An Application for Image Database Development for Mobile Face Detection and Recognition Systems

R. S. Ivanov[1]
[1]Technical University of Gabrovo, Bulgaria, rs-soft@ieee.org

Abstract -The development of the modern mobile communication systems and the accessibility of the mobile terminals with built-in multimedia possibilities enable the creation of various mobile applications, able to process multimedia information (audio and video) in soft real time. A typical example of such applications are the systems for face detection and recognition. One of the major problems in these systems is the absence of sufficient volume of training and test material – frames from the cameras of the mobile terminals.

This paper presents a client-server application which in on-line mode enables the development of image database, which can be used in training and testing of systems for mobile face detection and recognition. This is a platform independent application, and may be used by any mobile terminal, which supports Java and program access to the camera**.**

I. INTRODUCTION

Within the last several years, a trend is being observed for significant improvement of the functional characteristics of the mobile terminals with multimedia capabilities [9]. The combination of microprocessors with low consumption and clock frequency over 300MHz, operative memory over 1MB, NAND Flash over 100MB, cameras with resolution over 2 Mpixels, and 3G technologies allows for the development of Java and C#/C++ mobile applications, which few years ago would be unthinkable. The modern multimedia mobile terminals enable the development of applications for face detection and recognition which use relatively complicated algorithms. The creation of such application would be impossible without the existence of Application Program Interfaces (APIs), enabling the program access to the camera, microphone, flash-memory, and different network interfaces. The usage of HTTP protocol with (E)GPRS or UMTS bearers allows for the download and upload of multimedia information within a period that is close to real time.

Most of the mobile face detection and recognition algorithms use training and test data sets of images, obtained from professional cameras [1-3]. Furthermore, experimental results are reported using different image databases. In order to compare algorithms fairly, training and test sets with a large number of images, obtained from mobile terminals are required.

The training and testing of systems for face detection and recognition, adapted for mobile terminals, requires the availability of image databases, containing sufficient and balanced material. The not so good parameters of the mobile terminal cameras imposes that the testing be realized through snapshots, obtained from mobile devices. The main requirements to such image databases are as follows:

- The number of the snapshots to be sufficient for the reliable training and testing of the algorithms developed;
- A balanced number of different types of snapshots: number of faces in the frame, type of lightning (natural, artificial), drawbacks (presence of objects which cover part of the face, or with color that is close to human skin).

The paper presents a program system for on-line development of database with images, obtained by the cameras of mobile terminals with built-in Java Virtual Machine (JVM). The usage of HTTP protocol to upload the images guarantees 100% mobility of the clients, whose number is practically unlimited. The construction of the database is realized in on-line mode through a WEB application on the base of the information being sent by the mobile clients. The platform independence and portability of the application is guaranteed by the usage of Java technologies both from the client side (J2ME) and the server side (J2EE). The usage of Java by the client's side allows for the application to be installed in a far greater number of mobile terminals as compared to the usage of C# or C++.

The structure of the paper is as follows: Section II reviews the architecture of the suggested application. The application structure at the clients' side is described in section III, and at the server side – in section IV. The experimental results obtained are described in section V. The last section contains the necessary conclusions and indicates the direction for the future improvements of the application.

II. APPLICATION ARCHITECTURE

The proposed application has a typical client-server architecture and consists of two basic modules:

- Mobile Image Database Creator - Client (MIDC-C);
- Mobile Image Database Creator - Server (MIDC-S).

The application architecture is shown on Fig.1

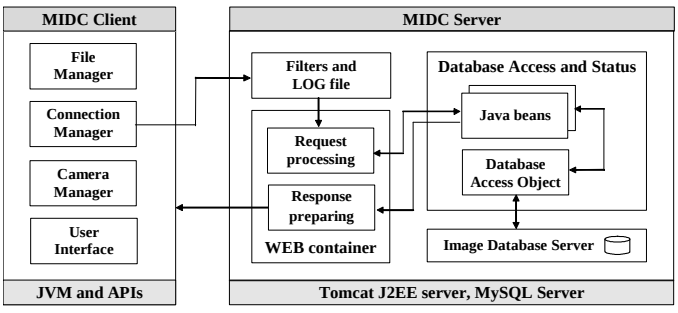

Figure 1. Application architecture.

