

PROGRESSIVE QUERY TECHNIQUE FOR IMAGE RETRIEVAL ON MOBILE DEVICES

Iftikhar Ahmad¹, Shafaq Abdullah², M. Serkan Kiranyaz³, Moncef Gabbouj³

¹Nokia Corporation, P.O.Box 88, FIN 33721, Tampere. Email: iftikhar.ahmad@nokia.com, Phone: +358 503480631, Fax +358 718047577

²Vega Technologies Oy, Pinninkatu 5, FIN 33100, Tampere. Email: shafaq.abdullah@vega.fi

³Signal Processing Laboratory, Tampere University of Technology, P.O.Box 553, FIN-33101 Tampere, Finland. Email: shafaq.abdullah@tut.fi, serkan@cs.tut.fi, moncef.gabbouj@tut.fi

ABSTRACT

Recent hype in multimedia and communication technologies has enabled hand-held devices to capture, store and retrieve digital media items. Content-based image retrieval on hand-held devices, offers a great challenge of reducing query retrieval time. The nature of mobile environments demands an expeditious retrieval of the most relevant results from large-scale image databases. In this paper, we present the implementation of an advanced retrieval scheme, the *Progressive Query (PQ)* on a mobile platform. *PQ* is designed to bring an effective solution especially for queries on large-scale multimedia databases and furthermore to provide periodic query retrievals along with the ongoing query process. The experimental results approve that the retrieval time among the mobile terminals can be reduced and the retrieval performance can be improved without the necessity of having a superior processing power or system performance.

1. INTRODUCTION

Mobile phone industry has witnessed a phenomenal change in the recent times. The tremendous advancement in multimedia and communications technology has led to an availability of mobile phones equipped with camera, cam-coder, possessing faster processing power and larger memory, with the support of high data rate networks such as GPRS, EDGE and 3G [6]. These factors have been contributing in creating a plethora of personal media content. The need of content organization and retrieval of desired media has been felt by the people using hand-held devices. The current state-of-the-art hand-held devices such as mobile phones, PDAs, pocket PCs support Java platform [11] in one of its profiles based on configuration such as MIDP [13]. The expeditious retrieval of multimedia content requires indexing of multimedia primitives along with extraction of low-level features. Systems such as MUVIS [7], Visual Seek [9], Photo-

book [7], VIRAGE [15] have feature of having a framework for indexing and retrieving images and audio-video files. The contemporary MUVIS has been developed as a system for Content-Based Multimedia Retrieval on a PC-based environment.

We have developed a system, "Mobile MUVIS", based on Multimedia Video Indexing and Retrieval System (MUVIS). The Mobile MUVIS (M-MUVIS) aims to provide a content-based image retrieval mechanism on any device supporting Java platform. The M-MUVIS system is based on client/server paradigm. The server called "M-MUVIS Server" is a Content Based Image Retrieval (CBIR) server.

There are already few methods for performing query from the mobile devices. In an earlier MUVIS implementation for Nokia 9210 [1] and Nokia 3650 [2] a single feature has been used for query. In that system, the query operation is performed within the entire database and the result is generated at the end of the query. So the client (the user with the mobile device) has to wait for a longer time to get the query results and this essentially caused longer and sometimes infeasible retrieval times for the mobile user point of view.

In order to achieve efficient retrievals especially for large image databases, it is therefore, vital to devise a method which not only reduces the query processing time but also performs the query operation without requiring a system equipped with high performance hardware such as fast processors, large memory and high speed I/O access. Our research is based on a novel technique called *Progressive Query (PQ)* [5] to (content-based) search and retrieve images from large-scale databases. *PQ* is primarily developed to provide periodic and faster retrievals along with the ongoing query process and it can be especially useful for databases lacking an efficient indexing structure. *PQ* achieves this by producing intermediate query results and delivering it to M-MUVIS client (phone user). *Progressive Query (PQ)* supported in the M-MUVIS differ from Normal Query (*NQ*) operation, in which the query results were based on calculating the

similarity distances using *Feature Extraction Modules (FeX)* [4] of all the images present in a database and performing ranking operation afterwards [3]. The *PQ* technique on the other hand, is able to provide intermediate results along with the ongoing query operation by dividing the entire image database into smaller sub-sets.

