Textual Search in Graphics Stream of PDF

A.Balasubramanian and C. V. Jawahar
Center for Visual Information Technology,
International Institute of Information Technology,
Gachibowli, Hyderabad - 500 032
jawahar@iiit.ac.in

Abstract

Digitized books and manuscripts in digital libraries are often stored as images or graphics. They are not searchable at the content level due to the lack of OCRs or poor quality of the scanned images. Portable Document Format (PDF) has emerged as the most popular document representation schema for wider access across platforms. When there is no textual (UNICODE, ASCII) representation available, scanned images are stored in the graphics stream of PDF. In this paper, we propose a novel solution to search the textual data in graphics stream of the PDF files at content level. The proposed solution is demonstrated by enhancing an open source PDF viewer (Xpdf). Indian language support is also provided. Users can type a word in Roman (ITRANS), view it in a font, and search in textual and graphics stream of PDF documents simultaneously.

Keywords

Document Search, Tools and Techniques for Digital Libraries, PDF, Information Storage and Retrieval, Word Spotting.

1. Introduction

Digital libraries (Lesk 1997) are becoming an integral part of our day-to-day life. Huge amount of multimedia information gets archived in digital form for wider use across large network of users. A prominent class of digital libraries, emerging in recent years, digitize large amounts of books and manuscripts (Vamshi *et al.* 2006) (Universal Library) (Pramod *et al.* 2006). The scanning process usually results in digital documents which are primarily images/graphics, and they are restricted in search/access.

Users of the digital libraries need to search the document collections for multiple purposes. They may (a) want to search for a specific document given the meta information (e.g., the title of a book) associated with it, (b) want to retrieve all related materials related to a specific query (e.g., retrieve all information related to the second world war), (c) want to search within a document for a specific word or a string (``show me where the mention of *Harappa* is present in this book or document"). The issues related to search using meta information are reasonably well understood. There are many scalable solutions using XML databases. Content-level access to textual document databases needs many language processing modules to make the search engines effective. In our earlier works (Jawahar *et al.* 2004) (Balasubramanian *et al.* 2006), we have demonstrated content-level access to the document *image* collections present in a digital library. Present work aims to address the last category of user needs; i.e., searching within a document with the help of a document reader/viewer. We propose a solution to search in the graphics stream of PDF and how the solution can be embedded in a document viewer. Our solution complements the textual search facilities available in the existing PDF readers.

Major contributions of this paper are summarized as below:

- We extend the conventional textual search (Section 2.1}) to graphics (image) representation of documents. Conventional text search is based on matching or comparison of textual description (say in UNICODE). These techniques can not be used to access content at the image/graphics level, where text is represented as pixels but not as UNICODE.

- For the first time, the objects in the graphics stream of PDFs are shown to be content-level accessible with the help of word spotting technique. Our earlier solutions for searching in document images (Jawahar *et al.* 2004) and building scalable digital library server (based on greenstone) (Balasubramanian *et al.* 2006) are adapted to the graphics stream of PDF.

- We have implemented the proposed solution in an open source PDF reader (Xpdf) to demonstrate that the textual search is possible in the graphics stream. This involves carrying out feature extraction in an offline manner and matching online with an efficient Dynamic Time Warping (Section 3) algorithm.

- Our solution also allows the query word to be entered as Roman (in ITRANS) and searched in the digitized graphics (image) content of the PDF files (Section 4). Present implementation supports Indian languages. Results are shown for Hindi and Telugu documents.

2. Search in Digital Libraries

A digital library is a library in which resources are available in digital format, accessible by means of computers. The digital content may be locally held or accessed remotely via computer networks. The advantages of a digital library are listed below.

- The user of a digital library need not to go to the library physically; people from all over the world can gain access to the same information, as long as an Internet connection is available. Also, people can gain access to the information at any time, night or day and the same resources can be used at the same time by a number of users.

- An exact copy of the original can be made multiple times without degradation in quality. Traditional libraries are limited by storage space, while digital libraries have the potential to store much more information, because digital information requires very little physical space to contain them. When a library has no space for extension, digitization is the only solution.

- Digital libraries provide access to much richer content in a more structured manner, that is, we can easily move from the catalog to the particular book then to a particular chapter and so on. User is able to use any search term up to the word or phrase in the entire collection. Digital libraries can provide very user-friendly interfaces, giving quick clickable access to its resources.

Digital libraries need to support various types of searches. This includes search at the meta data level. Basic meta data search can be limited to the title of the resource, the description, the assigned subject categories, or related URLs. Advanced search offers multiple options including boolean searches, ability to turn stemming off, and limits to category of a record.

