A Document Page Layout Analysis Technique:

Enhanced Scientific Document Understanding Through the Combination of Salient Features into a Support Vector Machine Trained for Equation Detection

in Partial Fulfillment of the Requirements for the Degree
M.S. in Computer Engineering

Presented to the Faculty of ECE
of Virginia Tech by

Jake Bruce

in July 2013

Supervisor: Prof. Lynn Abbott, Ph.D.
Co-Supervisors: Prof. Jason Xuan, Ph.D. and Prof. Jules White, Ph.D.

Introductory remarks

Version <number>, published in <month> <year>

The importance of the glossary. It is strongly recommended to read the glossary in full before starting with the first chapter.

Hints for screen use. This work is optimized for both screen and paper use. It is recommended to use the digital version where applicable. It is a file in Portable Document Format (PDF) with hyperlinks for convenient navigation. All hyperlinks are marked with link flags (►). Hyperlinks in diagrams might be marked with colored borders instead.

Navigation aid for bibliographic references. Bibliographic references to works which are publicly available as PDF files mention the logical page number and an offset (if non-zero) to calculate the physical page number. For example, to look up [Example :a01, p. 100_{-80}] jump to physical page 20 in your PDF viewer.

Declaration

Hereby I declare that I wrote this thesis myself with the help of no more than the mentioned literature and auxiliary means.

Up to now, this thesis was not published or presented to another examinations office in the same or similar shape.

<place>, <date>

place and date

signature (<first name> <last name>)

Abstract

<first paragraph title>. <The abstract of the diploma thesis is meta information resp. “management information”. Therefore it should cover at most two pages, sum up the thesis' essentials and contain the idea behind the thesis.>

Acknowledgments

<Here is one page to thank everybody who helped and supported the author to write this thesis.>

Table of contents

Table of contents

1 Introduction

Basically, our goal is to organize the world's information and to make it universally accessible and useful.

Larry Page – Co-founder of Google

1.1 Enhancing Information Accessibility through Document Analysis and Recognition

Never, since the invention of the printing press, has society seen such a radical change in its means of information distribution. Armed with powerful search engines roaming the vast expanse of the World Wide Web, nearly everyone in the world has, at their very fingertips, access to archives full of information. This enhanced information accessibility is having profound implications for society and could lead to a fruitful age of enlightenment.

The global effects of high speed Internet access are seen daily as hundreds of millions browse for information/multimedia, look up map directions, interact through email/social networks/video games, shop remotely, video chat, etc. Corporations like Google, Microsoft, Facebook, eBay, and Amazon continue building and extending the capacity of their server farms as the growth of user demand shows no signs of slowing down. By mid-2012, it was reported that nearly an eighth of the world's population was on the popular social networking site, Facebook [1]. As such figures continue to grow, studies are showing that technology is even affecting the manner in which we think and behave at the most fundamental levels. Whether or not the long-term effects of this relatively nascent medium of interaction prove to be largely positive or negative remains to be seen. One remaining certainty, however, is that continuing innovation is, for better or for worse, altering the manner in which we live out our daily lives.

It was Benjamin Franklin who once said that “genius without education is like silver in the mine.” One would be hard-pressed in arguing that, throughout history, all people have been able to realize their full potential to succeed and make a difference in the world. If that were true, many would argue that our knowledge would, by now, have long since surpassed its current state. In fact it was just under five hundred years ago, that Europeans were finally emerging from an age of intellectual darkness which had lasted for roughly a millennium. If we look back to the spread of knowledge throughout written history, starting from the earliest true writing systems developed

in ancient Egypt/Mesopotamia circa 3000 BC to the origins of philosophy, math, science, and theater in ancient Greece, all the way to the birth of the “modern era” which culminated itself in the scientific revolution of the sixteenth century AD, we notice a general trend of small bursts of knowledge spreading repeatedly, each time with greater strength than before, each one improving upon on its predecessor. Sir Isaac Newton exemplified this trait of humanity with his statement that “if I have seen further, it is by standing on the shoulders of giants.”

Although much of what defines us from a cultural perspective may indeed be passed from generation to generation through word of mouth, our tremendous advancements in math, science, art, and literature since the dawn of the modern era can be largely attributed to Johannes Gutenberg's invention of the printing press, which made mass distribution of books possible in Late-Medieval Europe. Prior to this key event in history, the stage was set in Europe for an age of scientific inquiry and revelation when the religious leader, Thomas Aquinas, embraced the separation between the purely theological and purely scientific schools of thought. Also of vital importance was the translation and recurrence of ancient Greek writings which had been studied and further developed by Arabic scholars. The first universities built in Medieval Europe were initially centered around classical Greek and religious studies and helped to lead Europe out of its age of darkness. This collaborative environment of scholastic endeavor helped set the framework for an age of enlightenment which would move humanity a step forward. Archaic ideas such as bloodletting were soon supplanted by discoveries leading to modern medicine and the commonly held geocentric model of our earth was replaced by a heliocentric one. Major breakthroughs were made in every field to foster the spread of knowledge which took society to where it is today. Without this ideal of scientific thinking combined with the means to distribute information, society would have never seen such tremendous improvements.

Moving forward to the present day, society has recently made technological breakthroughs which make the world's knowledge and information more accessible than ever before. In fact, many have suggested that the widely used search engine, Google, will go down in history as rivaling in importance with Gutenberg's printing press. It was only about a decade and a half ago that two Stanford Ph.D. students decided that they would like to take a shot at downloading and categorizing the entire internet. These two graduate students are of course the founders of Google [2], a now successful multinational corporation which, during the late nineties, left its search

engine competitors far behind. Google is unique in that its employees facilitate a diverse range of interesting projects ranging from cataloging the human genome, building autonomous vehicles, developing smart homes of the future, to developing augmented reality eye glasses, among many others. It is, however, in Google's core mission of finding ways to make the world's information “more universally accessible and useful,” that the company has had its greatest impact on the world at large. It was in keeping with this mission that, in 2005, in collaboration with HP Labs and the Information Science and Research Institute at UNLV, Google revived and open sourced an optical character recognition engine that had been developed as a Ph.D. project for HP Labs between 1985 to 1995. Although optical character recognition (OCR), the autonomous conversion of printed documents into digital formats, is a very mature area of research [3], development in this area continues in order to increase recognition support for the broad spectrum of languages, formats, and subject matter of printed documents. HP's OCR engine, named “Tesseract,” had proven itself as one of the industry's leading engines during UNLV's Fourth Annual Test of OCR Accuracy [4]. Eventually, however, HP subsequently went out of the OCR business, leaving the software to basically collect dust for about a decade.

