ISSN:0975-9646

EduPad-"A Tablet Based Educational System for Improving Adult Literacy in Rural India"

Mayuri Tawri, Prof. Richa Sharma
CSE, RTMNU

Abstract— Literacy is one of the great challenges in the developing world. But universal education is an unattainable dream for those children who lack access to quality educational resources such as well-prepared teachers and schools. Worse, many of them do not attend school regularly due to their need to work for the family in the agricultural fields or households. However, the rural areas of India are often at a disadvantage within the Indian Education System. An educational system called EduPad, to reduce the rural adult illiteracy using advanced technology. The device proposed here is an interactive Tablet, which is capable of teaching multiple languages. The software helps the user to learn to write as well as spell the alphabets.

Keywords: tablet; android; literacy; rural; educational

INTRODUCTION

1.1 Introduction

Advances in mobile computing have led to the development of a whole range of applications for providing education in a very diverse range of fields, from in situ education in school children with augmented reality [1], to museum guides [2], to developing bird watching skills [3]. It has been argued that new mobile technologies allow for learning programmed to be developed that (by comparison to traditional learning) are more personalized, learner-centered, situated, collaborative, ubiquitous and lifelong [4]. An additional advantage often overlooked (particularly for mobile phones) is that there is a natural affordance for phones to speak into and listen from (i.e. audio interaction). In this way, applications that involve speaking and listening may actually be better suited to mobile phones than the PC environment [5-7]. Such development of mobile and wireless technologies in education has opened up an array of possibilities for the domain of language learning [8], [9]. So far, there have been a few notable trials of language learning applications for mobile phones, dating as far back as 2001 (e.g. Stanford Learning Lab pilots [10]). Chinnery [11] provides a review of several mobile language learning developments since then, and while there have been some interesting uses of PDAs and iPods for language learning (e.g. [12-15], the focus has generally been on delivering simple features such as vocabulary learning and quiz drills in text format. However, if the goal is to further exploit the use of multimedia capability for more complex learning applications, there needs to be a more focused effort in developing design guidelines for these applications. This paper will serve as a preliminary step from which these guidelines could be developed. One of the first issues that need to be addressed is to systematically and comprehensively explore the affordance and design issues for mobile devices in language learning. There is a suggestion that mobile devices are unique in their ability to offer a personal and portable solution for learning as compared to ordinary computer-based learning [16]. However, they are also constrained by their physical properties (generally smaller screen sizes, etc.) and these properties ought to be taken into account in the design phase of any learning solution. As language learning can involve many different modalities (e.g. audio, text, static pictures), and there are a wide variety of different kinds of mobile devices that can be used for learning (e.g. game-based devices, blackberry, iOS and Android phones). Given that some devices (for example, game-based and iPod-type devices) generally offer a larger screen size and easier text entry, they may likely have a better 'affordance' for text and image-based applications. On the other hand, the mobile phone is a device that is specifically designed for speech and is already highly familiar to the potential user base; it might therefore have a better affordance for audio speech learning applications [5-7]. Smartphone's (particularly iOS and Android phones) could offer better affordance by combining phone and PDA functionality, or could fare worse by compromising the features of the two. Apart from the affordance issues on the learner side, there might also be other implementation issues that would only become apparent during testing (e.g. quality of rendered images or audio on certain devices). There appears to be a lack of empirical research that would guide design of language learning solutions for mobile devices. Yet, such work would be tremendously important not only for mobile language learning applications, but any using audio elements (e.g. musical training). To this end, as a first step, we conducted a small-scale learner study to examine prospective learner preferences, which is described in the next section.

1.2 Features

- ✓ Android Application framework enabling reuse and replacement of components.
- ✓ It provides better understanding about language to literate and unknown users.
- ✓ The developed programming has ability to implement in pc, tablets, and mobiles.

SOFTWARE ENVIRONMENT

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Google Inc. purchased the initial developer of the software, Android Inc., in 2005. Android's mobile operating system is based on the Linux kernel. Google and other members of the Open Handset Alliance collaborated on Android's development and release. The Android Open Source

Project (AOSP) is tasked with the maintenance and further development of Android. The Android operating system is the world's best-selling Smartphone platform. The Android SDK provides the tools and APIs necessary to begin developing applications Android platform using the Java programming language. Android has a large community of developers writing applications ("apps") that extend the functionality of the devices. There are currently over 250,000 apps available for Android.