Search by Meta data: Digital libraries contain huge wealth of multimedia data, like the video, audio, digital photographs, scanned document images and other multimedia data. But the multimedia data that is available cannot be searched at the content level. If one were to provide additional information, such as the meta information about them, then one can retrieve these documents that are indexed by the keywords based on the meta data. For example, the meta data for a scanned book, are its genre, author, title, ISBN, etc. Meta data associated with the item is often manually entered. In special situations, meta data could also get generated automatically.

Search by Content: This is the most popular form of searching. Many users prefer to search at the content level rather than at meta data level. Search engines like Google is widely used for searching the web-pages based on content. Content level search is more powerful and useful. However, it is difficult to obtain in many situations, especially in presence of images.

2.1 Portable Document Format in Digital Library

More than 200 million Portable Document Format (PDF) (Adobe Systems Inc. 2003) documents on the web today serve as evidence of the number of organizations that rely on PDF to store information. Today, PDF is the *de facto* standard for electronic exchange of documents, and also an industry standard for intermediate representation of printed material. The goal for developing PDF was to enable users to exchange and view electronic documents easily and reliably, independent of the environment in which they were created. Origins of the PDF (Adobe Systems Inc. 2003) dates back to the nineties. During those days, PostScript page description language was the widely accepted standard for printing purposes. PDF was built on top of the PostScript, so that not only it supported printing but also supported viewing capability. PDF *document* is a collection of *object*s. These *objects* can be located in a PDF file in any arbitrary order, but are connected to each other by a reference mechanism. Therefore, a viewer application should process a PDF file by following references from o*bjec*t to *object*, instead of processing *object* sequentially. Hence, every PDF file contains a *cross-reference table*, stored at the end of the file. Cross-reference makes sure that a PDF file containing very large number of documents can be accessed efficiently with almost no time constraints. A PDF document can be regarded as a hierarchy of objects contained in the body section of a PDF file. At the root of the hierarchy is the document's catalog dictionary. Most of the objects in the hierarchy are objects named *dictionaries*. For example, each page of the document is represented by *page object*, a dictionary that includes references to the page's contents. The individual page objects are tied together in a structure called *Page Tree*.

3. Word Search in a PDF file

A PDF file encapsulates a complete description of the document that includes the text, fonts, images, and 2D vector graphics. Importantly, PDF files do not encode information that is specific to the application software, hardware, or operating system used to create or view the document. This feature ensures that a valid PDF will render exactly the same regardless of its origin or destination. A PDF document is a data structure essentially made of *Objects*. A *Content Stream* is a PDF stream object whose data consists of sequence of instructions describing the graphical elements to be painted on a page. Each page object is represented by one or more content streams. Content streams are also used to package sequence of instructions as self-contained graphical elements, such as forms, patterns, certain fonts, and annotation appearances. PDF serves purpose for fundamentally two kinds of applications: the Producer (PDF generator) and the Consumer (PDF reader). Today, we have many opensource PDF viewers available, the popular among them are the Xpdf, Ghostscript and KPDF for Linux platforms and proprietary viewer such as the Adobe Acrobat for the Windows operating system. Each of these applications implement the textual query search, with additional functionalities such as searching a sequence of pages, specific selected pages, case-sensitive search, searching as a regular expression and provide navigating functionalities such as forward and backward search.

Figure 1 shows the entire procedure that a typical PDF reader application does when handling a PDF file for viewing and searching purposes. As shown in Figure 1, the PDF file structure consists of a one-line header identifying the version of the PDF specification to which the file conforms, a body containing Text and Graphic stream objects that make up the PDF, a Cross-reference table containing references to indirect object, a trailer giving the location of the cross-reference table and certain other special objects. A user typically presents a search string, and the *Text Search* module looks into all the *Text Stream* objects in a PDF file and displays the results to the user. This is how conventional PDF search works. We used an existing PDF reader application and modified it so as to implement the *Word Image Search* (within double rectangle) that looks into the *Graphics Stream* objects in a PDF file to match word images. The modules indicated by a double rectangle in Figure 1 are the routines plugged inside a conventional PDF reader application to handle word image search. The results from both the search modules are integrated and is presented to the user making the whole process of searching transparent to the user.