M-MUVIS client is sending the query request to the M-MUVIS server. M-MUVIS server initiates the query on the server side. *PQ* operation generates the intermediate results. M-MUVIS server selects the *PQ* user defined result index and sends the result to the M-MUVIS client.

The rest of the paper is organized as follows: Section 2 presents an overview of *PQ*. Section 3 describes the client-server architecture. Section 4 reports experimental results on *PQ* implementation on M-MUVIS framework and finally we draw conclusions from our work in Section 5.

2. PROGRESSIVE QUERY: AN OVERVIEW

PQ is composed of periodic series of *Progressive Sub-Queries PSQ*. A sub-query is based on the specific time period, which can be set by user preferences before initiating the query operation. A sub-query is a partial query operation performed over a sub-set of database items. The database items being used in the sub-query can be selected either randomly or sequentially.

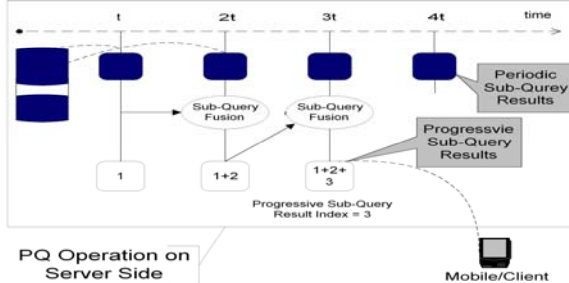


Figure 1: Progressive Query on Mobile Devices

Since *PQ* provides a series of intermediate results, each of which obtained from a (smaller) sub-set within the database, the chance of retrieving relevant database items that would not be retrieved otherwise via *NQ*, might be increased. It is from the fact due to the limited discrimination of the visual features used in image databases, a searching operation yield better result when the search space is smaller. Otherwise the probability of more and more irrelevant items retrieval in the higher ranks than the relevant ones increases. Since *PQ* partition the search space into several sub-sets, it uses this as an advantage and “might” retrieve “better” results in its earlier *PSQ* steps than the final (end) step as the *NQ* operations produce.

2.1. Sub-Query Fusion Operation

PQ basically generates intermediate query results that are then fused together in a sorted way. The final retrieval result of *PQ* converges to what Normal Query (*NQ*) operation produces. This is achieved by fusion operation applied on the intermediate *PSQ* retrievals. Let n_1 and n_2 be the number of items in the first and second sub-set, respectively. Since there are items to be inserted into the fused list one at a time, the fusion operation can take maximum comparisons $\text{Max}(n_1, n_2)$. In worst case whenever the items from both lists are evenly distributed with respect to each other. On the other hand if the maximum valued item (i.e. the last item) in the smaller list is less than the minimum valued item (i.e. the first item) in the bigger list, the number of comparisons will not exceed the number of items in the smaller list because once all of the items in it are compared with the (smallest) first item in the bigger list and henceforth inserted into the fused list, there will not be any more comparisons needed. Note that this is the direct consequence of the fact that the both lists are sorted (from minimum to maximum) beforehand and one of them is now fully depleted. Therefore, the fusion operation will take minimum comparisons $\text{Min}(n_1, n_2)$ respectively [5].

2.2. User Specifications for a PQ Operation

The *PQ* operation is designed so that it yields the *PSQ* after a fixed time period ($t=T_p$). So there are two parameters, which the user can specify before initiating a *PQ* operation: *PQ* period (T_p), which is used for the update of *PSQ* retrieval results. The other parameter is the *PSQ* result index (PSQ_i) indicating which particular *PSQ* retrieval should be sent to the M-MUVIS client side (the mobile user). The basic *PQ* operation and the user settings are illustrated in Figure 1, in which the user specifies $t = T_p$ (msec) as *PQ* time period and $PSQ_i = 3$ as the *PSQ* Result Index. Accordingly, the third *PSQ* retrieval result generated is fetched back to M-MUVIS client and displayed on the phone.