.Features

- **Application framework** enabling reuse and replacement of components
- **Dalvik virtual machine** optimized for mobile devices
- **Integrated browser** based on the open source Web Kit engine
- **Optimized graphics** powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- **SQLite** for structured data storage
- **Media support** for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- **GSM Telephony** (hardware dependent)
- **Bluetooth, EDGE, 3G, and Wi-Fi** (hardware dependent)
- **Camera, GPS, compass, and accelerometer** (hardware dependent)
- **Rich development environment** including a device emulator, tools for debugging, memory and performance profiling, and a plug-in for the Eclipse IDE

Fig 1. Android Architecture

Libraries

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

- **System C library** - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
- **Media Libraries** - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
- **Surface Manager** - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
- **LibWebCore** - a modern web browser engine which powers both the Android browser and an embeddable web view
- **SGL** - the underlying 2D graphics engine
- **3D libraries** - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
- **FreeType** - bitmap and vector font rendering
- **SQLite** - a powerful and lightweight relational database engine available to all applications

Android Runtime

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool.

The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

Linux Kernel

Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

The Linux kernel is an operating system kernel used by the Linux family of Unix-like operating systems. It is one of the most prominent examples of free and open source software.

The Linux kernel is released under the GNU General Public License version 2 (GPLv2), (plus some firmware images with various licenses), and is developed by contributors worldwide. Day-to-day development takes place on the Linux kernel mailing list.

The Linux kernel was initially conceived and created by Finnish computer science student Linus Torvalds in 1991. Linux rapidly accumulated developers and users who

adapted code from other free software projects for use with the new operating system. The Linux kernel has received contributions from thousands of programmers.[10] Many Linux distributions have been released based upon the Linux kernel.

The Linux kernel has extensive support for and runs on many virtual machine architectures both as the host operating system and as a guest operating system. The virtual machines usually emulate Intel x86 family of processors, though in a few cases PowerPC or ARM processors are also emulated.

At Google, the team led by Rubin developed a mobile device platform powered by the Linux kernel. Google marketed the platform to handset makers and carriers on the premise of providing a flexible, upgradable system. Google had lined up a series of hardware component and software partners and signaled to carriers that it was open to various degrees of cooperation on their part.

Speculation about Google's intention to enter the mobile communications market continued to build through December 2006. Reports from the BBC and The Wall Street Journal noted that Google wanted its search and applications on mobile phones and it was working hard to deliver that. Print and online media outlets soon reported rumors that Google was developing a Google-branded handset. Some speculated that as Google was defining technical specifications, it was showing prototypes to cell phone manufacturers and network operators.

Android Operation System

Android is an operating system based on Linux with a Java programming interface. It provides tools, e.g. a compiler, debugger and a device emulator as well as its own Java Virtual machine (Dalvik Virtual Machine - DVM). Android is created by the Open Handset Alliance which is lead by Google.

Android uses a special virtual machine, e.g. the Dalvik Virtual Machine. Dalvik uses special bytecode. Therefore you cannot run standard Java bytecode on Android. Android provides a tool "dx" which allows to convert Java Class files into "dex" (Dalvik Executable) files. Android applications are packed into an .apk (Android Package) file by the program "aapt" (Android Asset Packaging Tool) To simplify development Google provides the Android Development Tools (ADT) for Eclipse . The ADT performs automatically the conversion from class to dex files and creates the apk during deployment.

Android supports 2-D and 3-D graphics using the OpenGL libraries and supports data storage in a SQLite database.

Every Android applications runs in its own process and under its own userid which is generated automatically by the Android system during deployment. Therefore the application is isolated from other running applications and a misbehaving application cannot easily harm other Android applications.

