AC 2008-2513: DEVELOPMENT OF EDUCATIONAL APPLICATIONS FOR SMARTPHONES

Aleksandr Panchul, UTSA

Aleksandr Panchul received M.Sc. in Computer Science from Moscow Institute of Physics and Technology in 1997. He is currently a PhD student at the Department of Electrical and Computer Engineering of the University of Texas at San Antonio. His research interests include software engineering, digital communications, distributed systems, 3D animation, virtual environments, CPU emulators and mobile applications.

David Akopian, UTSA

David Akopian received the M.Sc. degree from the Moscow Institute of Physics and Technology in 1987 and Ph.D. degree from the Tampere University of Technology (TUT), Finland, in 1997, in electrical engineering. He is currently an Assistant Professor at the University of Texas at San Antonio. From 1999 to 2003 he was with Nokia Corporation. Prior to joining Nokia in 1999 he was a member of teaching and research staff of TUT and a research scientist with the Institute of Informatics and Automatization, Yerevan, Armenia. His current research interests include digital signal processing algorithms for communication receivers, dedicated hardware architectures, positioning methods, and wireless applications.

Development of Educational Applications for Smartphones

Abstract

Cell phones are one of most ubiquitous portable technology devices available. New services are added almost every day and cellular telephony became a bright example of co-evolution of human societies and new information technology.

This paper presents a study of using this widely available platform for educational purposes, specifically for digital signal and image processing (DSP/DIP) education. While cell phones are already used for different educational purposes they were not used for DSP and DIP which are fundamental disciplines in electrical and computer engineering.

The following learning and technology goals are addressed. (1) An educational software toolbox for cell-phones is developed. Image, voice and audio samples can be taken using phone's camera, microphone or downloading. The idea is to link engineering concepts with everyday experiences at any place and any time. The toolbox has a handy user interface and students can download and install it on their phones for applying various built-in DSP/DIP algorithms and visualizing results.

Students can learn simple DSP/DIP concepts without implementing their own algorithms. (2) Students can develop their own applications for cell phones using software development tools for mobile OS, thus learning topics which are normally not covered by conventional software development courses. The applications can be installed on so-called smart phones which allow application download. (3) The toolbox also allows communication of information with a centralized server for assessment in distributed learning environments.

Thus the presented toolbox provides an opportunity to use typical cell-phone data for educational purposes at any place, at any time, and in distributed environments. The idea of developing a cell phone based educational toolbox for DSP/DIP is motivated by recent studies and polls which indicate a wide acceptance of this platform by student constituency.

1. Introduction

Our motivation is based on the current popularity of cell-phone platforms and projected technology advances of mobile hardware and software. Mobile phones are highly popular all over the world. The current trend in mobile technology indicates that the tasks achievable by mobile handsets will soon exceed our imagination. Mobile phone sales worldwide are soaring and in 2004, there were more than 1.5 billion mobile phone users worldwide. This number reached 2 billion by end 2005 fuelled by strong demand from developing economies in Asia and Latin America.

Modern mobile phones are some of the most advanced and technologically complex devices that people use on a daily basis. They are equipped with cameras that can capture still images and also streaming video with reasonable resolution. These phones are also equipped with software

that can read, convert, manipulate and save multimedia in multiple formats. Some phones also have sophisticated software suites that include PC based applications. Modern mobile phones are also equipped with various communication technologies such as the Internet, Bluetooth, Wi-Fi, etc.

Almost all the university students in US own a mobile phone. This provides an excellent opportunity for teachers to post the assignments on special websites so students can access them at any time during the day. Nearly two-thirds of the teachers at Exeter High School in New Hampshire post homework and class assignments for their students each day on HomeWorkNow.com [5], and teachers encourage students to check the site via their cell phones. In Japan, a survey [6] among 333 Japanese university students revealed that 100% of them own a mobile phone, 99% send email on their phones, exchanging some 200 email messages each week. 66% email peers about classes; 44% email for studying. In contrast, only 43% email on PCs, exchanging an average of only 2 messages per week. Only 20% had used a PDA. 71% of the subjects preferred receiving educational materials on mobile phones rather than PCs. 93% felt that its valuable to use phones for teaching. In the UK it is estimated that 81% of 11-15 year olds and 96% of 16-24 year olds have a mobile phone [7]. Similar projects have been established in Europe as well. An example of a pan-European research and development study with partners in Italy, Sweden and the UK is the "m-learning" project [8]. Its aim is to use portable technologies to provide literacy and numeric learning experiences for young adults (aged 16-24) who are not in a full-time education environment. Their experience indicates that the combination of cellular phones and education is here to stay. Another European project, "Mobile Learning project" [9] develops course materials for mobile phones, tests and evaluated those course materials with cohorts of students and disseminates the results of the evaluations through public reports.