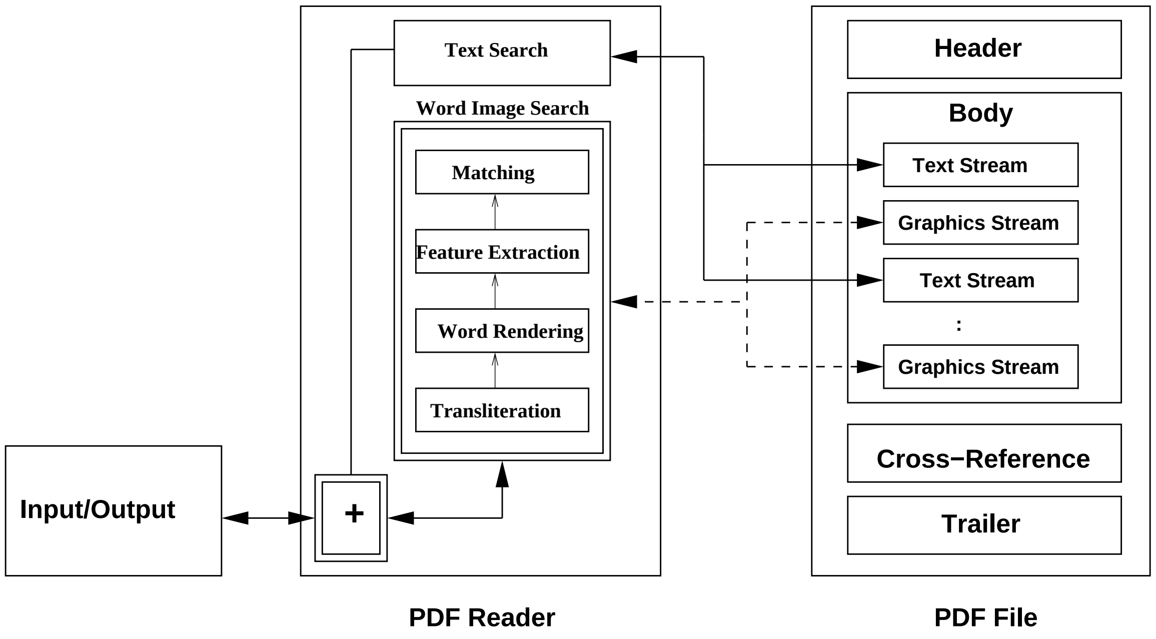


Figure 1: PDF Search Procedure. Our proposed search scheme integrates the conventional textual search in a transparent manner

3.1 Text Search in a PDF file

Most of the PDF readers/viewers support search (find) of query text. This search mechanism is handled *page* by *page* in a PDF file providing a means for inline searching. Whenever the user searches for a particular word, the word has to be searched in all the available pages of a PDF file. Typically, the search starts from the current page till the last page in the file and then continues from the first page. Every block (*Text Block*) in a page, is searched in a top-down sequential order. Text from every line in a block is extracted and is then compared with the input search string. The comparison is not straightforward, as the text extracted from these lines are read character by character (including spaces) from a line and then matched with the input search string. Additional information on search is handled appropriately during the search, like ignoring the case while searching. In this case, both the input search string and the text characters in the line are both converted to lower case (or upper case) and then compared. Information such as searching forward in a page or backward in page are also explicitly handled by the PDF reader applications. These search operations are PDF reader application dependent as some of them are capable of handling search at multiple lines and across blocks, while some of them handle only at the line level. Once a match has been found, the search string in the PDF file is shown in reverse video. The display related functionalities are handled separately and they are dependent on the *user coordinates*, which typically is dependent on the resolution of the display, the current zoom of the reader application and other such device dependent features.

3.2 Searching in the Graphics Stream

OCR-based Search: We often find PDF files that contain document images stored in the graphics stream. Document image contains textual information in the form of an image. Most of these images are scanned copies of technical reports, papers, journals or books. A PDF reader cannot extract text from an image and thus making it impossible to search within document images embedded in the graphics stream. During a typical text search in PDF documents, the image area is now ignored as it is not the region of interest, and is handled only during display related operations. It is sometimes in the interest of the user to search words within an image. One solution to the above problem is to use an Optical Character Recognizer (OCR) to convert the image data into text so that this text is available to the user to search. OCR indeed looks like a veritable solution and some applications (Adobe Acrobat 7.0 Professional) have used such techniques to search text inside images in a PDF file. This solution is better suited for languages that use the Latin Script. The fact that we have commercial OCRs for English with high accuracies makes the above approach possible. OCRs for Indian languages and other oriental scripts are not known for high accuracies and this makes the approach less effective for word search in PDF files that contain document images in Indian or oriental languages. Also the fact that Indian language text is non-standard (represented in custom fonts) makes even the textual search a difficult problem.