The program code at the client's side is developed as Java MIDlet, using the J2ME technology. The application needs the

presence of MIDP 2.0 profile [5] and Mobile Media API [6], in order to ensure program access to the camera.

The program code at the server side uses components of the J2EE technology, namely: servelets, filter sevelets, Java Server Pages (JSP) and Java beans.

The communication between the clients and the server is realized on XML-based requests and responses. The usage of Java technologies and XML guarantee the portability and reusage of the program code.

III. MIDC Client Description

The application for the clients is a MIDlet with an entirely graphic interface. Its functionality is realized by the next more important modules:

A. Camera Manager

Through the methods of the CameraManager class, a program access to the camera is realized in order to initialize it and to obtain a snapshot. The availability of Mobile Media API is necessary. The application provides a possibility for program control of the white balance, the lightning, and flash control, if the mobile terminal supports Advanced Multimedia Supplement (AMMS) specification [7].

B. File Manager

Through the methods of the FileManager class, a temporary buffering of a snapshots as a graphics file is possible. When using Java, the images received from the camera can be recorded as a file in several ways:

- Recording the snapshot on the flash disk of the mobile terminal as BMP, JPEG or PNG file. The support of File Connection API [8] is necessary. The upload of the files, obtained to a personal computer can be realized through a data-cable or Bluetooth™ interface.
- Recording the snapshot in the Record Management System (RMS) of the Java-application. This approach doesn't have a great practical sense, as the bigger part of the mobile terminals the RMS memory is with limited volume.
- Program access to Bluetooth™ interface in order to transfer each new snapshot to a personal computer.
- Upload of each new snapshot by means of the HTTP protocol to WEB application, which automatically updates the database and saves image to file. The availability of MIDP 2.0 is necessary.

In the proposed application, when there isn't access to the WEB application, a temporary buffering of the graphic files obtained is envisaged on the flash-disk of the mobile terminal. As the snapshot size is small, PNG encoding is used. By this, a compromise is made between the file size and image quality. The name of the files is in the following format:

IMG-ABCDEF-ID.PNG,

where ID is a 8-symbol string, obtained on the base of the current date and hour; ABCDEF is a hexadecimal number through which the information, contained in the shot, is encoded.

Table I describes the function and the possible values of the parameters A, B, C, D, E and F.

TABLE I
ENCODING THE SNAPSHOT PARAMETERS

Parameter	Function	Possible values
A	Human race	1= Caucasian 2=Afro-American 3=Asian 4=persons of different races in the shot
B	Lighting condition	1= outdoor snapshot, sunny 2= outdoor snapshot, cloudy 3= indoor snapshot, without lightening 4= indoor snapshot, luminescent lighting 5= indoor snapshot, incandescent lamp
C	Number of faces	0=no faces in the snapshot 1=one face 2=two faces 3=more than two faces
D	Drawbacks	0=none 1=snapshot from TV set 2=beard and/or moustaches 4=hair with face color 8=background with face color Combinations among different drawbacks are possible.
E	Face orientation	0=full face 1=half face 2=combination b/n full and half face in case of a number of faces in the snapshot.
F	AMMS settings	0=AMMS not supported 1=white balance on 2=flash on 4=auto-focus on Combinations between the different AMMS settings are possible.

The values of the quoted parameters are obtained depending on the currents settings of the application, selected by the user. This selection is made through the application menu. The parameters for each snapshot are uploaded to the WEB server, thus the online updating of the database becomes possible.

C. Connection Manager

Through the methods from this class, the communication with the server side is realized by means of HTTP v1.1 protocol: registering of the client and the upload of each PNG file (current file or buffered on flash disk). For that purpose, HTTP POST multipart request to the WEB application is used. Furthermore, except the file contents (Base64 encoded), also the following parameters are transmitted to the server:

- File name in IMG-ABCDEF.PNG format;
- Client identification code, received after its registration.