Meanwhile, by around 2004, Google had begun its Google Books Initiative [5], a large-scale library digitization project. This initiative began with the lofty goal of digitizing all of the world's printed documents such that they may be indexed and searched online. By around 2005, Google hired Ray Smith, the former lead developer of Tesseract, to return to his long-abandoned, yet ground-breaking, Ph.D. work and also brought Tesseract into the open source domain. In so doing, Google helped to spur further research interest into efficient and accurate document recognition[1]. In the roughly eight years since the project was revived, support has been added for recognition of over fifty languages. Advanced page layout analysis techniques have been implemented in order to detect various types of documents ranging from novels, magazines, newspapers, images, textbooks, sheet music, etc. Language and script detection modules have also been implemented in order to autonomously determine what processing should be carried out for any given world document [6]. If Google's endeavor is successful, then the resulting implications to society will be extraordinary, possibly similar to the impact that Arabic scholars had on Europe when sharing and translating ancient Greek literature. If Google is successful in the autonomous

[1] The term, recognition, is herein used to describe a machine's extraction of a document's contents. This requires both the document page layout analysis as well as algorithms which subsequently convert the page layout contents into a machine-understandable form. The field of document layout analysis is further discussed in Section 1.2.

digitization and recognition of any printed document regardless of its origin, then it will not be long before information from all of the world's documents become instantly accessible in every language and to everyone around the world. Such a development would certainly speed up the world's already significant progress toward an era of far greater enlightenment and wisdom than has yet been seen.

The autonomous recognition of all printed documents would not only expedite the global advancement of knowledge and wisdom, but would also have tremendous implications toward every individual in society. Such a breakthrough would be especially significant toward the endeavor of Assistive Technology. With many devices being developed and studies being carried out on ways to enhance human computer interaction (HCI) for visually or physically handicapped individuals, digital access to all printed documents could make finding information, not only more convenient, but also possible for many who would not otherwise have access. Global autonomous document recognition could also help open the doors toward breaking down language barriers in information accessibility.

As research and development continues to enhance the accurate translation of discourse between various languages [7], the successful recognition of printed documents could eventually allow them to be machine-translated according to the language preference of a given user. With instant access to all of the world's information, regardless of its language or origin, at one's disposal, collaboration and learning among individuals across the world will be significantly enhanced. All people in the world regardless of their language preference, geographical location, and physical ability will have access to the world's stores of knowledge, and the opportunity to have a profound impact on society through the medium of the World Wide Web. Enhanced document analysis and recognition capabilities will make a significant contribution toward this end. The following section will discuss the background as well as some of the fundamental problems faced in the fields of document analysis and recognition.

1.2 Introduction to OCR and Document Analysis: A Brief History

From Herbert Shantz's *History of Optical Character Recognition (OCR)* [3], it is clear that the OCR of printed documents has been studied extensively over the last century. In one of the earliest OCR patents [8] (Figure 1), a mechanical apparatus was used to measure the incidence of light reflected back from a printed character when illuminated through a set of character templates. A character detection would occur when the light emitted from the template overlapped the character (assumed to be in dark print) sufficiently to prevent light from being reflected upon the medium. Despite