Recognition-Free Search: Word level matching has been attempted for printed (Santanu *et al.* 2003) documents. They are useful for locating similar occurrences. None of these matching schemes are designed to do partial matching, which is very important for addressing word-form variations. There have been successful attempts on locating a specific word in a handwritten text by matching image features for historical documents (Rath, Manmatha 2003). Word Spotting (Rath, Manmatha 2003) is a technique wherein word images are matched using various image matching techniques. Dynamic Time Warping is a dynamic programming based procedure (Rath, Manmatha 2003) to align two sequences of feature vectors. This can also provide a similarity measure. This is a popular technique in speech analysis and recognition. Indexing and retrieval from document image collections were studied by converting the images to text (Doermann 1998). Success of these procedures depends on the performance of the OCRs, which convert the document images to text. Manmatha *et al.* (Rath *et al* 2004) built a search engine for historical manuscript images, wherein the retrieval system was trained using an annotated set of 100 pages of George Washington's manuscripts and is used to query a dataset containing images from the same collection. The current approach is to use meta data or indices, which are manually created. This makes automatic approaches to searching and accessing the content very attractive. Another approach to such a problem is to use handwriting recognizers followed by a text search engine. However, in real life, the documents, especially the historical documents, are of poor quality which makes the handwriting recognizers vulnerable to poor results. Rath *et al.* (Rath *et al* 2004) used an alternative approach bypassing explicit recognition. We (Balasubramanian *et al.* 2006) employed a similar methodology to retrieve printed document images using word matching techniques without recognition.

3.3 Recognition-Free Search in PDFs

We employ a similar idea for searching within the graphics stream of PDF files. This involves the following steps

- **Extraction of graphic streams**: Document images are extracted from the graphics stream of a PDF file in its entirety.
- **Word Segmentation**: The pre-processed image is then subjected to word-level segmentation and all the word images in a document image are then extracted.
- **Feature Extraction**: Feature values are extracted from these segmented word images that are
- used for matching purposes.
- **Matching**: Word Image matching techniques are employed to match word images using their
- feature values.

Feature Extraction: Word images are matched at their feature level. Feature extraction is one of the most important tasks after a word image has been segmented and extracted from a document image. Generic content-based image retrieval systems use colour, shape or/and texture features for characterizing the content. In the case of document images, features can be more specific to the domain as they contain image-description of the textual content in it. The features include profile features such as upper and lower word profiles and projection profiles. Further more, structural features such as normalized moments, first order moments, and statistical moments such as the mean, standard deviation, and skew are used. Word image features include the transform domain representations like Fourier coefficients. Feature values are normalized such that the word representations become insensitive to variations in size and font and various degradations commonly present in the text documents.

Dynamic Time Warping: The matching of two words based on the features are carried out using a dynamic time warping algorithm. Let the word images (say their profiles)
are represented as a sequence of vectors ***F* = *F_1, F_2, … F*M *and G* = *G*1, *G*2, …. *G*N.** The DTW-cost between these two sequences is D(M, N), which is calculated using dynamic programming is given by:

$$D(i,j)=\min\begin{Bmatrix} D(i-1,j-1) \\ D(i,j-1)+d(i,j) \\ D(i-1,j) \end{Bmatrix}$$

where, **d(i,j)** is the cost in aligning the **i**th element of F with **j**th element of G and is computed using squared Euclidean distance:

$$d(i,j)=\sum_{k=1}^{N}(F(i,k)-G(j,k))^2$$

Using the given three values **D(i, j - 1), D(i - 1, j)** and **D(i - 1, j - 1)** in the calculation of **D(i, j)** realizes a local continuity constraint, which ensures that no samples are left out during time warping. A global constraint using Sakoe - Chiba band (Sakoe *et al.* 1980) is imposed so as to ensure the maximum steepness or flatness of the DTW path. Score for matching the two sequences F and G is considered as D(M,N), where M and N are the lengths of the respective sequences.

