) images in a set of Q retrieval experiments, $R(k)$ is the rank of the k^{th} relevant retrieval within a window of W retrievals, which are taken into consideration for each query, q . If there are less than $N(q)$

Table 2 CBIR time statistics on different devices and networks in M-MUVIS

Network	<i>IQ</i>						<i>NQ</i>
	<i>CQT</i> (ms)		<i>SQT</i> (ms)		ANMRR	ARR	<i>CQT</i> (ms)
	μ	σ	μ	σ			μ
10,000 image database							
5800 (3G)	1954	48	663	108	0.5280	0.4128	8059
N95 (3G)	2864	1006	935	52	0.556	0.384	10919
N95 (WLAN)	771	28	568	54	0.5600	0.382	7891
20,000 image database							
5800 (3G)	2231	242	602	54	0.5462	0.4024	33740
N95 (3G)	2814	640	906	35	0.5444	0.4032	33851
N95 (WLAN)	860	89	444	105	0.5442	0.4032	32341
60,000 image database							
5800 (3G)	7942	541	4632	12	–	–	34101
N95 (3G)	8222	580	4932	11	–	–	35370
N95 (WLAN)	3750	60	3547	20	–	–	33269

relevant retrievals among W then a rank of $W+1$ is assigned for the remaining (missing) ones. $AVR(q)$ is the average rank obtained from query q . Since each query item is selected within the database, the first retrieval will always be the item queried and this obviously yields a biased Normalized Modified Retrieval Rank (NMRR) (q) calculation and is, therefore, excluded from ranking. Hence the first relevant retrieval ($R(1)$) is ranked by counting the number of irrelevant images a priori. Note that if all $N(q)$ retrievals are relevant, then $NMRR(q)=0$; and thus the best retrieval performance is achieved. On the other hand, if none of the relevant items can be retrieved among W then $NMRR(q)=1$, corresponding to the worst case. Therefore, the lower $NMRR(q)$ is the better (more relevant) retrieval for the query q . Keeping the number of QBE experiments sufficiently high, the average NMRR, ANMRR, as expressed in Eq. 3 can thus be used as the retrieval performance criterion. The ANMRR values shown in Table 2 are calculated based on 20 queries in the selected databases with the following settings: $N(q)=12$, $W=24$ and $Q=20$ for ANMRR whereas ARR is using $NQ=20$, $N(q)=12$ and $\alpha=2$. We use a smaller window of 12 images for retrieval performance due to the fact that mobile device user is not downloading many *QRIs* (each *QRI* contain 12 thumbnail images).

We present time statistics along with ANMRR of different *IQ* query operations in *M-MUVIS* over the sample databases. Basically, the mean (μ) and Standard Deviation (SD (σ)), are computed over 20 query operations performed in selected databases. The *CQT* is

Table 3 Time statistics of media (*QRI*) retrieval on different devices and networks

Network	<i>CMRT</i> (ms)		<i>SMRT</i> (ms)	
	μ	σ	μ	σ
5800 (3G)	3148	741	1922	624
N95 (3G)	5171	783	3672	500
N95 (WLAN)	1023	175	337	50

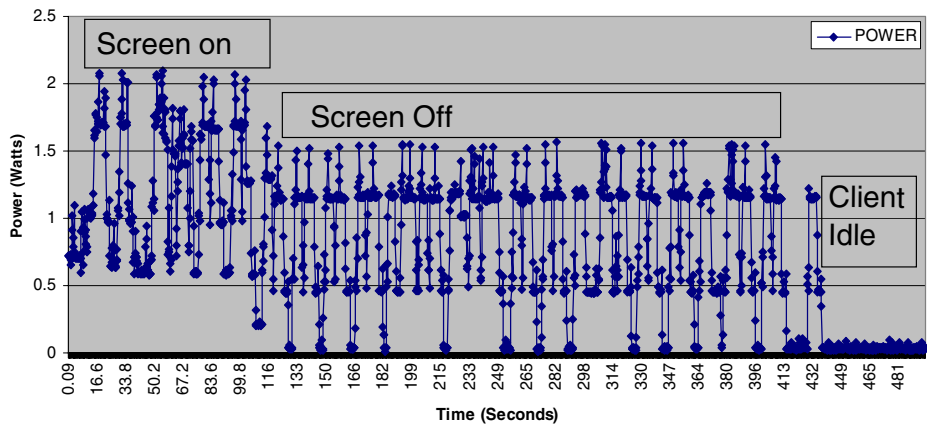
Table 4 CBIR time statistics on different devices and networks in *JspMUVIS*