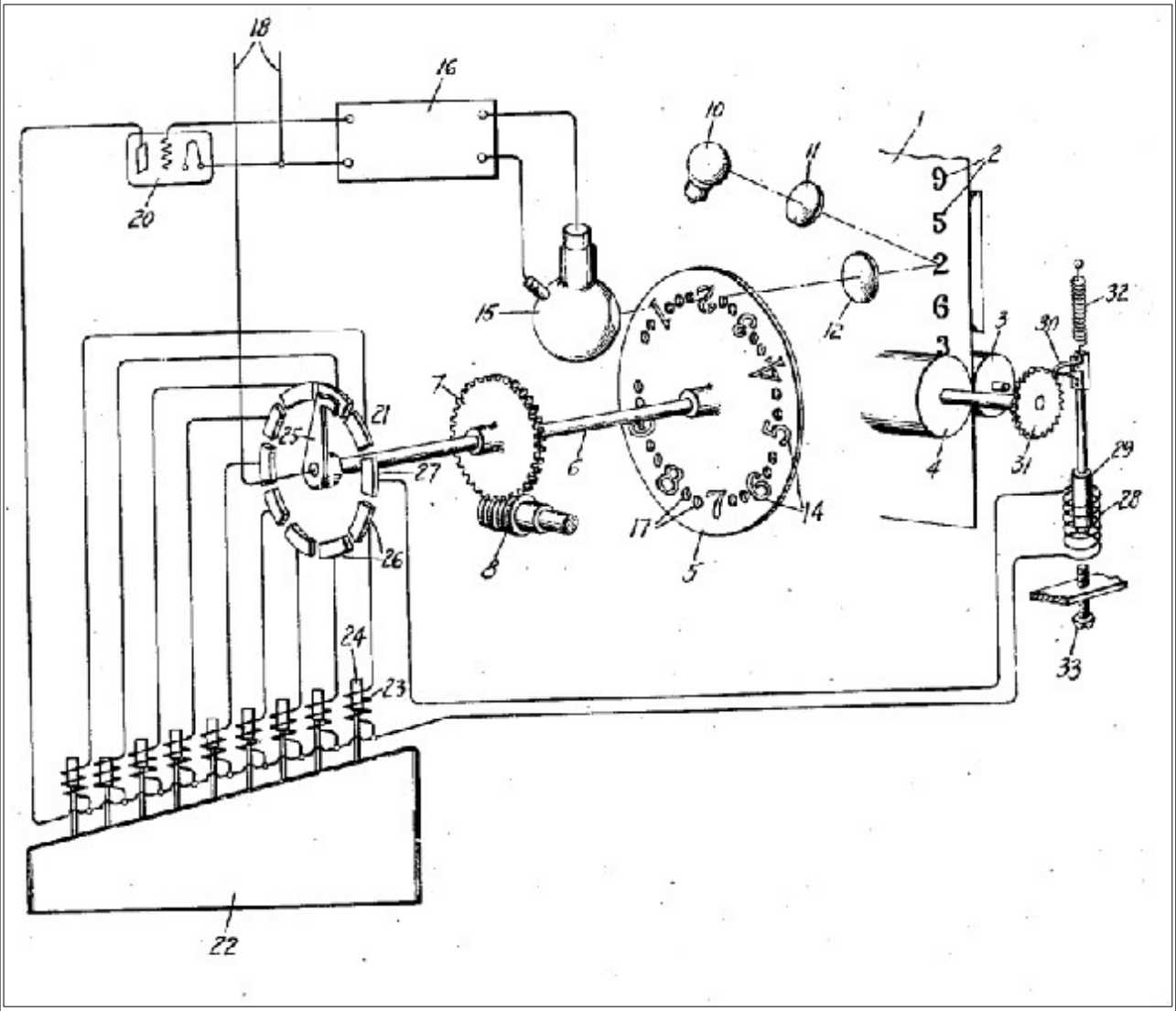


Figure 1: An illustration taken from the 1933 patent from Paul W. Handel, a former employee of General Electric, entitled "Statistical Machine" [8]. This is one of the earliest OCR devices ever invented.

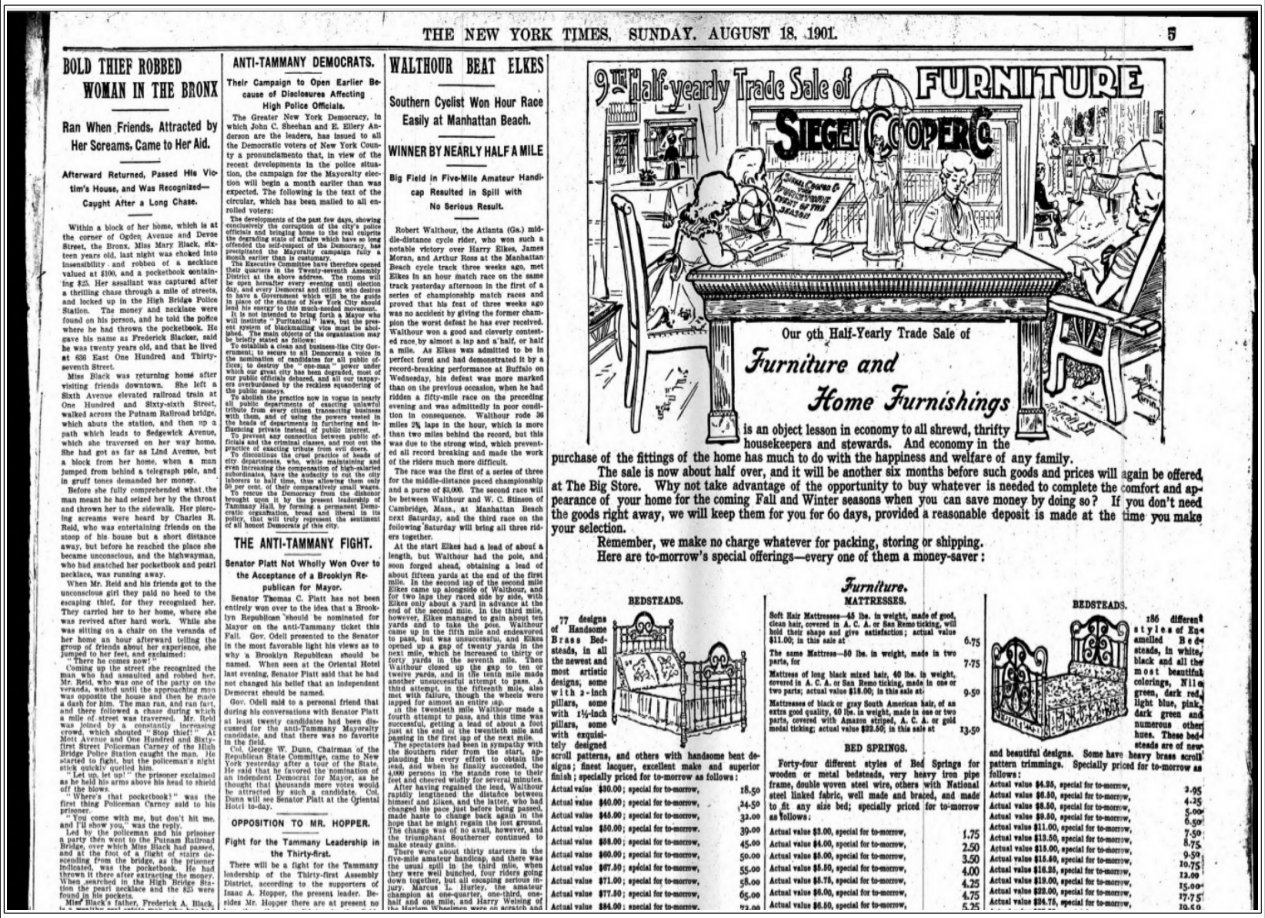
requiring a significant amount of human intervention to ensure proper alignment and being largely inefficient at best, the fundamental ideas which motivated this early initiative are seen repeatedly throughout the century, and even now, albeit on a much larger scale.

Although some of the first commercial OCR systems were released during the 1950's, their applicability was limited in that, by and large, they were only capable of handling a single font type with very strict rules on character spacing. It was not until the mid-late 1970's, with the invention of both the charge-coupled device (CCD) flatbed scanner and the “Kurzweil Reading Machine” [9] that it became possible for a computer to read a variety of documents with reasonable accuracy. Although the training process for a particular font would take several hours and multi-column page layouts or images had to be specified by the user manually, Kurzweil's software showed significant improvement over the state-of-the-art technologies of the time.