3. CLIENT-SERVER ARCHITECTURE

M-MUVIS system is based on a client-server paradigm [3]. The M-MUVIS client application is running on a hand-held device (Nokia 6630) with an inbuilt camera and a conventional support for network connection such as GPRS. M-MUVIS client application development is carried out using Java (Mobile Information Device Profile) MIDP to take advantage of portability and standard programming API. The M-MUVIS server consists of a Java Servlet [10] running inside a web server (i.e. Tomcat [12]) that in effect transforms the contemporary MUVIS into a web-based client-server application. It uses native libraries for efficient

computation of query operations through native calls from Java objects.

3.1. M-MUVIS Client

M-MUVIS client application is written in Java MIDP. The basic query operation is performed as follows: M-MUVIS client first forms the query request and then sends it to the server. The server performs the query operation accordingly and sends the (required) *PSQ* retrieval to M-MUVIS client. M-MUVIS client retrieves the query results and displays the images accordingly. Using the M-MUVIS client application on his/her phone, the phone user can set the query parameters and retrieve the query results as shown in Figure 2.

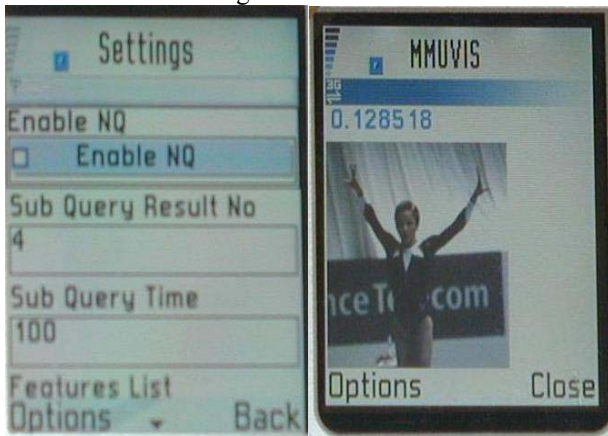


Figure 2: M-MUVIS Client Application

M-MUVIS client application consists of three components/packages, namely, User Interface (UI), Engine and Utility components.

UI components are responsible for the query result display. It presents the query options to the user. User can start the query accordingly. Since the mobile devices usually have a smaller screen size capable of showing one image at a time, the user can only view the retrieved images one by one, using a menu option in the UI.

The M-MUVIS engine part is responsible for making a query, sending the query to the M-MUVIS server and retrieving the query result. It is also responsible for the synchronization between M-MUVIS client and server.

Third part consists of some utility tools they are used in both UI and engine part.

3.1.1. Design Requirements of M-MUVIS Client

Mobile devices are resource limited devices in terms of input and output. Most of them are not equipped with a rich keyboard or pointer. Yet nowadays most of them have a camera, hence the user can take picture and send it to the M-MUVIS server for a query operation. M-MUVIS client application is basically designed considering such facts and physical limitations.

3.2. M-MUVIS Server

We are using Java servlet technology on the M-MUVIS server side. M-MUVIS server is making native calls for querying the database. Memory requirement is proportional to the database size and the amount of features used for a *NQ* operation. Due to the partition of the database into sub-sets, *PQ* operation reduced memory requirement as shown in Figure 3 [5]

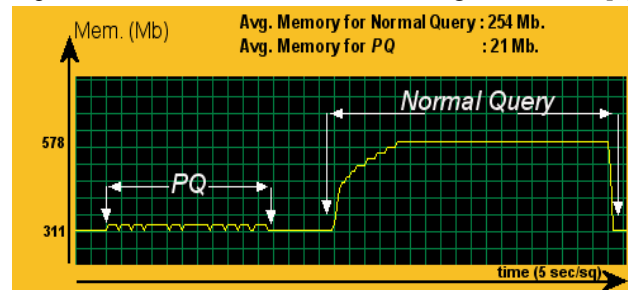


Figure 3: Memory Usage, PQ vs. NQ

3.3. Communication Protocol

A communication protocol is defined between M-MUVIS client and server. It is using Hypertext Transfer Protocol (HTTP) [8] as underlying protocol. This protocol specifies the media type (Image media), query type (random query from database, query by image data or query with an image from database), device platform, PQ time and PQ result index as shown in Figure 4