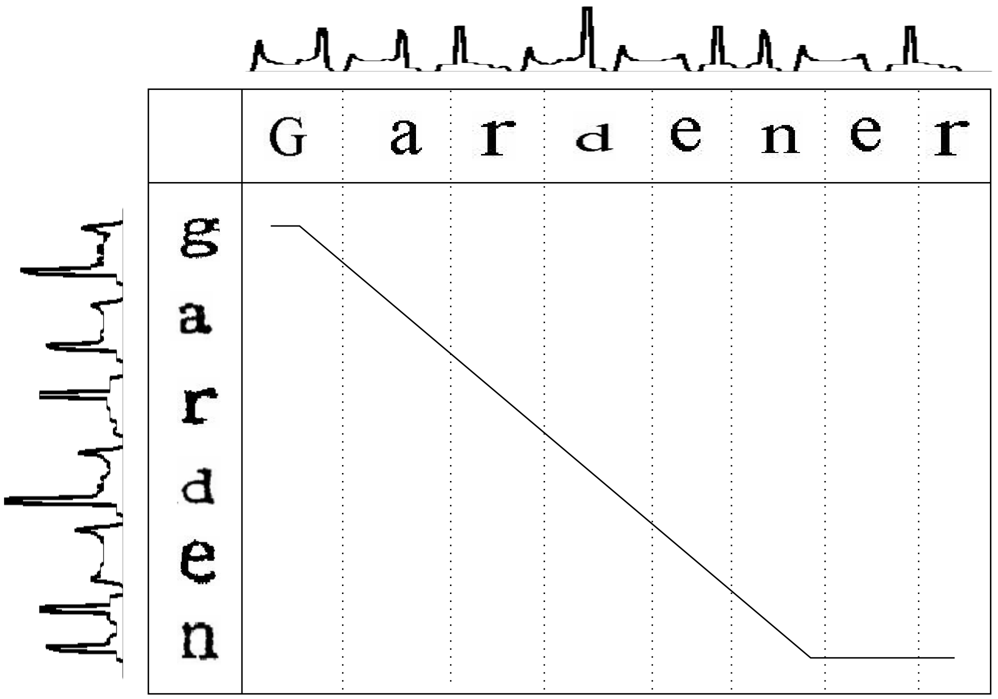


Figure 2: Dynamic Time Warping (DTW) Plot during Matching

A sample plot of matching profiles of two word images, ``Gardener'' and ``garden'', is shown in Figure 2. It can be seen that the last few characters of ``Gardener'' are not really contributing to the dissimilarity computation as trailing sequence of characters do not have matches. Also, note that the starting letter, "`**g**"} and "**G**", at the image level are different. It can be observed that for word variants the DTW path deviates from the diagonal line either in the horizontal or in the vertical direction from the beginning or end of the path, which tremendously increases the matching cost. After the matching process using the DTW, the path is backtracked so as to get the minimum distance traversed. Figure 2 shows the upper profile feature values for the above mentioned sample words.

3.4 Structure of the Implementation

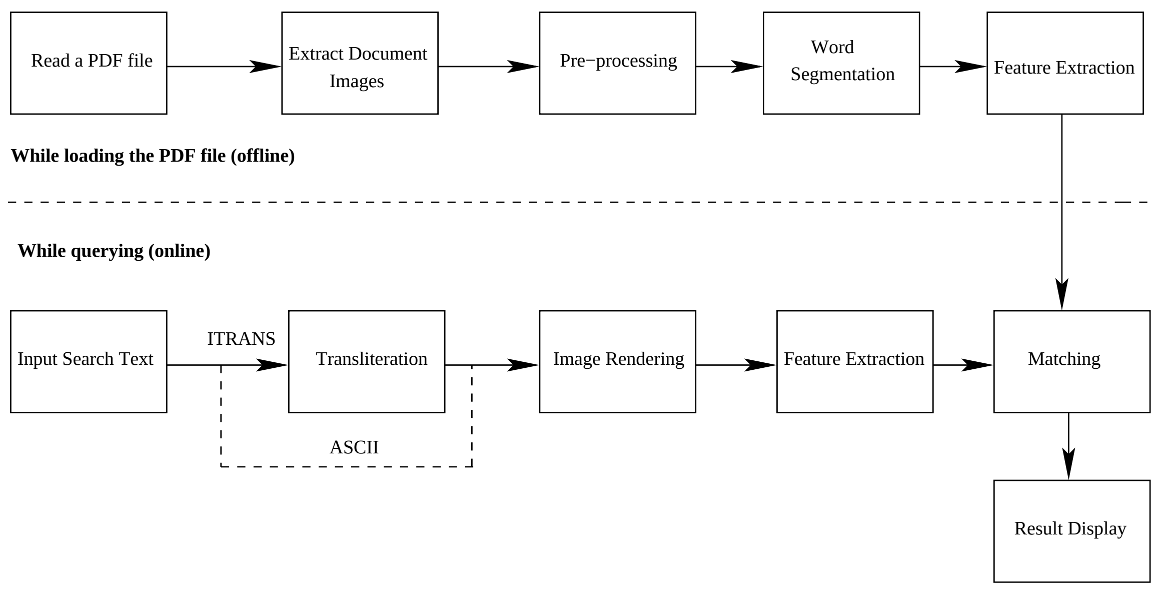


Figure 3: Block diagram illustrating our approach.