The mentioned trend to use mobile phones for education is understandable as people spend more than 50% of their time outside their office or classroom [10]. Recent surveys show that students are willing to use their cellular phones to improve spelling, reading and math [11].

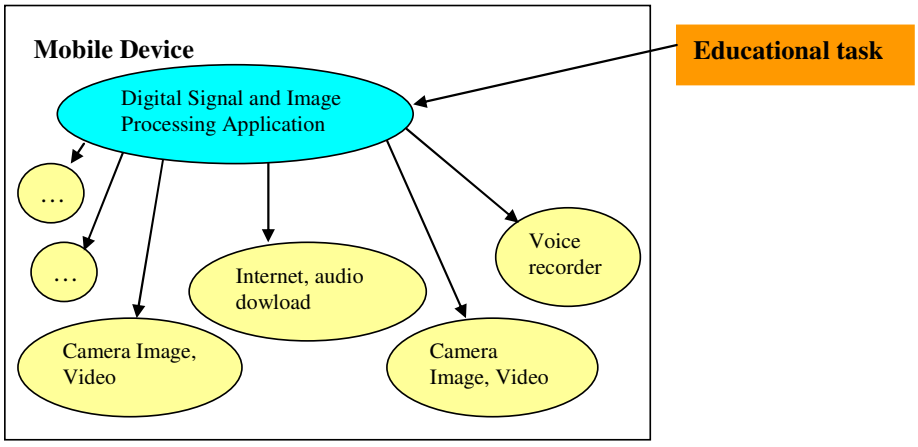


Figure 1. General illustration of the links in an advanced cell-phone based advanced learning technology for DSP and DIP disciplines.

For a wide category of mobile devices known as smart phones external third party applications can be developed and embedded by the users. Educational applications can be developed as small sets of learning experiences on mobile device. Digital signal and image processing (DSP/DIP) disciplines are standard courses in electrical and computer engineering and science. Many excellent books are available and new titles appear almost every year [12]-[14]. Bringing DSP/DIP concepts closer to everyday experiences and usual platforms (phones) will have radical popularizing effect among students of all ages. It will allow using a sophisticated platform with many data collection and communication capabilities (see Fig. 1) for education in distributed environments. This paper presents an introduction to our software and describes sample assignment scenarios. It also outlines advanced capabilities such as algorithm development/embedding and user-to-server (student-to-teacher) communication.

2. The Development Platforms

The perceived value of cell phones is influenced by the availability of high-quality services and applications. Diversification among handset designs and capabilities has increased dramatically. In terms of user interface of competitive devices it caused minimal similarities. The major operating systems currently licensed by the mobile phone manufacturers are Symbian™ OS, Windows for Mobile™, Linux and Palm™ OS. The Symbian™ OS is enormously popular with the software development community due its open architecture, extensive documentation, excellent developer support through Software Development Kits (SDKs). Consequently mobile phones packaged with Symbian™ OS outsold the competition and had a global market share of more than 60% (See Table 1).

Table 1. Worldwide total smart mobile device market, market shares by OS Q2 2005, Q2 2006 [15]

OS vendor	Q2 2005	% share	Q2 2006	% share	Growth Q2 05/Q2 06
Symbian	7,648,920	75.19%	12,720,920	69.7%	66.31%
Linux	1,448,320	14.24%	3,541,870	19.41%	144.55%
PalmSource	496,310	4.88%	562,960	3.08%	13.43%
Microsoft	355,650	3.5%	898,440	4.92%	152.62%
RIM	134,540	1.32%	475,860	2.61%	253.69%
Others	89,490	0.88%	51,360	0.28%	-42.61%
Total	10,173,230		18,251,410		79.41%

One of our goals is to have the applications operational on as many devices as possible accounting various platforms available on the market. Due to prevailing market share the Symbian OS is our first choice for implementation environment. At the same time we also develop alternative software versions for Java ME (J2ME) platform and using Wireless Application Protocol (WAP).