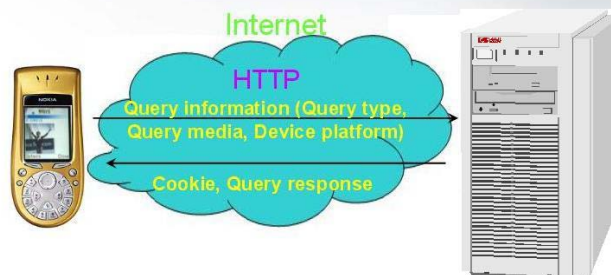


Figure 4: Communication Protocol between M-MUVIS Client and Server

M-MUVIS client is sending the query in a formatted string to M-MUVIS server. The M-MUVIS server is parsing the query string and extracts the query information. M-MUVIS server sends the query results to the M-MUVIS client in another formatted string, which is then parsed by the M-MUVIS client for retrieving the query results.

A session is created in the M-MUVIS server once the query request is received from M-MUVIS client (Figure 4). A cookie is sent back to M-MUVIS client, which is used for session tracking in M-MUVIS client. The specified *PSQ* result is fetched back to M-MUVIS client with M-MUVIS server storing all intermediate results at *PSQ* instance specified earlier in the query request. Session tracking allows the M-MUVIS client to retrieve an intermediate *PSQ* result.

4. EXPERIMENTAL RESULTS

The M-MUVIS client application has a query settings dialog as shown in Figure 2. M-MUVIS client can use that dialog to set the *PQ* period in milliseconds and *PSQ Result Index*. In Figure 5 we show the relationship between total *PQ* time and *PSQ Result Index*. We have obtained results by keeping *PQ* period constant over one curve. We increment the *PSQ Result Index* and record query processing time (Total *PQ* Time (millisecond)) on M-MUVIS server.

As shown in the graph of Figure 5, total progressive query time increases as M-MUVIS client selects higher *PSQ Result Index*. *PQ* period is used to change the size of the sub-sets that are formed inside the database.

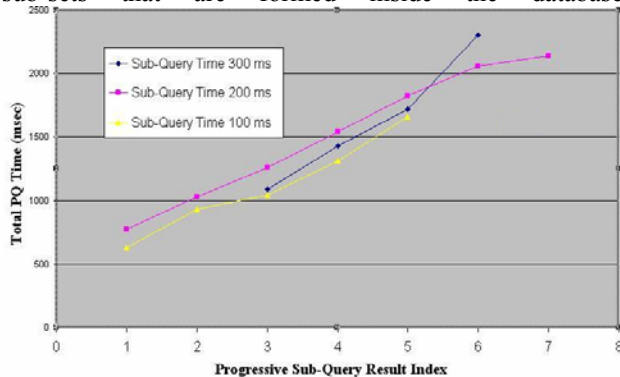


Figure 5: Progressive Query on Image Database

In Figure 6, *PQ* retrieval time versus number of *PSQ* results is plotted. It is based on experiment conducted on M-MUVIS server side by varying *PQ* period and recording the number of *PSQ Retrieval Results* generated. The database used contained around 2000 images scaled down to match resolution supported by mobile devices. Consequently the number of *PSQ* retrievals is inversely proportional to the *PQ* period.

The advantage of this scheme is that M-MUVIS client can get intermediate results from an ongoing query process and if the user is satisfied with the query results then the query process can be immediately stopped. Otherwise the user can either set a higher *PQ* period to increase the sub-set size or a higher *PSQ* result index to get the retrieval results after more sub-sets are covered by the (ongoing) *PQ* operation. In both cases more database items are searched.