The graphics stream of a PDF file has been only used for viewing purposes and thus making it inaccessible to search. We have used the existing text search framework in PDF files to locate and then search graphic stream objects (refer Figure 1). The entire process of locating the graphics stream objects in a PDF file and then searching and matching is explained below. As can be seen from Figure 3, there are two stages: the PDF file load stage and immediately after loading, the search stage, separated by the horizontal dashed lines at the middle. During the load stage, a PDF file is read as it is normally read. All images in the PDF file are then extracted in the *Read PDF file* module. In this routine we extract all images in its entirety (full size) and some additional information such as, its width, height, location in the *page* of a PDF document, and page number are extracted and stored in a separate data structure. Since every image in a PDF file need not be a document image, the *Extract Document Image* module determines whether the current image is a document image or not. If the current image is not a document image, then it is ignored while document images are subjected to further processing. The extracted document image then goes through the *Pre-processing* module that does various preprocessing operations like Skew-correction, Thresholding and Binarization. This pre-processed image is then sent to the *Word Segmentation* module that segments the pre-processed image into word images. These word images are then sent to the *Feature Extraction* routine that extracts feature values from the word images. We have used features (Jawahar 2004) such as the horizontal profile, the vertical profile, and the background to ink transition. All these features are normalized so that variations due to font size are taken into account. This process is repeated for all the document images in the PDF file. This entire offline process should not interfere with the PDF readers loading and displaying contents of a PDF file, which is very fast. Keeping this in mind, we fork out the entire offline activity in order to facilitate the user to query the text content until the features are extracted for all the document images in a PDF file.

During the search phase, the input text is processed along with the language information. There are two kinds of input, the ASCII and the ITRANS (ITRANS 2001). In the ASCII mode, the English text is handled (Latin scripts) as is. While in the ITRANS input mode, the user chooses a target Indian language in which he has to search. ITRANS is a transliteration scheme wherein the user types input search text in Roman characters while the output is in the particular Indian language. After determining the input mode (ASCII or ITRANS) and language, the input text is rendered as a word image. This word image is then subjected to *Feature Extraction* process. In the *Matching* stage, the features of the rendered input word image is matched with all the features of the word images that were extracted during the load process. We use the

Dynamic Time Warping (DTW) (Rath *et al.* 2003) based matching technique to match the input word image with every other word image in the PDF file. Partial matching (Jawahar 2004) is accommodated so that word form variations are also searched and taken care of. The resultant matches are restricted according to a threshold and the matched word images are grouped according to their page numbers. The display is also in reverse video, with appropriate conversion from the image coordinate space to the display coordinate space. The load process is initiated every time a user opens a new PDF file or updates the PDF document. This search is integrated within the text search (refer Figure 1), so that the search result is transparent to the user irrespective of whether the search result was from a Text stream or Image stream.

4. Open Source Implementation and Indian Language Support

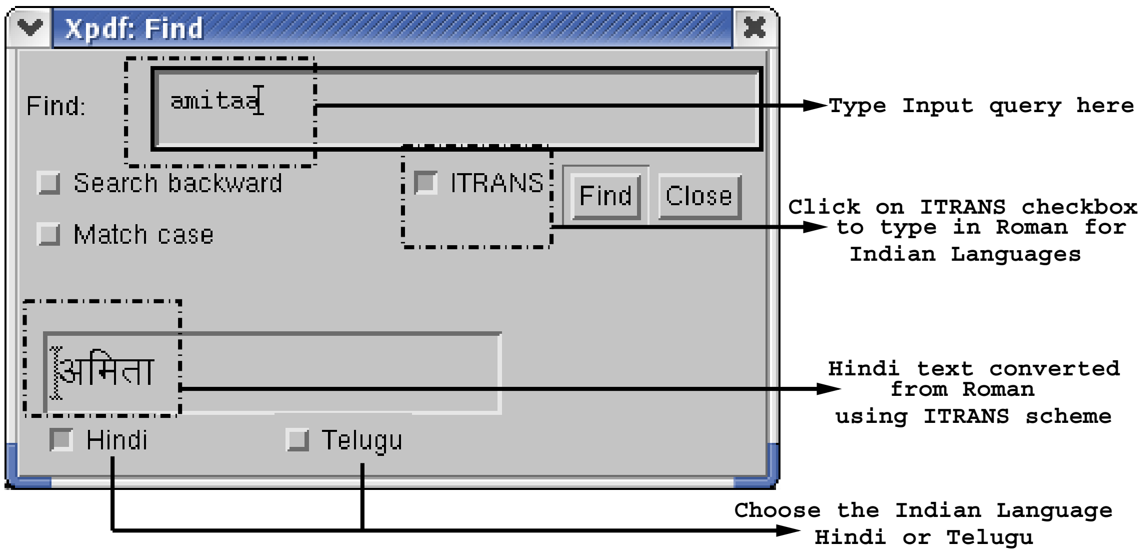

Xpdf: Find
Find:
amitaa
Search backward
Match case
ITRANS
Find
Close
अमिता
Hindi
Telugu
Type Input query here
Click on ITRANS checkbox to type in Roman for Indian Languages
Hindi text converted from Roman using ITRANS scheme
Choose the Indian Language Hindi or Telugu