This paper presents our first implementation for Symbian OS using Series 60 developer platform [2]. Nokia has made it available for licensing by other handset manufacturers enabling them to bring phones to market with equivalent and compatible functionality. That means that common APIs and supported technologies allow services and applications to interoperate seamlessly. Series 60 Developer Platform 2.0 contains the following technologies:

J2ME Java APIs	XHTML browsing over TCP/IP
MIDP 2.0	MMS messaging with Synchronized
CLDC 1.0	OMA Digital Rights Management (DRM) (forward-lock)
Wireless Messaging API (JSR 120)	Multimedia Integration Language (SMIL)
Mobile Media API (JSR 135)	Symbian OS v7.0s native APIs
Bluetooth API (JSR 82)	OMA Client Provisioning

Series 60™ platform has a large number of resources for application developers including SDKs, white papers and discussion groups. The SDKs have all the tools required to develop real-world applications including an emulator, target compiler, example applications, and extensive documentation. The following are specific minimum requirements for a device being targeted for a port of the Series 60 Platform:

Display	176 x 208 pixel, 256 color display.
Input	Two soft-keys, five-way navigation, an application launching and swapping key, as well as Send and End keys. To improve and facilitate text input, it includes a Clear key and an alpha toggle key. It uses a standard 12-key number keypad with alpha printings.
Processor	It is recommended that the target device use a 32-bit ARM processor.
Code Size (ROM)	16 MB
RAM Usage	8 MB

3. An Overview of the DSP/DIP Educational Toolkit

The application MEF stands for 'Mobile Engineering Framework'. The underlying basic components of MEF are one- and two- dimensional algorithms. The basic operations are divided into several categories: Manipulations, Transformations and Analysis. Manipulations include straightforward re-arrangement of elements (image rotation, mirroring, transposition, etc). Some of the manipulations are binary operations (e.g. adding two images). Transformations include Fourier transformations and others, normally associated with the information loss and significant change of ranges of values. MEF Analysis category includes a set of image evaluation procedures, such as construction of the magnitude histogram or bitmap inversion. The menu structure is shown in Figure 2 along with examples of cell phone screens from this application.

We call 'user scenario' an exercise in which a student runs the compiled application without writing any C++ code, compiling or installing it. (Even though, installing of a cell phone application might be considered also an end user realm these days).

Figure 2. The proposed simple end user demonstration set-up. Mobile phone with a camera, pieces of paper and a pen.

The end user needs to upload and install the application. The details of that are described in [2]. There is a .sis file on the server (http, or any other connection) that is all the user need to have the application on his phone.