The possibility to send a further file exists, before the contents of the previous one has been sent. For that purpose, a program queue is used, in which the sequence of the files to upload, is recorded. If the RAM memory of the mobile terminal is insufficient, the files are temporarily buffered on the flash disk, if possible.

D. User Interface

The methods from this class realize the access to the program menu and the processing of the events from the keyboard. The user interface as shown in Fig.2 is entirely graphics and maximum simplified.

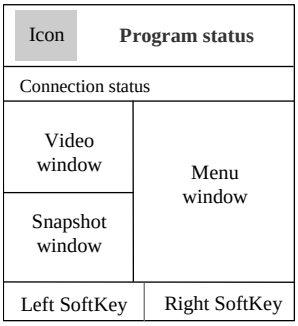

Figure 2. User interface.

The application is adapted to the display size, which must be bigger than 170x200 pixels. The field with video information is with 160x120 or 320x240 pixels depending on the display size.

The following information is visualized on the display:

- Program status;
- Connection status;
- Program buttons (LeftSoftKey, RightSoftKey) – Get, Upload, Save snapshot.
- Video from the camera (Video window). When the user gets a snapshot, the video player is stopped. The user decides whether to sent the snapshot to the server, or to receive a further snapshot.
- Snapshot window (last snapshot from camera).
- Application menu (Menu window).

The elements of the application menu are given in Table II.

TABLE II
MENU ELEMENTS

Menu elements	Function
Upload mode	Image buffering method selection: recording in the flash disk, upload to the WEB server.
Image size	Selection of frame size in pixels. The application automatically gets information for the possible frame sizes, received in Video snapshot mode.
Lighting	Selection of lighting type: sunny, cloudy, room without lighting, luminescent lamp, incandescent lamp.
Faces	Selection of number of faces in the frame: none, 1, 2, more than 2.
Drawbacks	Selection of drawbacks: none, image from TV set, moustaches and/or beard, hair with skin color, background with face color.
AMMS (optional)	AMMS settings: white balance, flash, auto focus.

If the mobile terminal supports AMMS API dynamically, in the application menu an element AMMS is inserted, thus enabling the control of:

- White balance (on/off);
- Flash (on/off);
- Auto focus (on/off).

The navigation in the application menu is realized with the following buttons: (<↑>, <↓>, <→>, <←>, <select>) or with their substitute buttons (<2>, <8>, <6>, <4>, <5>).

IV. MIDC SERVER DESCRIPTION

The programming at the server side is entirely in Java language. As J2EE server is used Apache Tomcat, and the server for database management is MySQL. The WEB application consists of the following major modules:

- Request Processing – a servelet to process all clients' requests and to classify them as valid and invalid. The clients generate two types of requests: HTTP POST, XML-based request for registration, and HTTP POST multipart request upon uploading the contents of PNG files. In case of successful registration the WEB application returns a unique ID number for each client, which is buffered in RMS and is used for every upload request.
- Response preparing - Java Server Page (JSP), by means of which the reply to every mobile client is formed. Each reply is in XML-format.
- Database Access and Status – the support of the application status, the user registrations and the access to the server for image database control is realized by a group of Java beans. This solution guarantees not only the portability of the code, but makes it reusable – keeping the application working upon its dynamic shift to another WEB server. The necessary information for access to the image database (driver name, database name, user name and password) is saved in the configuration file web.xml.
- Filters and LOG file – A servelet filter is used, realizing the restriction of the access to the application in case the request is not from a mobile terminal, and supporting a LOG-file at user level.

Initially, an empty database with a single table is created. Its fields are described in Table III.

TABLE III
IMAGE DATABASE FIELDS

ImageID	Size	Race	NumOfFaces
char (16)	int (1)	int (1)	int (1)

Drawbacks	Orientation	Lighting	AMMS
int (1)	int (1)	int (1)	int (1)

Each image gets a unique 16-symbol identification code which is used as a primary key and in the formation of the file name. The values of the other fields are formed depending on the values of the parameters A, B, C, D, E and F from the name of the PNG-file, transmitted as a parameter of the multipart requests. The database is updated in on-line mode, depending on the customers' requests.