In the 1980's, a company called Calera Recognition Systems [10] introduced an omnifont system that could read pages containing a mixture of fonts while also locating pictures and columns of text without any user intervention or extra training. The progress of the state-of-the-art in document recognition will be further discussed in the Chapter 2 Literature Review. More recent commercial OCR systems such as ABBYY FineReader [11], OmniPage Professional [12], and Readiris [13], are all quite accurate, not only in recognizing individual words or characters, but also in understanding and reproducing document layout structure. A magazine or newspaper page may, for instance, contain an intricate heading structure followed by multiple columns of text, pull-out quotes, in-set images, and/or graphs as demonstrated in the historical New York Times article shown in Figure 2 [14].

In order to understand and recognize content of such a document, it is essential to first carry out document layout analysis techniques which will determine how the document is partitioned. The text will be recognized with an understanding of where the columns of text are, which portions of text indicate headings or quotes, and which segments correspond to images, tables, captions, etc. If the text is not partitioned appropriately prior to recognition then the textual output will become unpredictable. With columns, paragraphs, or other structures merged together incorrectly, the text will lose much of its intended meaning and become far less readable to the human eye. For these reasons, sophisticated page layout analysis algorithms are of the utmost importance, not only for document recognition accuracy, but also in ensuring that the generated output is formatted correctly.

1 Introduction



THE NEW YORK TIMES, SUNDAY, AUGUST 18, 1901. 5

BOLD THIEF ROBBED WOMAN IN THE BRONX

Ran When Friends, Attracted by Her Screams, Came to Her Aid.

Afterward Returned, Passed His Victim's House, and Was Recognized—Caught After a Long Chase.

Within a block of her home, which is at the corner of Ogden Avenue and Devoe Street, the Bronx, Miss Mary Black, sixteen years old, last night was choked into insensibility and robbed of a necklace valued at $100, and a pocketbook containing $25. Her assailant was captured after a thrilling chase through a mile of streets, and locked up in the High Bridge Police Station. The money and necklace were found on his person, and he told the police where he had thrown the pocketbook. He gave his name as Frederick Slacker, and he was twenty years old, and that he lived at 626 East One Hundred and Thirty-seventh Street.

Miss Black was returning home after visiting friends downtown. She left a Sixth Avenue elevated railroad train at One Hundred and Sixty-sixth Street, walked across the Putnam Railroad bridge, which abuts the station, and then up a path which leads to Sedgewick Avenue, which she traversed on her way home. She had gone as far as Lind Avenue, but a block from her home, when a man jumped from behind a telegraph pole, and in gruff tones demanded her money.

Before she fully comprehended what the man meant he had seized her by the throat and thrown her to the sidewalk. Her piercing screams were heard by Charles B. Reid, who was entertaining friends on the stoop of his house but a short distance away, but before he reached the place she became unconscious, and the highwayman, who had snatched her pocketbook and pearl necklace, was running away.

When Mr. Reid and his friends got to the unconscious girl they paid no heed to the escaping thief, for they recognized her. They carried her to her home, where she was revived after hard work. While she was sitting on a chair on the veranda of her home an hour afterward telling the group of friends about her experience, she jumped to her feet, and exclaimed:

"There he comes now!"

Coming up the street she recognized the man who had assaulted and robbed her. Mr. Reid, who was one of the party on the veranda, waited until the approaching man was opposite the house and then he made a dash for him. The man ran, and ran fast, and there followed a chase during which a mile of street was traversed. Mr. Reid was joined by a constantly increasing crowd, which shouted "Stop thief!" At Mott Avenue and One Hundred and Sixty-first Street Policeman Carney of the High Bridge Police Station caught the man. He started to fight, but the policeman's night stick quickly quelled him.

"Let go, let go!" the prisoner exclaimed as he held his arms above his head to shield off the blows.

"Where's that pocketbook?" was the first thing Policeman Carney said to his prisoner.

"You come with me, but don't hit me, and I'll show you," was the reply.

Led by the policeman and his prisoner a party then went to the Putnam Railroad Bridge, over which Miss Black had passed, and at the foot of a flight of stairs descending from the bridge, as the prisoner indicated, was the pocketbook. He had thrown it there after extracting the money. When searched in the High Bridge Station the pearl necklace and the $25 were found in his pockets.

Miss Black's father, Frederick A. Black,

ANTI-TAMMANY DEMOCRATS.

Their Campaign to Open Earlier Because of Disclosures Affecting High Police Officials.

The Greater New York Democracy, in which John C. Sheehan and E. Ellery Anderson are the leaders, has issued to all the Democratic voters of New York County a pronunciamento that, in view of the recent developments in the police situation, the campaign for the Mayoralty election will begin a month earlier than was expected. The following is the text of the circular, which has been mailed to all enrolled voters:

The developments of the past few days, showing conclusively the corruption of the city's police officials and bringing home to the real culprits the degrading state of affairs which have so long offended the self-respect of the Democracy, has precipitated the Mayoralty campaign fully a month earlier than was expected.

The Executive Committee has therefore opened their quarters in the Twenty-seventh Assembly District at the above address. The rooms will be open hereafter every evening until election day, and every Democrat and citizen who desires to have a Government which will be the gainsay in place of the shame of New York City should lend his energy to this much-needed movement.

It is not intended to bring forth a Mayor who will enforce sumptuary laws, but the present system of partisanship and vice must be abolished. The main object of the organization may be briefly stated as follows:

To establish a clean and business-like City Government, to secure to all Democrats a voice in the nomination of candidates for all public offices, to destroy the "one-man" power under which our great city has been degraded, most of our public officials debased, and all our taxpayers overburdened by the reckless squandering of the public moneys.

To abolish the practice now in vogue in nearly all public departments of exacting unlawful tribute from every citizen transacting business with them, and of using the powers vested in the heads of departments in furthering and financing private instead of public interest.