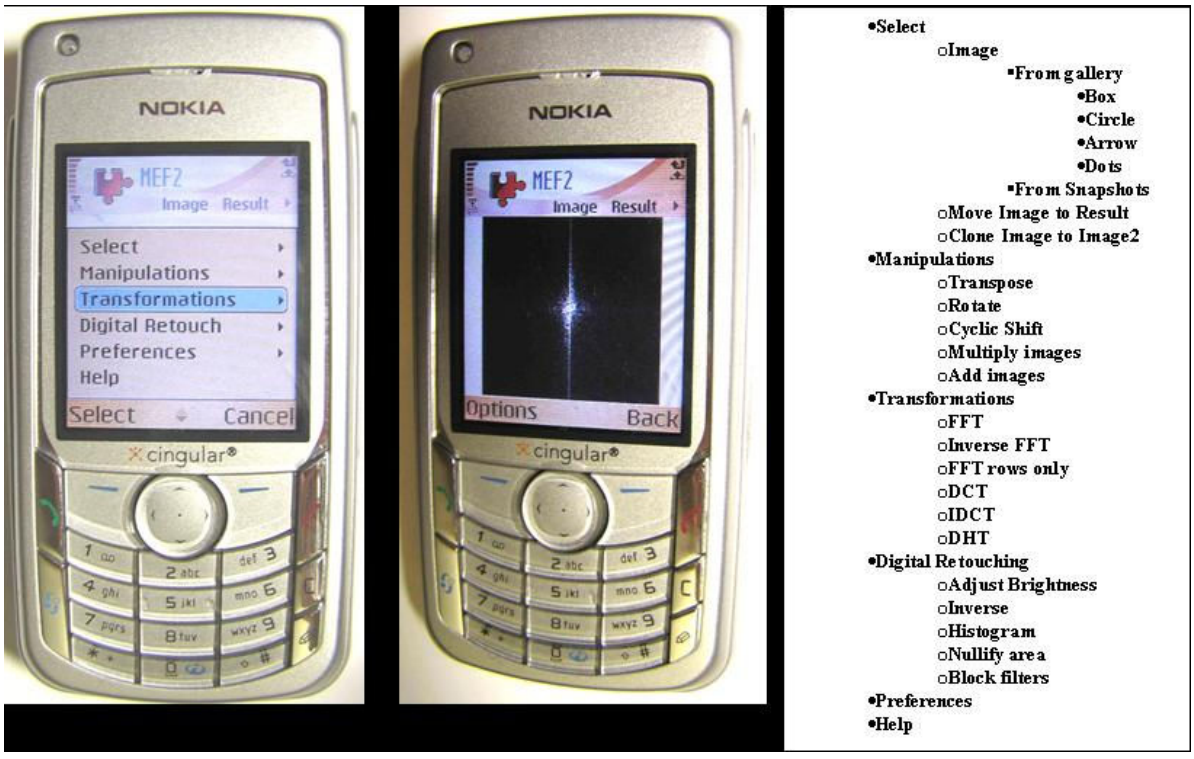


Figure 3. The menu structure (on the right) and an implementation example of a popular fast Fourier transform (FFT) applied to an image in the real phone (on the left)

The application operates on the images taken by the phone's camera, or uploaded (Figure 2). Some special kinds of pictures could be generated inside MEF. It is the images what the application treats as a two-dimensional signals. Cell phone has very limited resolution, so we focused on two-dimensional objects rather than simple one-dimensional signals. Histogram of an image is an example of one-dimensional signal (see Figure 4)

The top level menu includes the following operations:

- Manipulations(Transpose; Rotate; Cyclic Shift; Multiply images; Add images)
- Transformations(FFT; Inverse FFT; FFT rows only; DCT; IDCT; DHT)
- Digital Retouching(Adjust Brightness; Inverse; Histogram; Nullify area; Block filters)

There are three basic planes, they are marked 'Image', 'Result' and 'Im2' ('Image 2' cut short due to the lack of space in the particular platform). Each of these planes is associated with a two-dimensional signal. They have a lot of common functionality, and serve similar to CPU registers. For the unitary operations leave the results in the same register, while binary operations use 'Image' and 'Im2' as operands and 'Result' as a place-holder for the resulting signal.

The 2D signals could be moved or cloned from one plane into another. So, for demonstration purposes the 'Result' plane could be used to store the signal obtained by one way and set to be duplicated by other sequence of actions in the plane 'Image'.

The result of a transformation is normally a set of floating point values. It needs to be quantized and represented by reasonable colors on the screen. There are cases when the distribution is not even, being put into the uniform table, we receive a black screen with few extremely bright pixels. This fact could be observed on the example of the histograms (Fig.4).

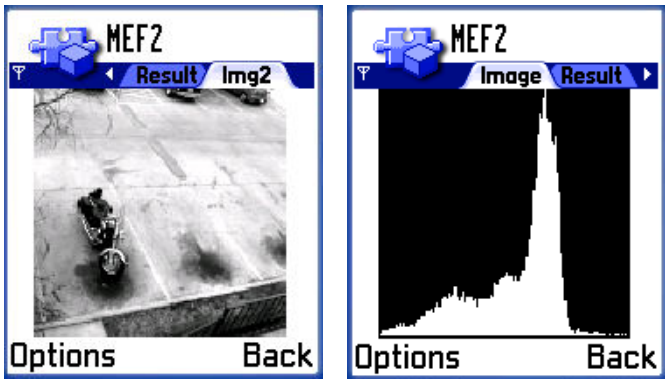


Figure 4. Example image and its histogram.

MEF includes almost all commonly used transformations and manipulations, e.g. see [1] or [3].