Network	CQT (ms)		SQT (ms)	
	μ	σ	μ	σ
10,000 Image database				
5800 (3G)	6945	902	2004	51
N95 (3G)	7138	1248	2125	54
N95 (WLAN)	2579	603	1766	40

the waiting period for the query results, recorded by an *M-MUVIS* client. An experimental CQT in milliseconds is measured on Nokia N95 and Nokia 5800 as shown in Table 2. We measured the CQT by using different networks: the fastest query time is achieved in the Wireless Local Area Network (WLAN) as compare to 3G. However, we observed a higher variance in 3G as compared to WLAN networks due to the wider dynamic nature of wireless networks. With the help of IQ a user can adjust the CQT . As shown in Table 2, NQ takes a longer time in large databases, but with the help of IQ on *HCT*, CQT can be reduced and the most relevant (ANMRR <0.6) items in the predefined time (T_p) are obtained.

Client Media Retrieval Time ($CMRT$) is the time on the *M-MUVIS* client for the QRI retrieval. As shown in Table 3, the mean (μ) and SD (σ) are measured on Nokia 5800 and N95. WLAN has minimum network latencies, so the fastest media retrieval operations can be achieved over WLAN.

In a dynamic network environment where the available bandwidth changes between different mobile devices due to their connection type (3G, WLAN etc.) and Internet traffic, it is desirable for the framework to control the media quality in an adaptive way according to the dynamics of the underlying network. *MMRS* manages limited network resources efficiently by changing the image quality. Table 3 shows the Media Retrieval Time (MRT) on the *MMRS*. It is inversely proportional to the JPG quality of QRI . Lower quality value reduces the QRI data size which in fact reduces the network traffic. Consequently, QRI retrieval time is reduced. Reducing the network traffic actually makes this service feasible for practical usages. $SMRT$ is inversely proportional to the quality of the JPG; the lower quality, server obviously requires more time to compress the QRI . During the experiments, it was observed that JPG quality 20 to 25 is enough to display the QRI without any

**Fig. 12** *M-MUVIS* client power consumption for the 50 query operations on Nokia N95

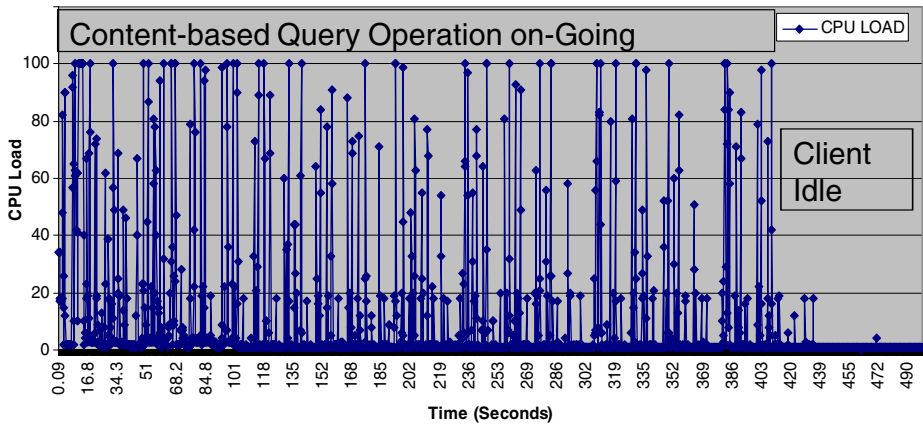


Fig. 13 M-MUVIS client CPU load for the 50 query operations on Nokia N95

noticeable degradation of user experience and that is also suitable for the *MMRT* servlet (less computation in the *QRI* creation). As current mobile devices do not support a high-resolution display, this high compression rate can be used conveniently.