Important Android components

An Android application consists out of the following parts:

- Activity - Represents the presentation layer of an Android application, e.g. a screen which the user sees. An Android application can have several activities and it can be switched between them during runtime of the application.
- Views - The User interface of an Activities is build with widgets classes which inherent from "android.view.View". The layout of the views is managed by "android.view.ViewGroups".
- Services - perform background tasks without providing an UI. They can notify the user via the notification framework in Android.
- Content Provider - provides data to applications, via a content provider your application can share data with other applications. Android contains a SQLite DB which can serve as data provider
- Intents are asynchronous messages which allow the application to request functionality from other services or activities. An application can call directly a service or activity (explicit intent) or asked the Android system for registered services and applications for an intent (implicit intents). For example the application could ask via an intent for a contact application. Application register themself to an intent via an IntentFilter. Intents are a powerful concept as they allow to create loosely coupled applications.
- Broadcast Receiver - receives system messages and implicit intents, can be used to react to changed conditions in the system. An application can register as a broadcast receiver for certain events and can be started if such an event occurs.
- A Java Virtual Machine (JVM) enables a set of computer software programs and data structures to use a virtual machine model for the execution of other computer programs and scripts. The model used by a JVM accepts a form of computer intermediate language commonly referred to as Java bytecode. This language conceptually represents the instruction set of a stack-oriented, capability architecture. Sun Microsystems states there are over 4.5 billion JVM-enabled devices
- A JVM can also execute bytecode compiled from programming languages other than Java. For example, Ada source code can be compiled to execute on a JVM. JVMs can also be released by other companies besides Oracle (the developer of Java) — JVMs using the "Java" trademark may be developed by other companies as long as they adhere to the JVM specification published by Oracle and to related contractual obligations.
- Java was conceived with the concept of WORA: "write once, run anywhere". This is done using the Java Virtual Machine. The JVM is the environment in which java programs execute. It is software that is implemented on non-virtual hardware and on standard operating systems.
- JVM is a crucial component of the Java platform, and because JVMs are available for many hardware and software platforms, Java can be both middleware and a platform in its own right,[clarification needed] hence the trademark write once, run anywhere. The use of the same bytecode for all platforms allows Java to be

described as "compile once, run anywhere", as opposed to "write once, compile anywhere", which describes cross-platform compiled languages. A JVM also enables such features as automated exception handling, which provides "root-cause" debugging information for every software error (exception), independent of the source code.

- A JVM is distributed along with a set of standard class libraries that implement the Java application programming interface (API). Appropriate APIs bundled together form the Java Runtime Environment (JRE).
- Java's execution environment is termed the Java Runtime Environment, or JRE.
- Programs intended to run on a JVM must be compiled into a standardized portable binary format, which typically comes in the form of .class files. A program may consist of many classes in different files. For easier distribution of large programs, multiple class files may be packaged together in a .jar file (short for Java archive).
- The Java application launcher, java, offers a standard way of executing Java code. Compare javaw.[2]
- The JVM runtime executes .class or .jar files, emulating the JVM instruction set by interpreting it, or using a just-in-time compiler (JIT) such as Oracle's HotSpot. JIT compiling, not interpreting, is used in most JVMs today to achieve greater speed. There are also ahead-of-time compilers that enable developers to precompile class files into native code for particular platforms.
- Like most virtual machines, the Java Virtual Machine has a stack-based architecture akin to a microcontroller/microprocessor. However, the JVM also has low-level support for Java-like classes and methods, which amounts to a highly idiosyncratic[clarification needed] memory model and capability-based architecture.

Download the Android SDK

Welcome Developers! If you are new to the Android SDK, please read the steps below, for an overview of how to set up the SDK.

If you're already using the Android SDK, you should update to the latest tools or platform using the ***Android SDK and AVD Manager***, rather than downloading a new SDK starter package. See Adding SDK Components.

Here an overview of the steps you must follow to set up the Android SDK:

1. Prepare your development computer and ensure it meets the system requirements.
2. Install the SDK starter package from the table above. (If you're on Windows, download the installer for help with the initial setup.)
3. Install the ADT Plugin for Eclipse (if you'll be developing in Eclipse).
4. Add Android platforms and other components to your SDK.
5. Explore the contents of the Android SDK (optional).