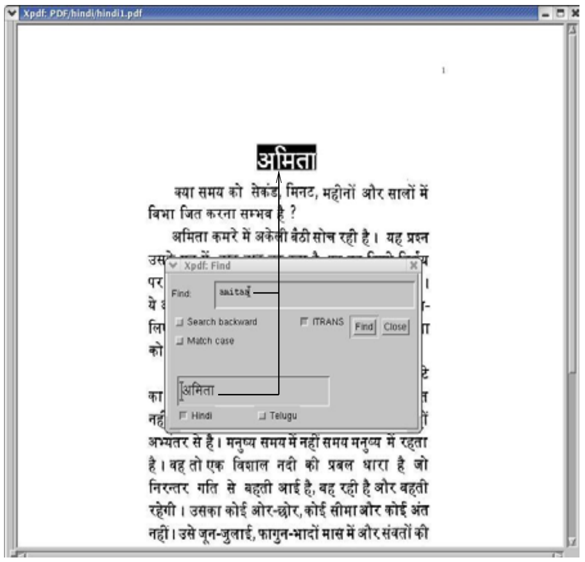
Figure 4: Screenshots of Find Dialog box in Xpdf. User can enter the query word in Roman (ITRANS), view the script and see the results in reverse video.

We verified our algorithm on an opensource PDF reader (Xpdf) (XPDF 2005). Implementation integrates the textual and image (graphics) search in a transparent manner. Xpdf is designed to be small and efficient. The Xpdf source code implementation clearly separates out the front-end (User Interface) from its core which is the back-end. This independence of the User Interface from the core is of major advantage to users who are interested to extract text and images from a PDF file without having to view it. The User Interface has been developed using *Motif*, an X Windows based user interface builder in Linux, while the core implementations concerning the PDF file has been implemented using C++. All the above operations are achieved using Xpdf by appropriately adding and modifying code snippets into the Xpdf source code. The textual search operations are handled well within the Xpdf code, while our module of word image search is integrated within the text search module of the Xpdf source code. Integration process is explained in Figure 1. Indian language scripts have complex layout.

That is why OCR for Indian languages do not claim of high accuracies similar to Latin scripts. Indian language content is often stored as images (graphics) in the PDF files. To search we need an input mechanism which can be compatible with the Roman script. To enter an Indian language text as UNICODE, ISCII or font is a cumbersome process. The alphabet set for Indian languages are typically high when compared to English. The presence of *Samyuktakshar* (compounded letter) in Indian languages makes the rendering process of the word images all the more difficult. It is very difficult to enter a search string for Indian languages following a specific font encoding for that particular language. This makes it contingent for the user to be well-versed with font encoding for every Indian language. This process is not intuitive and ITRANS (ITRANS 2001) fits in as a perfect workaround. ITRANS is a transliteration scheme wherein the user enters the text in Roman such that the scheme is common across Indian languages.

ITRANS text is then converted to UNICODE, and compatible fonts are used to render word images for the UNICODE text. Though there have been attempts for Indian language keyboard layout, known as INSCRIPT, its support at the operating system level is not satisfactory. One also needs mapping information to convert the ITRANS text to a specific language UNICODE. Word image rendering is handled by an image rendering routine, which takes a font-encoded text, its font name, its size, and its colour as input and then renders the word image.

Figure 4 shows the *Find* dialog box in Xpdf, which has been modified to show Indian language content depending on the language chosen (e.g.,: Hindi or Telugu). The user chooses the *ITRANS* check box in the *Find* dialog in order to search Indian language content. As the user keeps typing in the search *Text Box*, the corresponding text in the chosen Indian language (Hindi in the above example) will appear. As can be seen from Figure 4, the example shows the search word "amitaa" that was queried. Figure 5 (a) shows the results highlighted in the reverse video. One has to take note of the fact that the content displayed in the PDF is a document image, but not a textual (ASCII) content. The modified Xpdf code also contains facility to search in Telugu document images (Figure 5 (b)) other than Hindi. All that the user has to do is to select the Telugu check box in order to search Telugu PDF files. The converters have been written from ITRANS to a specific font. The same interface is used to search both inline text and word images. Essentially all the available and existing functionalities of Xpdf are preserved.