Table 4 presents the time statistic of *JspMUVIS* on different devices and networks. *CQT* μ is higher in *JspMUVIS* as compared to *M-MUVIS*, due to the markup language tags in *JspMUVIS* for a small image database. However with the help of data repost methodology in a Web browser this difficulty can be overcome. As the network bandwidth is increasing in mobile devices. *JspMUVIS* can be widely used for multimedia retrieval.

As stated earlier, mobile devices are limited in processing power, battery and network data exchange, therefore the client application is designed to use them efficiently. As shown in Figs. 12, 13 and 14 the client application is working in a burst mode. When the client is idle (i.e. not performing a query operation), it is neither using CPU nor consuming network bandwidth or battery power.

Nokia Energy Profiler (NEP) [26] is used to measure the power consumption (Fig. 12), CPU load (Fig. 13) and network data exchange (Fig. 14). The device (N95) is draining more power when the display is on, as expected. Once the display is off it only drains power when the client is performing the query operation. Figure 12 shows that the device is

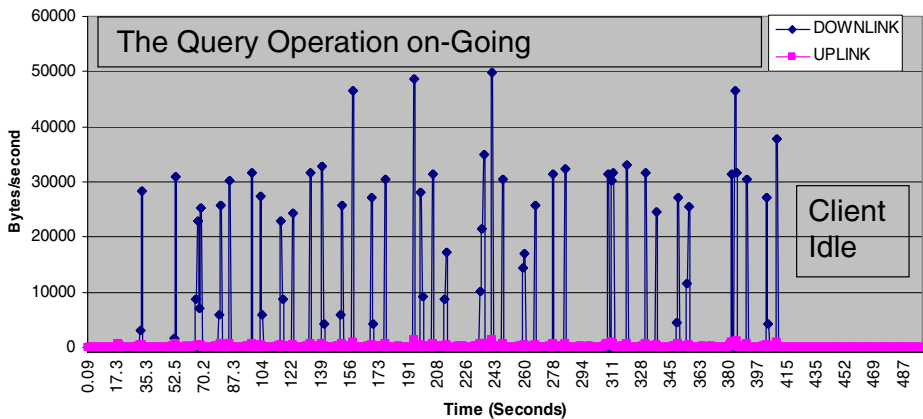


Fig. 14 M-MUVIS client data transfer for the 50 query operations on N95

not consuming power when the client is idle. Therefore, it can be left running in the background and used only when it is required.

As NEP is also a software running on the same hardware (Nokia N95) and Symbian is a multitasking operation system there are some small peaks in CPU load (in Fig. 13) due to other applications running in the background.

As stated in Section 2.1.2, session tracking is used to reduce the upload data. It is shown in Fig. 14 that data upload reduces considerably which helps to reduce the *CQT*. As shown in Figs. 12, 13 and 14 *M-MUVIS* client is using resources efficiently to perform the content-based query operation.

4 Conclusions

In this paper a generic image retrieval framework *CMD* for wireless network is presented, which integrates the *M-MUVIS* and the *JspMUVIS* systems to achieve efficient image retrieval. The client side adapts to the look and feel of the device, and with the help of session tracking, *IQ* over an indexed database provides efficient retrieval.

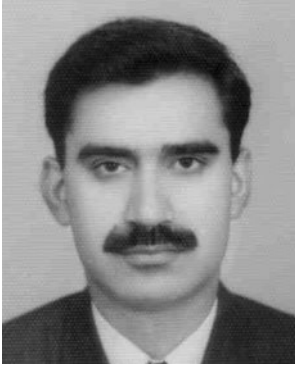
Since the existing information delivery structure of the World Wide Web is not suitable for content distribution and reusing the information for diverse range of platforms including mobile phones and PDAs, we proposed an improved framework. This has the capability to cope with information management across heterogeneous networks and a wide range of devices. The proposed framework has been tested with a number of image databases. We have obtained quite encouraging results for the content-based image retrieval application. However, we are facing limiting factors such as hardware/software features in mobile devices and unpredictable network responses. Such limitations and drawbacks can be overcome with 3G and WLAN which are available for high-end devices. The *CMD* holds a promising future in media content management and retrieval, as it can be tuned to different combinations and weights of low-level features to meet the requirements of vast majority of applications in the area of CBIR.