To prevent any connection between public officials and the criminal classes, and root out the practice of exacting tribute from evil doers.

To discontinue the cruel practice of heads of city departments, who, while maintaining and even increasing the compensation of high-salaried subordinates, reduce the wages of the poor laborers to half time, thus allowing them only 50 per cent. of their comparatively small wages.

To rescue the Democracy from the dishonor brought upon it by the present leadership of Tammany Hall, by forming a permanent Democratic organization, broad and liberal in its policy, that will truly represent the sentiment of all honest Democrats of this city.

THE ANTI-TAMMANY FIGHT.

Senator Platt Not Wholly Won Over to the Acceptance of a Brooklyn Republican for Mayor.

Senator Thomas C. Platt has not been entirely won over to the idea that a Brooklyn Republican should be nominated for Mayor on the anti-Tammany ticket this Fall. Gov. Odell presented to the Senator in the most favorable light his views as to why a Brooklyn Republican should be named. When seen at the Oriental Hotel last evening, Senator Platt said that he had not changed his belief that an independent Democrat should be named.

Gov. Odell said to a personal friend that during his conversations with Senator Platt at least twenty candidates had been discussed for the anti-Tammany Mayoralty candidate, and that there was no favorite in the field.

Col. George W. Dunn, Chairman of the Republican State Committee, came to New York yesterday after a tour of the State. He said that he favored the nomination of an independent Democrat for Mayor, as he thought that thousands more votes would be attracted by such a candidate. Col. Dunn will see Senator Platt at the Oriental Hotel to-day.

OPPOSITION TO MR. HOPPER.

Fight for the Tammany Leadership in the Thirty-first.

There will be a fight for the Tammany leadership of the Thirty-first Assembly District, according to the supporters of Isaac A. Hopper, the present leader. Besides Mr. Hopper there are at present no

WALTHOUR BEAT ELKES

Southern Cyclist Won Hour Race Easily at Manhattan Beach.

WINNER BY NEARLY HALF A MILE

Big Field in Five-Mile Amateur Handicap Resulted in Spill with No Serious Result.

Robert Walthour, the Atlanta (Ga.) middle-distance cycle rider, who won such a notable victory over Harry Elkes, James Moran, and Arthur Rose at the Manhattan Beach cycle track three weeks ago, met Elkes in an hour match race on the same track yesterday afternoon in the first of a series of championship match races and proved that his feat of three weeks ago was no accident by giving the former champion the worst defeat he has ever received. Walthour won a good and cleverly contested race by almost a lap and a half, or half a mile. As Elkes was admitted to be in perfect form and had demonstrated it by a record-breaking performance at Buffalo on Wednesday, his defeat was more marked than on the previous occasion, when he had ridden a fifty-mile race on the preceding evening and was admittedly in poor condition in consequence. Walthour rode 36 miles 264 laps in the hour, which is more than two miles behind the record, but this was due to the strong wind, which prevented all record-breaking and made the work of the riders much more difficult.

The race was the first of a series of three for the middle-distance paced championship and a purse of $5,000. The second race will be between Walthour and W. C. Stinson of Cambridge, Mass., at Manhattan Beach next Saturday, and the third race on the following Saturday will bring all three riders together.

At the start Elkes had a lead of about a length, but Walthour had the pole, and soon forged ahead, obtaining a lead of about fifteen yards at the end of the first mile. In the second lap of the second mile Elkes came up alongside of Walthour, and for two laps they raced side by side, with Elkes only about a yard in advance at the end of the second mile. In the third mile, however, Elkes managed to gain about ten yards and to take the pole. Walthour came up in the fifth mile and endeavored to pass, but was unsuccessful, and Elkes opened up a gap of twenty yards in the next mile, which he increased to thirty or forty yards in the seventh mile. Then Walthour closed up the gap to ten or twelve yards, and in the tenth mile made another unsuccessful attempt to pass. A third attempt in the fifteenth mile, also met with failure, though the wheels were clipped for almost an entire lap.

In the twentieth mile Walthour made a fourth attempt to pass, and this time was successful, getting a lead of about a foot just at the end of the twentieth mile and passing in the first lap of the next mile.

The spectators had been in sympathy with the Southern rider from the start, applauding his every effort to obtain the lead, and when he finally succeeded, the 4,000 persons in the stands rose to their feet and cheered wildly for several minutes.

After having regained the lead, Walthour rapidly lengthened the distance between himself and Elkes, and the latter, who had changed his pace just before being passed, made haste to change back again in the hope that he might regain the lost ground. The change was of no avail, however, and the triumphant Southerner continued to make steady gains.

There were about thirty starters in the five-mile amateur handicap, and there was the usual spill in the third mile, when there were well bunched four riders going down together, but all escaping serious injury. Marcus L. Hurley, the amateur champion at one-quarter, one-third, one-half and one mile, and Harry Weinig of the Harlem Wheelmen were on scratch and

Figure 2: This excerpt from a 1901 New York Times article [14] was optically recognized by ABBY Fine Reader 8. The "New York Times" heading at the very top, the "Furniture and Home Furnishings" label embedded in the illustration, and the layout of the three columns at the bottom right were all incorrectly recognized by the commercial system.

Although most publishers keep digital copies of their more recent documents, there is also great demand for older documents which, unless they are digitized, will largely become forgotten by society. This would be unfortunate in that it is often surprising how pertinent older information and ideas can be. For companies such as Google who would like to make the world's information more readily available and accessible as well as to the Assistive Technology community, this is of the utmost importance. For this reason, a standard OCR output format called hOCR, which embeds OCR output within well-defined and widely available HTML and CSS structures has been put into place [15]. In order to ensure the quality of textual output generated by OCR for the wide variety of possible document layout structures, sophisticated document layout techniques are critical.