4. Examples of "User" Processing Scenarios

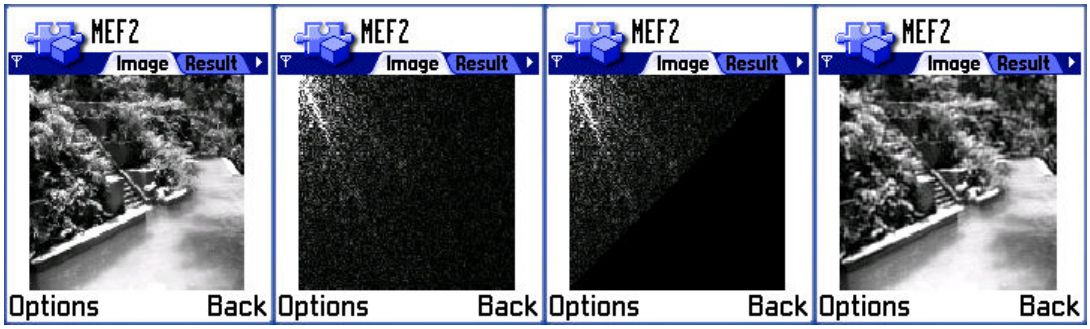
The demonstration scenarios variety is unlimited. In this section several example demonstrations are described. Students will be able to study many aspects of existing DSP/DIP algorithms and add their own. Few examples:

- When selecting an image, check how the processing results are different for symmetric and asymmetric objects. For example, if an image of a rectangle was rotated 180 degrees, the result is indistinguishable from the one obtained by a mirroring. MEF sample set contains an

image “Arrow” that does not contain symmetry of any kind. “Box” image, on the other hand, has central symmetry and is symmetrical with respect to vertical and horizontal axes. A circle is most symmetrical object.

- Different textures have different signatures. When texture is too small, its orientation might not be easily identified. Students can try to experiment with objects having different textures. Many tests or demonstrations are best performed with just simple hand-drawn figures.
- When performing tests with the perspective of the loss of information (e.g. image compression), students can study textures of different scales within one image simultaneously. When the loss of information occurs, it might affect different textures differently.

These examples are just few of many possible scenarios. Next we describe an experiments that can be performed with the proposed tool to study fundamental DSP/DIP concepts as an example of an educational task. In this study the compression properties of Discrete Cosine Transform (DCT [14]) are studied. DCT is used e.g. in JPEG compression standard. It is also used in many video compression standards. For this demonstration, natural images are the best. Figure 5 shows a demonstration of performing operations on a mobile phone. It can be observed that the energy is accumulated on the up-left corner. Low intensity (darker) area corresponds to low energy DCT coefficients.



(a) (b) (c) (d)

Figure 5. The compression property of DCT for JPEG imaging format: (a) original image; (b) DCT of the image; (c) half of DCT coefficients is discarded; (d) original image reconstructed

An example of assignment:

1. Select an image.
2. Perform DCT on it
3. Using Digital Retouching change values of an area of the image into 0.0
4. Perform Inverse DCT
5. Observe that the original image is still identifiable even though a large fraction of coefficients is nullified.

5. Developer Scenario

University has a constant flow of the students who has their very first experience with not only cell phone programming, but also with programming in general. That is why the toolkit includes

means for own software application development. If beginners can use included libraries for the experimenting without any additional programming, then advanced users can use our development templates to include their own algorithms. The basis of the application and its building blocks could be dissected to show how it works inside and why it was the best way to do it.

Development and debugging

The development of the application has the cycle: planning, writing the code, compiling, testing, debugging, changing the original requirements(back to the first step, planning).

Figure 6. Top level menu items as seen on emulator (on the left) and the user menu.

It is a normal software development cycle except for the fact that most of the development is being done on the simulator instead of the actual cell phone. The look and feel is basically the same, but things get slightly complicated when it gets to the resources, internalization and network connections.

The description of the GUI and some of the String resources are defined in separate files. If the SDK installed properly, the makefile, or .sln file of MS Visual C++ prepare all the components seamlessly, but it could be to some degree confusing as to the proper way to manage resources. We adopted the way resources are categorized in the example applications of the Symbian 60. That is, out of all the directory structure (aif, data, gfx, inc, sis, src, tsrc), the folder 'data' and 'inc' hold most of the definitions - .rss files for the dialog definitions and .loc file for localization data. GFX is the folder for the images and .mbm containers.

Development of a custom algorithm class

The algorithm implementation a regular student is most likely to write will have a number of global variables or constants, maybe some auxiliary functions or initializing procedures. To hedge the system from all the problems, the algorithmic block is required to be derived from the CMEFProcessor class. It has abstract method Process():

virtual int Process(const CMEFSignal *inputsignal, CMEFSignal *Outputsignal)=0;
It is this method that after the instantiation is being called by the GUI class handling the events from the View.