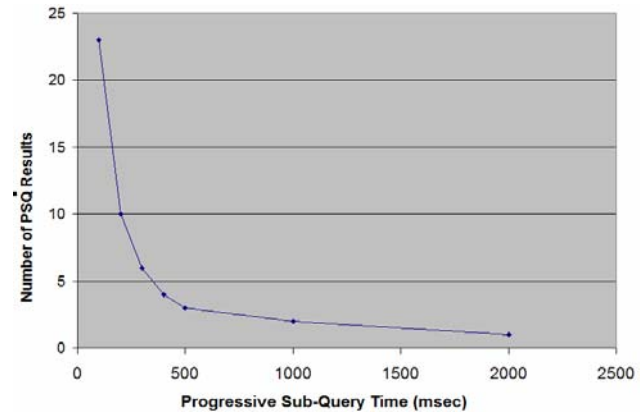


Figure 6: Total PQ Time vs. Progressive Sub-Query Result Index for the fixed Sub-Query Time

5. CONCLUSION AND FUTURE WORK

With the advantage of new mobile devices with new technologies such as GPRS, EDGE, 3G and faster processor in the device with color display user can make the query from the hand held devices. As more and more Java enabled devices are out in the market it is possible to make *PQ* as a service for the public or private use. A user can run the server on his machine at home and can query in his personal collection from anywhere.

PQ is developed to provide faster retrievals along with the ongoing query process. In this way we can avoid the longer query times for the mobile users. More importantly the user can initiate a query operation from his/her mobile device for an image database and he/she can access the earlier (intermediate) results whenever he/she wants to. Retrieval time can be improved with the help of *PQ* on indexing scheme; M-MUVIS client will get the most relevant results in earliest possible instant.

6. 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, F. Alaya Cheikh, B. Cramariuc and M. Gabbouj, "Query by image content using Mobile Information Device profile (MIDP)", Proc. of the 2003 Finnish Signal Processing Symposium, Finsig'03, Tampere, Finland, May 2003.
- [3] I. Ahmad, S. Abdullah, S.Kiranyaz, M.Gabbouj, "Content-Based Image Retrieval on Mobile Devices", Proc. SPIE (Multimedia on Mobile Devices) 5684, Electronic Imaging Symposium 2005, San Jose, California (USA), 16-20 Jan. 2005.
- [4] S. Kiranyaz, K. Caglar, E. Guldogan, O. Guldogan, and M. Gabbouj. "MUVIS: a content-based multimedia indexing and retrieval framework", Proceedings of the Seventh International Symposium on Signal Processing and its Applications, ISSPA 2003, Paris, France, pp. 1-8, July 2003.
- [5] S. Kiranyaz, M. Gabbouj, "A Novel Multimedia Retrieval Technique: Progressive Query (WHY WAIT?)", WIAMIS Workshop, Lisboa, Portugal, April 2004.
- [6] J. Lempiäinen, M. Manninen, Radio Interface System Planning for GSM/GPRS/UMTS, Kluwer Academic Publishers, 2001.
- [7] 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.
- [8] HTTP Pocket Reference (Pocket Reference (O'Reilly)) by Clinton Wong
- [9] J.R. Smith and Chang "VisualSEEK: A fully automated content-based image query system", ACM Multimedia, Boston, Nov. 1996.
- [10] Java Servlet Programming (O'Reilly) by Jason Hunter with Willaim Crawford
- [11] Java How to Program, 5th Edition by H. M. Deitel, P. J. Deitel, Harvey M. Deitel, Paul J. Deitel
- [12] Professional Apache Tomcat 5 by Vivek Chopra, Amit Bakore, Jon Eaves, Ben Galbraith, Sing Li, Chanoch Wiggers
- [13] The Complete Reference J2ME by James Keogh, McGrawHill OSBORNE Edition.
- [14] "MUVIS", <http://muvis.cs.tut.fi>
- [15] "Virage," <http://www.virage.com>