1.3 Google Books Initiative

There are various languages, dialects, and page layout formats for which Google's Tesseract software is being developed. Among them are mathematical equations, tables, graphs, and other figures which can be found in any standard science or math text book. While Smith's original work was optimized solely for the recognition of English newspaper formats, Google's continued efforts are aimed at recognizing page formats from a much broader scope [5]. Much of Google's ideas regarding document recognition are essentially in their infancy, and have a long way to go before being fully realized. Although an experimental equation detector has been added to the Tesseract software, its results, although showing significant promise, have been tested to have fairly limited accuracy. A table detector implemented by Google has also been tested on some sample images [16] (Figure 3) to show that, it too, could use significant improvement (Figure 4). Notice that, in the left-most table in Figure 4, the software failed to indicate the years as either belonging to the table or the normal text. They were simply disregarded. Also, the software was unable to determine where exactly the table boundaries are (which should be labeled green). In the right-most table, notice that although a better job was done, while the bottom portion of the text consists of footnotes, it is therein incorrectly labeled as part of the table. Also, the second line of all column labels are not recognized as part of the table when they clearly should be.

The problem of efficiently and accurately detecting equations, tables, graphs, and other figures for the broad spectrum of possible document types is certainly no easy one to solve. Although from a human's perspective, this problem may seem trivial, programming a machine to sum up a document with the same accuracy as the human eye proves to be a daunting task, as will be further discussed in the literature review chapter of this paper. As the inventors of Google continue to work toward their dream of creating an online “Library of Alexandria,” there is significant progress to be made before such a large-scale endeavor can be fully realized. The Google Book Search initiative has opened up many avenues for future research in document understanding and recognition, of which, this project is certainly one of the many to come.

1 Introduction

	Total acreage under corn crops.	Total acreage under wheat only.
1870	7,570,279	3,247,973
1875	7,528,543	3,128,547
1880	6,993,699	2,745,733
1885	6,569,105	2,349,305
1890	6,281,494	2,255,694
1895	5,718,997	1,339,806
1898	5,731,463	1,987,385

Almost the entire reduction in the acreage under corn crops, it will be seen, must be due to the reduction of the acreage under wheat, which is a great and conspicuous fact, implying remarkable changes in the economic and political condition of the country. Similarly, there has been an increase of the acreage

CATTLE.

	Millions in or about 1870.	Millions in or about 1898.
United Kingdom	9·2	11·1
France	11·3	13·4
Germany	15·8	18·5 [4]
Austria	7·4	8·6 [5]
Hungary	5·3	6·7 [6]
Italy	3·5 [1]	5·0 [5]
Belgium	1·2 [2]	1·4 [6]
Holland	1·4	1·6
Denmark	1·2	1·7
Sweden	2·0	2·6
Norway	1·0 [3]	1·0 [7]
Russia in Europe (excluding Poland)	21·4	32·9 [7]
Total	80·7	104·5

[1] 1875-6. [2] 1866. [3] 1865. [4] 1897. [5] 1890. [6] 1895. [7] 1900.

Figure 3: The above text includes excerpts from two different pages taken from a scan of Sir Robert Griffin's Stastics textbook (circa 1913) [12]. On the left is a table followed by a paragraph of text, while on the right is a larger table. These images were extracted from a PDF which was made available online by Google.

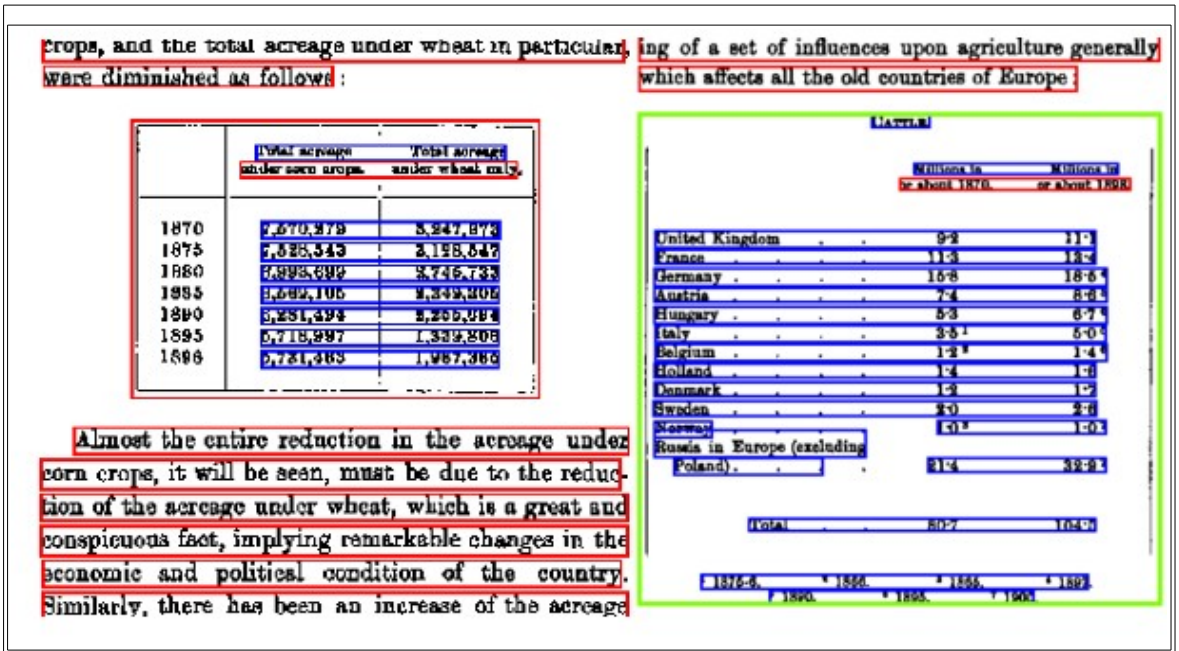


Figure 4: Above is the text from Figure 3 after having been labeled by Tesseract's table detection software. The text within the blue rectangles was identified as belonging to a table while the text within red rectangles was not. The green rectangle should encompass the entire table figure. As can be seen there are both false negatives and false positives.

1.4 Contributions of this Thesis

This thesis introduces a novel approach to detecting mathematical expressions during the document layout analysis stage of OCR. The focus of this thesis is toward enhancing the OCR quality of printed documents which may contain mathematical formulas. The motivation for mathematical expression detection is illustrated by Figure 5. From Figure 5, it is clear that, when presented with mathematical expressions as input, Google's OCR System, Tesseract, will fail. With reliable expression detection, it becomes possible to prevent this mangled output from occurring, and also allows existing equation recognition algorithms, which have been extensively studied in the literature, to be better utilized.

