CS 297 Report

Enhancing Viewability of Images of Text in PDF in Mobile Devices

Long N Vuong

Fall 2006

Advised by Professor Chris Pollett

Department of Computer Science

San Jose State University

ABSTRACT

This report is a detailed summary of the first semester in Computer Science Master's Writing Project. This report contains research on wireless programming, extracting information from PDF files, and extracting text from image files. In mobile devices, standard PDF readers such as Adobe Reader for Mobile Devices enable one to view PDF files provided they contain mainly text and at most small images. However, if we have a PDF file whose contents contain larger images, equations and scanned text in a large image, etc. then these readers just display the whole image. This can be especially bad if the image is larger than the screen size itself as one has to scroll both vertically and horizontally to try to understand the document. In this project, we will develop a reader which solves the problem of displaying scanned text in large images and which solves the problem of displaying equations.

Table of Contents

Introduction **4**

Deliverable 1: J2ME “Hello World” program **5**

Goals **5**

Implementation and results **5**

Deliverable 2: Extracting text in PDF and save as JPEG images **7**

Goals **7**

Implementation and results **7**

Deliverable 3: Extracting images in PDF and save as various image types **8**

Goals **8**

Implementation and results **9**

Conclusion **11**

Bibliography **12**

Introduction

In the first semester of this writing project, we researched, read about and implemented in three deliverables. The first deliverable involved having a simple “Hello World” program for mobile devices. The second deliverable was a Java program extracting text from PDF file and converts each word to a JPEG image file. The third deliverable was to extract images from PDF file and save them as various image file types. In addition, we researched and learned to extract words in JPEG image files and save them as separate image files.

We will put these deliverables together in the next semester to develop a PDF reader which will detect words and equations in large images using the surrounding white space. The program will save these words as smaller images which can then be flowed to the small screen of mobile devices. Our system will be robust, as it only needs to detect white space rather than do fancier techniques like optical character recognition, which might be hard in the case of math equations, handwriting, or nonstandard scripts.

In the following of this report, we will describe the main goals and the implementations of our three deliverables. The first deliverable was the first step to setup our Java development environment and deploys a program onto mobile devices. The second deliverable was to get to know the PDF file structure and specification. The third deliverable was another program to know more about the PDF file structure and also the processing of different image types in Java.

Deliverable 1:

Warming up with a J2ME "Hello World" program

Goals

The first step to start a project is to setup the development environment; and then testing it with a simple "Hello World' demo program. This deliverable was to produce a simple J2ME demo program to run in mobile device. For this deliverable, we had to read and learn about Sun Microsystems's J2ME technology. We learned to deploy the program onto a mobile device via online server.

Implementation and results

During this development, we had to setup Java JDK 1.5.0, J2ME and Wireless Toolkit 2.5 environment. J2ME is basically a slimmed down version of J2SE. However, J2ME devices require special interface and event handling code and such applications are limited with UI.

The program will start with a TextField asking user to enter his/her name.

After hitting enter button, the program will show a greeting to the user.

We created HelloWorld.jar, Manifest.mf and HelloWorld.jad to deploy the program onto mobile devices via internet. This was not too hard to create those files; however, we had to modify the mine in our online server to handle the files. When a browser on mobile device accesses the jad file, it will prompt user to install the program on his/her mobile device.

Deliverable 2:

Extracting text in PDF and save them as JPEG images

Goals

After successfully creating a developing environment, we now stepped up to the next level that was getting to know the PDF file structure and learning one Java library for PDF - iText - to process PDF files. The goal of this deliverable was to produce a program that extracted text from PDF file and saved them as JPEG image files. To be able to do this program, we read the "PDF Reference – fifth edition" to understand the PDF file structure and its elements. Then, to implement this deliverable in Java language, we relied on a Java Library for PDF that was iText library. The book "iText in Action" was a very good reference to learn about iText Java library.

Implementation and results

At first, we downloaded and added iText library jar file to our program.

This program will read a PDF file that is specified by the user. Then it gets all the reference numbers of pages in the PDF. For each page, it gets the content stream object and extracts string text from the content stream. Then, all the strings will be split into words. For each word, before we save it as a JPEG image, we get its entire boundaries using TextLayout. The program then sets color background of the image, color of the word, font and size of the word. At last, we can save each word as JPEG image file using Java's ImageIO class.

These are sample outputs of the program:

jAvA	PDF	Program
Pic. 1	Pic. 2	Pic. 3

Deliverable 3:

Extracting images in PDF and save them as various image types

Goals

After successfully extracting text from PDF files, we now came closer to an important part of the whole project. We were able to extract images from PDF files. The goal of this deliverable was to produce a program that extracted images in a PDF file and saved them as JPEG, TIFF, PNG or GIF files. To implement this deliverable in Java language, we chose JPedal as Java library to extract images, Sun's JAI to encode TIFF format and Shetline's GIFOutputStream to encode GIF format.