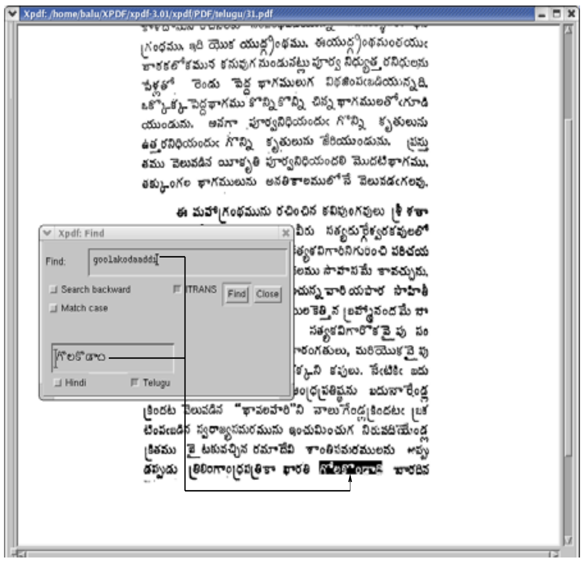
amitaa

अमिता

अमिता

क्या समय को सेकंड, मिनट, महीनों और सालों में
बिभा जित करना सम्भव है ?
अमिता कमरे में अकेली बैठी सोच रही है। यह प्रश्न

अभ्यंतर से है। मनुष्य समय में नहीं समय मनुष्य में रहता
है। वह तो एक विशाल नदी की प्रबल धारा है जो
निरन्तर गति से बहती आई है, बह रही है और बहती
रहेगी। उसका कोई ओर-छोर, कोई सीमा और कोई अंत
नहीं। उसे जून-जुलाई, फागुन-भादों मास में और संवतों की



goolakodaadda

(a) (b)

Figure 5: Screenshots of Word Image Search results in Xpdf in Hindi and Telugu PDF files.

5. Conclusion

Here we have presented a word spotting based PDF search for document word images using an existing opensource PDF viewer, Xpdf. We have effectively handled the issues arising out of Indian languages and have supported all the functionalities well within the existing source code of Xpdf and have demonstrated it for PDF files containing Hindi and Telugu document images. This solution can help the readers of Digital library by giving access to the textual content stored in the graphics stream.

Acknowledgements

We wish to thank the Ministry of Information and Technology, Govt. of India for its support in this initiative.

References

Lesk M. 1997 , Practical Digital Libraries: **Books, Bytes, and Bucks**. Morgan Kaufmann.

Balasubramanian A., Million Meshesha, and Jawahar C.V. 2006. "**Retrieval from document image collections**," *in Proceedings of Seventh IAPR Workshop on Document Analysis Systems , 2006 (LNCS 3872)*, Nelson, NewZealand, pp. 1-12.

Doermann D., 1998. "**The Indexing and Retrieval of Document Images: A Survey,**" *Computer Vision and Image Understanding: CVIU*, vol. 70, no. 3, pp. 287-298.

Jawahar C.V, Million Mesha, and Balasubramanian A. 2004. "**Searching in document images**," *in Proceedings of the Indian Conference on Vision, Graphics and Image Processing*, pp. 622--627.

Pramod Sankar K, Vamshi Ambati, Lakshmi Hari, and Jawahar C.V. 2006, "**Digitizing a million books: Challenges for document analysis**," *in Proceedings of Seventh IAPR Workshop on Document Analysis Systems* , 2006 (LNCS 3872), Nelson, NewZealand, pp. 425-436.

Rath T. and Manmatha R. 2003. "**Features for word spotting in historical manuscripts**," *in International Conference on Document Analysis and Recognition*, pp. 218-222.

Rath T. and Manmatha R. 2003. "**Word Image Matching Using Dynamic Time Warping**," in *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 521--527.

Rath T., Manmatha R, and Victor Lavrenko. 2004. "**A search engine for historical manuscript images**," *in Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 369-376.

Santanu Chaudhury, Geetika Sethi, Anand Vyas, and Gaurav Harit. 2003. "**Devising interactive access techniques for Indian language document images,**" *in Proceedings of International Conference on Document Analysis and Recognition*, pp. 885-889.

Sakoe H. and Chiba S. 1980. "**Dynamic Programming Optimization for Spoken Word Recognition**," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. 26, pp. 623-625.

Vamshi Ambati, Balakrishnan N, Raj Reddy, Lakshmi Pratha, Jawahar C.V. 2006. "**The Digital Library of India Project: Process, Policies and Architecture**", i*n the Proceedings of 2nd International Conference on Digital Libraries(ICDL).*

Adobe Systems Inc., 2003. PDF Reference, Fourth Edition, *http://partners.adobe.com/public/developer/pdf/index_reference.html*

ITRANS 2001- Indian Language Transliteration Package, *http://www.aczoom.com/itrans/*

XPDF 2005 - an open source PDF viewer, *http://www.foolabs.com/xpdf/*

Universal Library, *http://www.ulib.org*