V. EXPERIMENTAL RESULTS

We carried out a number of experiments for the performance evaluation of the proposed client-server application. The experiments have been performed with the usage of the following hardware and software platforms:

- Mobile phone – Nokia 6630 (microprocessor ARM9 220MHz) and Nokia N95 (microprocessor ARM11 332MHz);

- Server – host with microprocessor Pentium 4, 2.6GHz, 1.5GB DRAM, Apache Tomcat 5.0 WEB server, and operating system Windows XP SP2.

Table IV shows the following average time intervals in ms:

- T1 – image capture time and prepare request time;
- T2 – establish connection with server time;
- T3 – upload image file;
- T4 – image file save time;
- T5 – database update time (100 records);
- T6 – response time.

TABLE IV
TIMING RESULTS

Bearer / Interval, ms	GPRS	UMTS
T1	19	19
T2	156	78
T3	6953	4128
T4	18	18
T5	694	694
T6	72	39
Total	7912	4976

According to the results, described in Table IV, major portion of time is taken up for image upload over HTTP channel. Server side takes only 9-14% of total time. When UMTS bearer was used upload time is reduced 1.7 times.

Client side application user interface is shown on Fig.3.

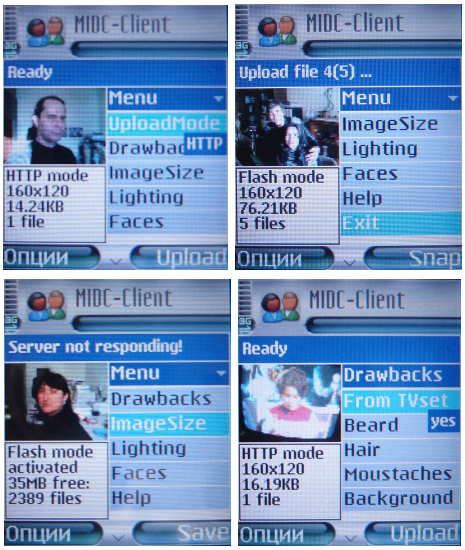

Figure 3. User interface (Nokia 6630).

VI. CONCLUSIONS AND FUTURE WORKS

A client-server application has been developed for online development of databases wit images, received from the cameras of mobile terminals. Such databases may be used for benchmarking of mobile applications for face detection and recognition [4].

The application is distinguished by 100% mobility of clients whose number is practically unlimited; mass applicability; online development of the image database without intervention of administrator.

As a disadvantage of the application the necessity to pay the traffic could be pointed out, realized by the mobile clients. However, this cost is insignificant. For example, for a database with 2000 images (160x120 pixels) with average size of the upload requests of 16KB the price is about 190 BGN, VAT included.

The development of the database can be realized absolutely free, if the snapshots are buffered on a flash disk of the mobile terminal, and transferred later to the personal computer through a data cable or Bluetooth™ interface.

For the future, the development of a WEB application is envisaged, which will enable the administrator of the database to edit and update it in offline mode, and to export the contents under a preset combination of criteria.

REFERENCES

[1] A.C.Loui, C.N.Judice, S.Liu, *An Image Database for Benchmarking of Automatic Face Detection and Recognition Algorithms*, Proc. IEEE Int. Conf. Image Processing, 1998, pp.146-150.

[2] K.K.Sung, T.Poggio, *Example-based Learning for View-based Human Face Detection*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol.20, no.1, 1998, pp.39-51.

[3] P.J.Philips, H.Moon, S.A.Rizvi, P.J.Rauss, *The FERET Evaluation Methodology for Face Recognition Algorithms*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol.22, no.10, 2000, pp.1030-1034.

[4] R.Ivanov, *Algorithm for Face Detection Adapted for Platforms with Limited Resources*, J. Information Technologies and Control, no.1, 2007, pp.37-44.

[5] http://jcp.org/en/jsr/detail?id=118.

[6] http://jcp.org/en/jsr/detail?id=135.

[7] http://jcp.org/en/jsr/detail?id=234.

[8] http://jcp.org/en/jsr/detail?id=75.

[9] http://www.gartner.com/it/page.jsp?id=498310.