SYSTEM ANALYSIS

EXISTING SYSTEM:

For language learning process users have book materials, video clips, and Audio system. For using the book material user need some basic knowledge to read. In the case of video clips and Audio system user need proper platform and the existing system is also not it should not be a portable.

DISADVANTAGES OF EXISTING SYSTEM:

- User need proper hardware device for learning purposes.
- It should be difficult to use by literate people.

PROPOSED SYSTEM:

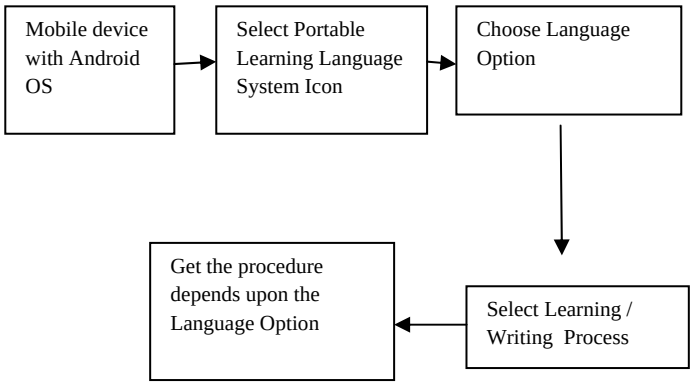
In proposed system we implement the learning process in mobile environment, it provide portable facilitate to the user. Android is a mobile operating system it helps the developer to simulate learning process to our Android based mobile. The Android SDK provides the tools and APIs for developing applications on the Android platform using the Java programming language. Developers write programs in the Java language using Eclipse IDE. Dalvik virtual machine is an interpreter for eclipse IDE it optimized for use on low power consumption, rich libraries, non-fragmented application programming interfaces, low memory devices like phones.

ADVANTAGES OF PROPOSED SYSTEM:

- Android Application framework enabling reuse and replacement of components.
- It provides better understanding about language to literate and unknown users.
- The developed programming has ability to implement in pc, tablets, and mobiles.

IMPLEMENTATION

ARCHITECTURE:



MODULES:

- Learning Alphabet Module
- Alphabet Example Module
- Writing Alphabet Module
- Quiz Module

MODULES DESCRIPTION:

Learning Alphabet

In our system the Alphabets in English language are showing one by one to the user (illiterate user), user having controls to move to the next and previous character, the

Character is shown to the user in big screen, so even elder user can also read the alphabets, and user have the option to change from upper case to lower case and vice versa, the correspondent character in the screen to the user. User can listen to the alphabet while they seeing the alphabet in the screen. It helps the user to learn character and pronunciation too.

Alphabet Example

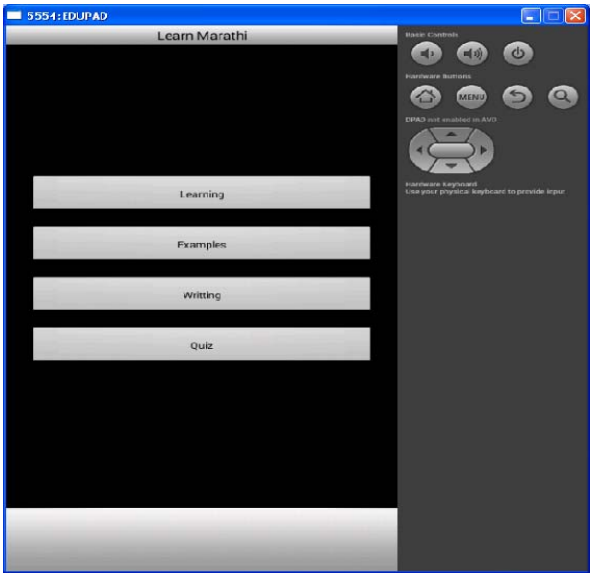
Example for the alphabet is shown to the user with the Object name and its picture, to easily understandable by the user by seeing the object in the screen, and name displayed. And the user has the controls to go to next and previous as in the learning module. When the user presses the play button, the sentence is read out to the user. User can easily understand the alphabet and its usage by listening the example sentence produced as sound.