So, the minimalist definition of a custom algorithm implementation may look like:

```
#ifndef __COLOR_CUSTOM1_H__
#define __COLOR_CUSTOM1_H__
#include "MEFProcessor.h"
#include "MEFSignal.h"
class CMEF2AppUi;
class C2D_ColorCustom1 : public CMEFProcessor
{
public:
        C2D_ColorCustom1(CAknViewAppUi* myiUIptr);
        virtual ~C2D_ColorCustom1(void);
        virtual int Process(const CMEFSignal *inputsignal, CMEFSignal *outputsignal);
};
#endif
```

There of course could and should be added all the internal variables and static data. Instead of explaining a student what static data is, the whole processor class is made into a singleton structure. One instance of it is created if it is being called and later reused. In dealing with signal processing it would save significant time and resources if the algorithm contains any kind of a constant array with lengthy sequence that is being correlated during the block run.

Compiling and installation. Attaching a new algorithm to the MEF GUI.

There are several ways to compile the code, most convenient for debugging is the one via the IDE. Another way is to do it from the command line, using the makefile. Press button F9 from MS Visual C++ to compile, or refer to the Symbian documentation for the instructions on compiling from the command line [2].

The application contains several resource definition files that might require some additional utilities to be processed and Nokia PC Suite for the installation into the phone.

The newly created class definition .h file need not be included into the project specification .mmp file of the application. Instead the source code .cpp file should be put via the following statement:

```
SOURCEPATH ..\src
SOURCE  ColorCustom1.cpp
```

Here are the steps to add a new method:

- Open the command line window and go to the following directory: <SDK_installation_directory>\Series60Ex\MEF2\group\

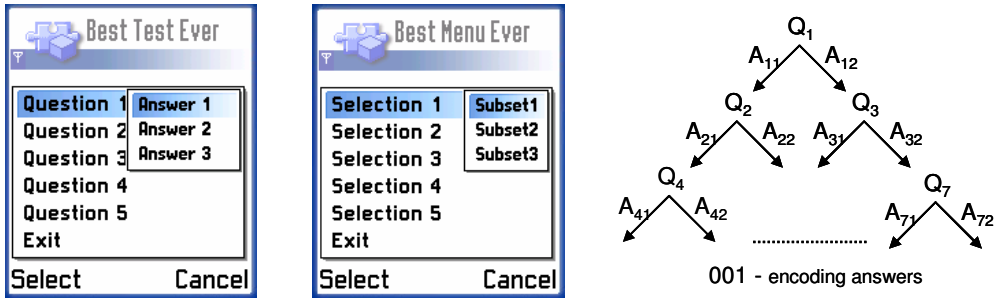
- Create the Microsoft Visual Studio .NET solution and project files: makmake mef2.mmp vc7. This will create the mef2.sln solution file and mef2.VCPROJ project file in the mef2\group\ directory.
- Start the Microsoft Visual .NET IDE and open the solution file, mef2.sln, by selecting File > Open Solution from the menu bar. The Visual C++ IDE opens a new project window for the project, the MEF2 project window.
- Build the application from the Microsoft Visual .NET IDE. Select Build > Build Solution from the Main menu bar (or press Ctrl+Shift+B). This will compile the source files and link the object files into a udeb build of the MEF2 application. The udeb build enables you to view the application in the emulator.
- Start the emulator from the Microsoft Visual .NET IDE. Select Debug > Start from the Main menu bar. When starting the first time, the Executable For Debug Session dialog appears. Browse to the <SDK_installation_directory> and locate \epoc32\release\wins\udeb\epoc.exe.

A description of the windows ideology and the event-driven interface in general is given in [4].

There are additional steps as to how to install the application into the cell phone. The manufacturers of the phones provide this information. It could be done via Bluetooth, Internet or any other data transmitting technologies.

6. Distributed Data Collection and Processing on the Server (Student-Teacher Communication)

One of the recent extensions of our mobile phone library is the ability to send information to a server. Server interaction could consist of some type of a quiz. For example, if assignments are sent to students then they can answer questions through consecutive menu selections (Figure 7a). Answer selections can be encoded using an answers' tree and communicated to the server (Figure 7c). Communication with the server will be implemented using "http" connection. On the server each set of answers encoded by a binary combination can be used for automatic grading using a predetermined grading table or the teacher may choose to do it manually. Text and image communication option between the mobile and the server will be also available. The server application will be implemented as a Java servlet which can be placed in the Apache Tomcat container. Received data will be placed in a database.