Currently, the *CMD* server is only a CBIR server and we are planning to make it a fully functional content-based multimedia indexing and retrieval server. We aim to make the *CMD* clients more users interactive, so that in the future, the user will be able to select the features for which image retrieval is intended.

References

1. Ahmad I, Kiranyaz S, Gabbouj M (2005) An efficient image retrieval scheme on java enabled mobile devices, MMSP 05, International Workshop on Multimedia Signal Processing, Shanghai, China, November
2. Alipr. <http://www.alipr.com/>
3. Arreympi J, Dastbaz M (2002) South Bank University, "Issues in delivering multimedia content to mobile devices", Sixth International Conference on Information Visualisation (IV'02), London, England, July, pp 622
4. Boll S, Westermann U (2003) Meeting experience: MediaEther: an event space for context-aware multimedia experiences. Proc ACM SIGMM 2003 Workshop on Experiential Telepresence, Berkeley, CA, November, 2003, pp 21–30
5. Chandrasekharan P, Joshi A (2002) MobileIQ: a framework for mobile information access. Proc Mobile Data Management Conference, Singapore, Jan
6. Chopra V, Bakore A, Eaves J, Galbraith B, Li S, Wiggers C (2004) Professional apache tomcat 5, published by Wrox, May, ISBN 0764559028
7. Deitel HM, Deitel PJ, Deitel HM, Deitel PJ (1999) Java how to program, 5th edition, published by Prentice Hall, December
8. Facebook. <http://www.facebook.com/>
9. Flickr. <http://www.flickr.com/>

10. Gandhi B, Martinez A, Bentley F (2004) Intelligent multimedia content management on mobile devices, Multimedia and Expo, 2004. ICME '04, 2004 IEEE International Conference, Taipei, Taiwan. June, 3:1703–1706
11. Guldogan O, Gabbouj M (2005) Content-based image indexing and Retrieval framework on Symbian based mobile platform, European Signal Processing Conference, EUSIPCO 2005, Antalya, Turkey, September
12. Keogh J (2003) The complete reference J2ME, published by McGrawHill OSBORNE Edition. Feb 27. ISBN: 0072227109
13. Kim CY, Lee JK, Cho YH, Kim DH (2004) VISCORS: a visual-content recommender for the mobile web. *IEEE Intell Syst* 19(6):32–39
14. Kiranyaz S, Gabbouj M (2006) Interactive query implementation over high precision progressive query scheme. In Proc of WIAMIS Workshop 2006, Korea, 19–21 April
15. Kiranyaz S, Gabbouj M (2007) Hierarchical cellular tree: an efficient indexing scheme for content-based retrieval on multimedia databases. *IEEE Trans Multimed* 9(1):102–119
16. Kodakgallery. <http://www.kodakgallery.com/>
17. Kurz B, Popescu I, Gallacher S (2004) FAÇADE—a framework for context-aware content adaptation and delivery. Communication Networks and Services Research Conference (CNSR 2004), Fredericton, N.B., Canada, May
18. Li S, Houle P, Wilcox M, Phillips R, Mohseni P, Zeiger S, Bergsten H, Ferris M, Ayers D (1999) Professional java server programming, published by Peer Information Inc., August, ISBN: 1861002777
19. Linux Devices. <http://www.linuxdevices.com/>
20. Manjunath BS, Ohm JR, Vasudevan VV, Yamada A (2001) Color and texture descriptors. *IEEE Trans Circ Syst Vid Technol* 11(6):703–715
21. Manjunath BS, Salembier P, Sikora T. Introduction to MPEG-7 Multimedia content description interface, published by John Wiley & Sons, LTD., ISBN 047148678 7
22. Markup. <http://www.w3.org/Markup/>
23. MS Windows mobile. <http://www.microsoft.com/windowsmobile/>
24. MUVIS. <http://muvis.cs.tut.fi/>
25. Nokia. <http://www.nokia.com/>
26. Nokia Energy Profiler. http://www.forum.nokia.com/Tools_Docs_and_Code/Tools/Plugins/Enablers/Nokia_Energy_Profiler/
27. OVI. <http://www.ovi.com/>
28. Partio M, Cramariuc B, Gabbouj M, Visa A (2002) Rock texture retrieval using gray level co-occurrence matrix. In Proc of 5th Nordic Signal Processing Symposium, Trondheim, Norway, October
29. Pentland A, Picard RW, Sclaroff S (1994) Photobook: tools for content based manipulation of image databases. Proc SPIE (Storage and Retrieval for Image and Video Databases II), April
30. Pham B, Wong O (2004) Computer human interface: “Handheld devices for applications using dynamic multimedia data” Proc. 2nd International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia, Singapore, June, 2004, pp 123–130
31. Photobucket. <http://photobucket.com/>
32. Roth V (1999) Panel: content-based image indexing and retrieval with mobile agents. ASAMA '99 Proceedings of the First International Symposium on Agent Systems and Applications and Third International Symposium on Mobile Agents. Palm Springs, California USA October, pp 260
33. Sarvas R, HerrarteE, Wilhelm A, Davis M (2004) Metadata creation system for mobile images. Proc of the 2nd international conference on Mobile systems, applications, and services, MobiSys, Boston USA, June, pp 36–48
34. Sarvas R, Viikari M, Pesonen J, Nevanlinna H (2004) MobShare: controlled and immediate sharing of mobile images. In Proc of Multimedia 2004, ACM, New York, NY, USA, October, pp 724–731
35. Smith JR, Chang SF (1996) VisualSEEK: a fully automated content-based image query system. ACM Multimedia, Boston, November
36. Sorvari A, Jalkanen J, Jokela R, Black A, Koli K, Moberg M, Keinonen T (2004) Usability issues in utilizing context metadata in content management of mobile devices. NORDICHI, Tampere, Finland, October 2004
37. Stumbleupon. <http://www.stumbleupon.com/>
38. Symbian OS. <http://www.symbian.com/>
39. Tollmar K, Yeh T, Darrell T (2004) IDEixis: image-based Deixis for finding location-based information. Mobile HCI, Vienna, pp 781–782
40. Virage. <http://www.virage.com/>
41. WAP. <http://www.wapforum.org/>
42. WebShot. <http://www.webshots.com/>
43. Wong C (2000) HTTP pocket reference, (Pocket Reference (O'Reilly)) published by O'Reilly. June. ISBN: 1565928628
44. YouTube. <http://www.YouTube.com/>
45. 3gpp. <http://www.3gpp.org/>