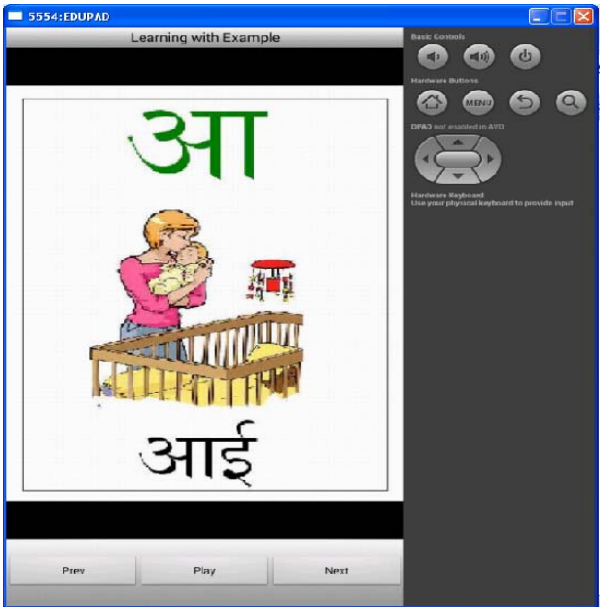
Writing Alphabet

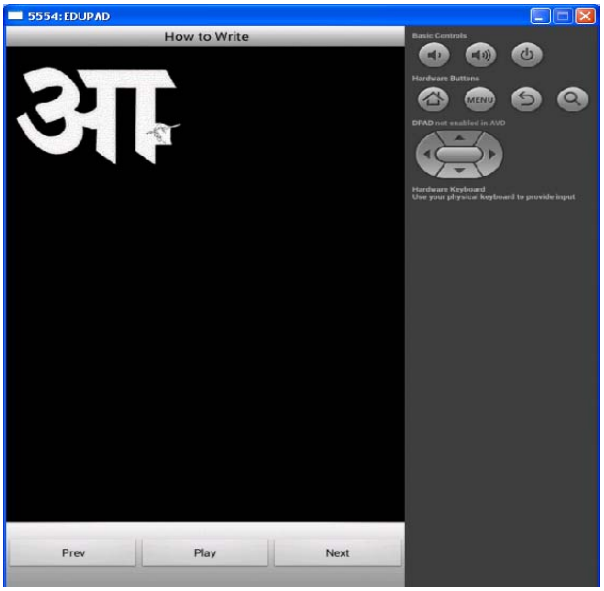
In this module the step by step writing animation for each alphabet is shown to the user. User can try to write the alphabet in the screen itself, by following the animation shown in the screen. User has the option to navigate to the next and previous alphabet and can learn to write all English alphabets.