(a) (b) (c)

Figure 7. (a) an example of questions & answers menu; (b) an example of assignment selections; (c) encoding answers for communication with the server for the assessment.

7. Future Work

The proposed toolbox will be implemented as an educational hands-on educational toolbox including many modules. It can be used in various courses which address signal and image processing concepts. It will be provided as an open source material with detailed documentation on how to use it and how to extend by incorporating new algorithms. We will use multiple methods of data collection for assessment purposes. This will allow us to surface patterns of uses and attitudes. Examples of planned data collection: (a) Assessment by students. The web portal will have download rate counters for each project module and visitors can rank modules and the portal overall. Also portal access and download rates will indicate the interest in our program. (b) Distribution of questionnaires and feedback collection. Surveys will include items to measure personal attitudes toward mobility, attitudes toward media formats, and attitudes toward included assignment

8. Conclusions

This paper describes an educational toolbox developed on cell phone platforms for engineering and science students. We demonstrated that educational content can be embedded in phones and it can address wide audience because of massive phone availability and usage. The toolbox can be used by diverse student population from novices to advanced users. The beginners can simply check various processing scenarios by menu selections. Advanced students can embed their own algorithms. Our approach will help to motivate studies in engineering and science disciplines. The cell phones are already the part of student lives and our software will deliver signal and image processing concepts to this platform and allow them to experiment at different levels.

Bibliography

1. Discrete-time Signal Processing by Alan V. Oppenheim, Ronald W. Schafer, with John R. Buck. 2nd ed. ISBN 0-13-754920-2
2. Series 60 2nd Edition SDK for Symbian OS, Supporting Feature Pack 2.
3. Digital Communications, Bernard Sklar, 2nd ed. ISBN 0-13-084788-7
4. Windows 2000 Graphics API Black Book by Damon Chandler and Michael Fotsch ISBN 1-57610-876-7
5. C. Branigan, "Schools dial up cell-phone content," eSchool News online at www.eschoolnews.com, Oct. 20, 2004.
6. P. Thornton, C. Houser, "Using mobile phones in education," 2nd IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE'04), 2004.
7. J. Lubega, R. McCrindle, S. Williams, U. Armitage, I. Clements, "Uses of mobile phones in higher education," In Cantoni & McLaughlin (eds) Proceedings of ED-MEDIA 2004, Lugano, Switzerland.

8. J. Attewell, C. Savill-Smith, “Mobile learning and social inclusion: focusing on learners and learning,” (2003), http://www.lsda.org.uk/files/pdf/1440.pdf
9. http://www.leonardo-ireland.com/leonardo2/proj.html#mobile-learning
10. P. Hayes, D. Joyce, P. Pathak, “Ubiquitous learning – an application of mobile technology in education,” in Cantonini & McLaughlin (eds.) Proceedings of ED-MEDIA 2004, Lugano, Switzerland.
11. J. Attewell, C. Savill-Smith, “Mobile learning and social inclusion: focusing on learners and learning,” (2003), http://www.lsda.org.uk/files/pdf/1440.pdf
12. J. G. Proakis, D. K. Manolakis. Digital Signal Processing: Principles, Algorithms and Applications (3rd Edition). Prentice Hall, 1995.
13. S. K. Mitra. Digital Signal Processing: A Computer-Based Approach, 2e with DSP Laboratory using MATLAB. McGraw-Hill Science/Engineering/Math. 2001.
14. R. C. Gonzalez, R. E. Woods, S. L. Eddins. Digital Image Processing Using MATLAB. Prentice Hall; 1st edition. 2003.
15. www.canalys.com
16. .A. Panchul, D. Akopian, "On porting computer applications into Symbian cell phone platform", IEEE Region 5 Conference, April 2006, San Antonio, TX.
17. .A. Panchul, D. Bhupathiraju, S. Agaian, D. Akopian, "An imaging toolbox for smart phone applications", accepted to Mobile Multimedia/Image Processing for Military and Security Applications, SPIE Defense and Security Symposium Symposium, 17-21 April 2006, Orlando, FL
18. Aleksandr Panchul, David Akopian, "Educational Applications Development for Smartphones" UTSA ESB '2007 Conference November 9, 2007
19. Hoanglan Nguyen, Thuy Nguyen, Roseann Trevino, Elizabeth Teran-Delgado, Aleksandr Panchul, David Akopian, "A case study of context-driven application as an example of educational project on cell-phone platforms" UTSA ESB '2007 Conference November 9, 2007