Iftikhar Ahmad has completed his Licentiate in 2003 on “A Framework of Content-based Retrieval in Mobile Devices”. He is currently working as senior engineer, in Java, Application Platforms, Technology Platforms, Nokia Corporation. His research interests include multimedia content-based retrieval for mobile devices.



Moncef Gabbouj (M’85–SM’95) received his BS degree in electrical engineering in 1985 from Oklahoma State University, Stillwater, and his MS and PhD degrees in electrical engineering from Purdue University, West Lafayette, Indiana, in 1986 and 1989, respectively. Dr. Gabbouj is a Professor at the Department of Signal Processing at Tampere University of Technology, Tampere, Finland. He was Head of the Department during 2002–2007. Dr. Gabbouj was a visiting professor at the American University of Sharjah, UAE, in 2007–2008 and Senior Research Fellow of the Academy of Finland in 1997–1998 and 2007–2008. His research interests include multimedia content-based analysis, indexing and retrieval; nonlinear signal and image processing and analysis; and video processing and coding.

Dr. Gabbouj served as Distinguished Lecturer for the IEEE Circuits and Systems Society in 2004–2005. He served as associate editor of the *IEEE Transactions on Image Processing*, and was guest editor of Multimedia Tools and Applications, the European journal *Applied Signal Processing*. He is the past chairman of the IEEE Finland Section, the IEEE CAS Society, Technical Committee on DSP, and the IEEE SP/CAS Finland Chapter.

Dr. Gabbouj was the recipient of the 2005 Nokia Foundation Recognition Award and co-recipient of the Myril B. Reed Best Paper Award from the 32nd Midwest Symposium on Circuits and Systems and co-recipient of the NORSIG 94 Best Paper Award from the 1994 Nordic Signal Processing Symposium. He is a member of IEEE SP and CAS societies.