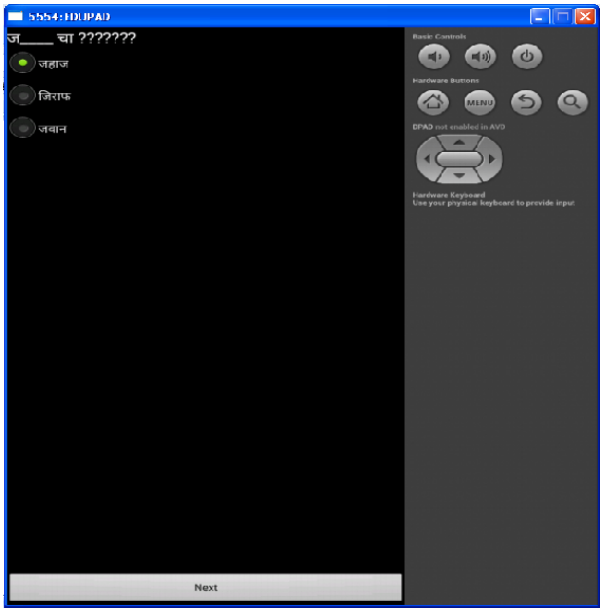
Quiz

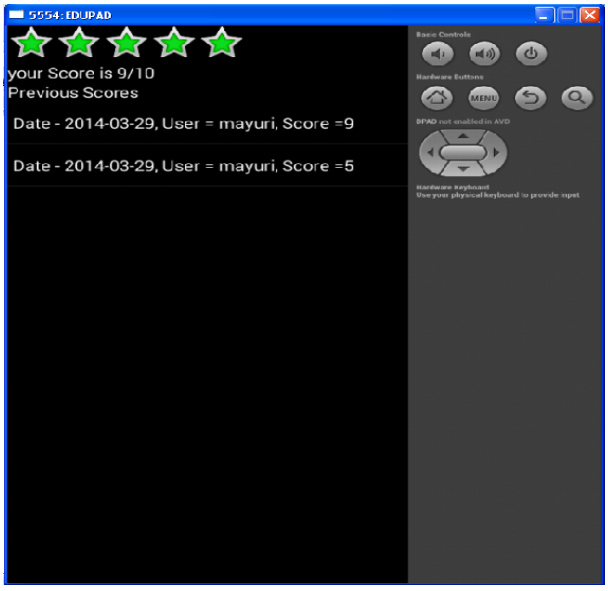
In this module the user can solve the quiz. By this module we can analysis that how much the user can understand. User has the option to navigate to the next question and in the last the user score will be shown with his/her previous history. User can also practice for the witting on the emulator











SYSTEM MODELS

HARDWARE REQUIREMENT

CPU type : Intel Pentium 4
Clock speed : 3.0 GHz
Ram size : 512 MB
Hard disk capacity : 40 GB
Monitor type : 15 Inch color monitor
Keyboard type : internet keyboard

SOFTWARE REQUIREMENT

Operating System: Android
Language : ANDROID SDK 2.3
Back End : SQLite
Documentation : Ms-Office

SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

CONCLUSION

India is a country where the problem of adult literacy still prevails. We believe that the EduPad based educational system will help the illiterate people of rural India to become literate through an interactive and enjoyable method without affecting their day to day life. Most of the illiterate people of rural India rely on manual labor for their living and are unable to attend regular study classes. So, the EduPad can be a convenient method for the s of rural India to become literate.

REFERENCES

[1] Census of India 2001, Office of Registrar General of India.
[2] Census of India 2011, Office of Registrar General of India.
[3] Md. Shafiqul Alam, "Technology based Literacy Education Through Distance Mode in Bangladesh: Problems and Prospects"
[4] EFA Global Monitoring Report, 2011.
[5] Filmer, D., and L. Pritchett. 1999,"The Effect of Household Wealth on Educational Attainment: Evidence from 35 Countries."
[6] Brij Kothari, Avinash Pandey, Amita R Chudgar, "Reading out of the 'idiot box': Same-language subtitling on Television in India," In journal of Information Technologies and International Development Volume 2 Issue 1, Sept 2004.
[7] Matthew Kam, Anuj Kumar, Shirley Jain, Akhil Mathur, and John Canny, "Improving Literacy in Rural India: Cellphone Games in an After-School Program," Proceeding in ICTD'09 Proceedings of the 3rd international conference on Information and communication technologies and development.
[8] Ritu Dangwa, "Public Computing, Computer Literacy and Educational Outcome: Children and Computers in Rural India," Proceeding of the 2005 conference on Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences: Sharing Good Practices of Research, Experimentation and Innovation.
[9] F. He, L. L. Linden, and M. MacLeod, "How to teach English in India: Testing the relative productivity of instruction methods within the Pratham English language education program," working paper, Jul. 1,2008.
[10] J. E. Horowitz, L. D. Sosenko, J. L. S. Hoffman, J. Ziobrowski, A. Tafoya, A. Haagenson, and S. Hahn, "Evaluation of the PBS Ready to Learn cell phone study: Learning letters with Elmo," reported prepared by WestEd, Sep. 2006.
[11] M. Kam, A. Agarwal, A. Kumar, S. Lal, A. Mathur, A. Tewari, and J. Canny, "Designing e-learning games for rural children in India: A format for balancing learning with fun," in *Proc. of ACM international conference on Designing Interactive Systems*, Cape Town, South Africa, Feb. 2008.