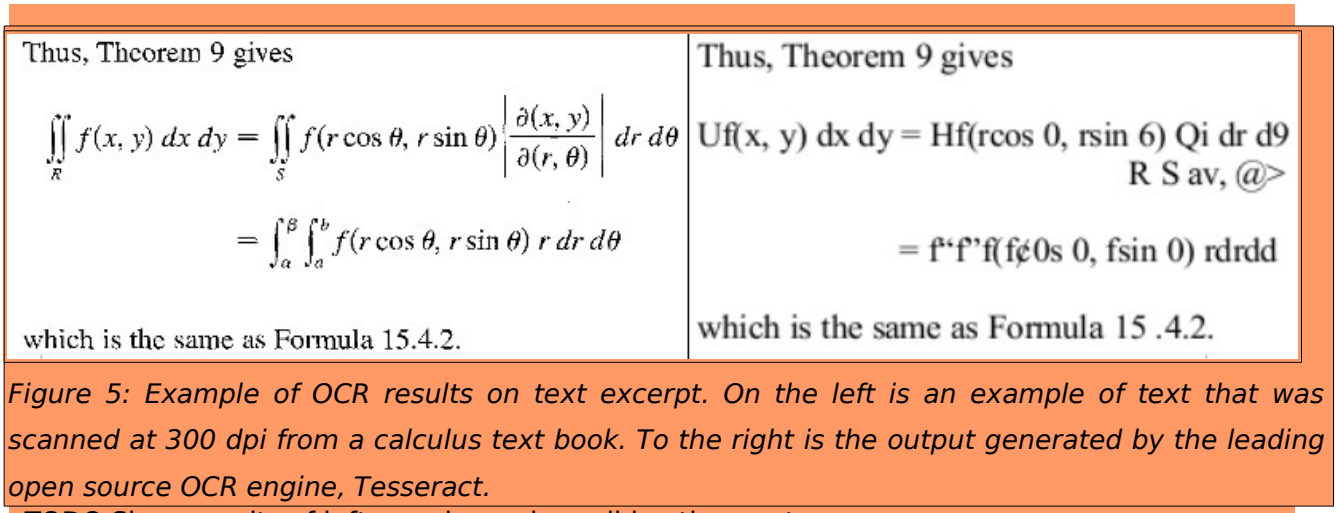


Figure 5: Example of OCR results on text excerpt. On the left is an example of text that was scanned at 300 dpi from a calculus text book. To the right is the output generated by the leading open source OCR engine, Tesseract.

TODO Show results of infty reader and possibly other systems

By utilizing and interfacing with the existing data structures and algorithms present within Google's open source OCR engine, Tesseract, much of the more well-studied areas of OCR / document analysis research are surpassed so that a study of the relevant problem of equation detection can be explored in much greater detail than would be possible otherwise. As the Tesseract software, much like commercial state-of-the-art systems, is capable of partitioning a document into columns, paragraphs, headings, etc., the software implemented in this work searches Tesseract's resulting partitions in order to detect regions of interest. Greater document understanding is accomplished through recognition of a variety of relevant features, many of which have yet to have been explored in existing research. Relevant features are subsequently combined with a binary support vector machine (SVM) classifier.

1 Introduction

The feature recognition and classification steps in the proposed system are carried out in two separate passes: the first of which detects areas of interest at the individual character level while the second uses results from the first to combine the equation regions into their full partitions. The contributions of this work are summarized in the list below. All of the subjects for which the author could not find an in-depth previous study involving equation detection are marked with an asterisk (*).

- (*) Generation of an extensive ground-truth training set of equation test data, taken from scientific text books and articles. These publications are all in the public domain and thus will be freely available online for future research endeavors.
- Recognition of the following combination of features on equation detection. Although many of these features have indeed been studied to some extent, they have yet to all be used within a single framework as of this date to the author's knowledge.
  - (*) Measurement of the number of horizontally adjacent characters to the right of a given character within some vertical threshold (see chapter x.x).
  - Sub-script/super-script recognition
  - (*) Running a Tesseract classifier trained using Infty Reader's database of mathematical expressions and comparing the result to normal language output to detect math expressions (possibly subsequent recognition?) (see chapter x.x)
  - Use of n-grams to locate expressions embedded within text (see chapter x.x)
  - Testing whether or not a character's language[2] classification result falls under a category of potential math symbols such as <,>,_,+,-,/,%, etc.
  - Vertical distance to nearest character neighbor above and below (within a horizontal threshold), horizontal distance to nearest neighbor left and right (within some vertical threshold).
  - Ratio of horizontal and vertical distance from nearest neighbor to the left and above to nearest neighbor to the right and below respectively. ????
  - Height of characters as compared to average character height within a page
  - Detection of Italicized text
  - Horizontal bar detection

[2] Here the language classification result indicates the result of a classifier that was trained for a particular language. Although in the context of this work English is all that is tested, testing of existing techniques in various languages is encouraged for future work.

    - Features extracted from PDF if available
    - Detection of indentation or centering of text
    - Measurement of character vertical distance from a row of text's baseline
- (*) Use of a Support Vector Machine (SVM) classifier for equation detection
- Thorough evaluation of the proposed system's results, includes a comparison with Google's system.

1.5 Organization of Thesis

The work to be discussed in this thesis is aimed toward moving the world a step closer to realizing some of the lofty goals set by Google's engineers and scientists. Chapter 2 presents a review of existing document analysis techniques with extra emphasis on those involving mathematical/scientific documents. Although there are a wide variety of problems which need to be tackled in the area of document recognition, the primary focus is on enhancing equation detection accuracy through the use of feature recognition and a support vector machine (SVM) classifier. The remainder of this thesis is organized as follows: Chapter 3 consists of a theory section discussing the image processing and classification techniques employed as well as software optimization techniques utilized by Google's open source OCR engine, Tesseract. Chapter 4, the method section, discusses the ground truth generation procedure, feature recognition algorithms, classification technique, and result evaluation. Chapter 5, the results section, will involve a discussion of all results and their significance. Chapter 6, the conclusion, summarizes important points and discusses recommendations for future work.