Implementation and results

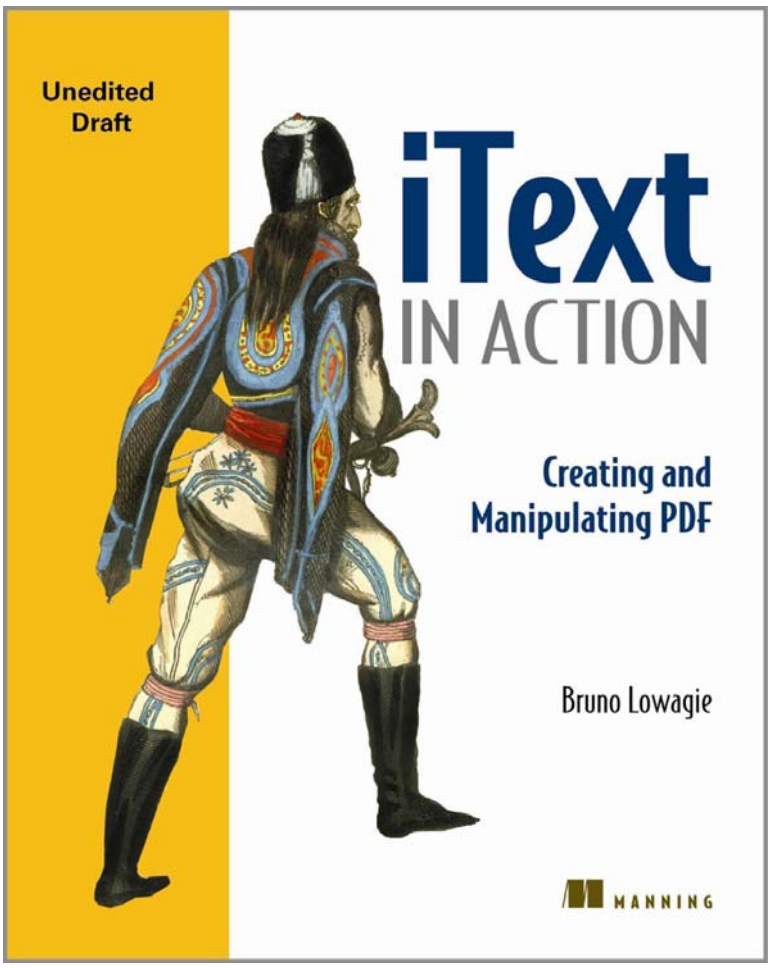
At first, we downloaded and added the JPedal-library jar file to the program. Also we imported the TIFFEncodeParam class and GIFOutputStream class.

The program asks the user to input a PDF file and select the image file format that the user want all the images to be saved as. In this program, user can save all the images as TIFFs, JPEGs, PNGs or GIFs. The program then read the PDF file and decodes it. For each page of the PDF file, it extracts all draw images in the page. And then for each image it extracted, it puts in BufferedImage object. The BufferImage object is saved as a new image file format which was specified by the user.

For TIFF image format, the TIFFEncodeParam class helps us to setup properties of the TIFF file such as compression; and then create the TIFF file with the BuffteredImage object. For GIF image format, the GIFOutputStream class saves the BufferedImage object as a GIF file with 256 Colors that is specified in the program. For PNG and JPEG image formats, the ImageIO class of Java can write the BufferedImage object as PNG or JPEG image easily.

The following are sample outputs of the program:

Pic. 4



Pic. 5

All the goals in this deliverable were achieved. It was a good experience of processing images in PDF file and various image types. Although we want to support more image type formats such as EPS, finding free Java library for it was difficult.

Conclusion

At the end of this semester we built the foundation of our project. We had done and successfully deployed a program to run on our mobile devices. We learned to know the PDF file structure and its elements. Moreover, we could manipulate PDF files with all necessary features that we are going to implement in the next semester such as extracting text and images. Moreover, we have researched and implemented part of the most import part of our project which is extracting word from an image file using the surrounding white space.

In the next semester, we will improve the method to extract words and equations from an image file. At last, we will put together all that we have in the first semester and the second semester into two programs that can extract texts in big image in PDF and render all of them to fit the small screen of mobile devices. The first program is called the PDF_Mobile_Helper and the second program is called The PDF_Mobile_Viewer. The PDF_Mobile_Helper takes in a PDF file, extracts all images in the PDF file and then extracts all texts in the images. Finally it generates a file that will be displayed nicely on mobile devices by the PDF_Mobile_Viewer. Moreover, The PDF_Mobile_Viewer runs on mobile device to read the file that was generated by The PDF_Mobile_Helper and arranges the text to fit the mobile-device's screen. Finally, we will prepare the final report to defense our project.

Bibliography:

[2000] Practical Algorithms for Image Analysis: Descriptions, Examples, and Code. Michael Seul, Lawrence O'Gorman, Michael J. Sammon. Cambridge University Press. April 15, 2000.

[2005] Acrobat SDK User's Guide. Adobe. http://partners.adobe.com/public/developer/en/acrobat/sdk/pdf/intro_to_sdk/UserGuide.pdf.

[2005] Acrobat and PDF Library API Reference. Adobe. http://partners.adobe.com/public/developer/en/acrobat/sdk/pdf/plugins/APIReference.pdf.

[2001] Core J2ME Technology. John W. Muchow. Prentice Hall PTR. December 21, 2001.

[1997] Finding Text In Images. V. Wu, R. Manmatha and E. M. Riseman. http://www.cs.umass.edu/Dienst/UI/2.0/Describe/ncstrl.umassa_cs%2FUM-CS-1997-009.

[2002] Extraction of Text from Images. Pooja Nath. http://www.cse.iitk.ac.in/research/btp2002/98263.html.