2 Literature Review

“We are like dwarfs sitting on the shoulders of giants. We see more, and things that are more distant, than they did, not because our sight is superior or because we are taller than they, but because they raise us up, and by their great stature add to ours.”

John Salisbury

2.1 The Beginnings of OCR

2.1.1 Fixed-font

Over the past century, Assistive Technology has been a primary incentive for OCR research and development. While machine understanding was initially the most commercially viable domain for OCR, reading devices for the blind have been commonly implemented over the years. In 1914, one of the earliest reading devices, the Optophone [17], could allow blind individuals to understand printed text without relying on braille. The device projected light upon a character of interest, focusing the light's reflection upon a selenium photosensor. A sound with a frequency corresponding to the reflected light would then be emitted to alert the reader of the current character. A blind individual trained to use such a device, however, could only expect to read at a mere one word per minute.

While there were some OCR patents released in subsequent years [18][8], it was not until the late forties and early fifties that there was any commercial development in the OCR industry. In 1949, RCA engineers were working on an OCR system which used an early text-to-speech synthesis technology to read individual characters out loud [19]. This system required the user to move a “eye” (a cathode ray tube) across the letters of interest. The rays were then reflected upon a photosensor connected to a complex processing unit (Figure 5). The project, however, was discontinued prior to completion due to its nonviable commercial-ability.

In 1953, David Shepard patented an OCR system, “Gismo”, which could read all 26 fixed-font letters of the English alphabet, understand musical notation, and comprehend Morse Code [20]. Shepard founded Intelligent Machines Research Corporation (IMRC) and released the world's very first commercial OCR systems. Credit card reading, although now carried out through magnetic strip recognition, was one of the first commercially successful applications of OCR. The Farrington B numeric

font, still widely used on the front of credit cards to this day, was invented by Shepard in order to minimize recognition errors.

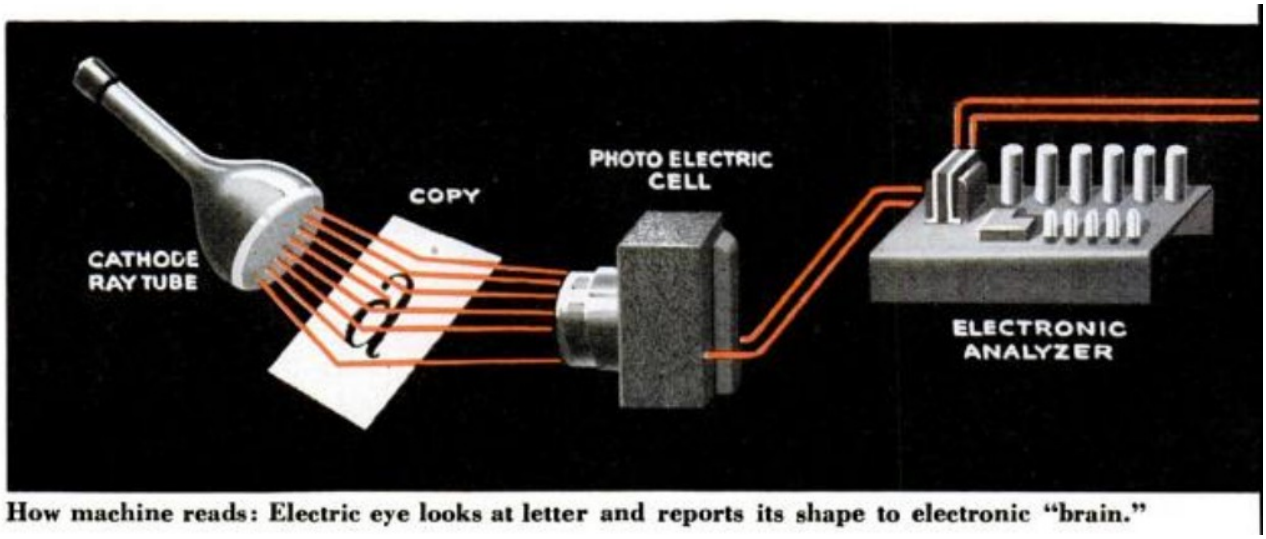


Figure 6: An image of RCA's 1949 OCR system taken from an issue of Popular Science [19]. The system was discontinued prior to completion due to its high costs .

IBM utilized Shepard's patents over subsequent years while also improving upon the accuracy of fixed-font OCR. The IBM 1408 Optical Character Reader [21] was packaged with the IBM 1401 Data Processing System (Figure 6) in 1960. The entire system, which included printer, optical reader, central processing unit, magnetic storage, etc. was sold for $146,600 [22], a price tag which, if sold by today's standards, would amount to over a million dollars. The IBM 1418 Optical Character Reader could only handle the ten numeric characters, the dash symbol, and the lozenge symbol. A later model, however, the IBM 1428, was alphanumeric. The alphanumeric reader could be programmed to read several document layout types assuming that they were printed in the correct font and format. Recognizable documents included premium notices, charge sales invoices, operations and route slips, payroll and dividend checks, and mail orders [23]. Throughout the 1960's, fixed-font OCR continued to be utilized and improved upon due to its usefulness in a variety of industrial applications. Some of the devices from this era are, in fact, still used even to this day for applications such as mail sorting and banking.

Although commercial OCR systems from the 1960's and early 1970's were primitive by today's standards, they were quite successful during their time. Maintenance costs for word processing, a then expensive resource, could be reduced significantly with ordinary typewriters used for drafting and their OCR results used for

final editing [24]. Fixed font OCR, although primitive, indeed proved to meet most of the requirements set by industry. For purposes of Assistive Technology, however, it was of little to no use. The blind or visually impaired community needed an optical reader to understand, not only OCR-specific fonts and layouts, but a wide variety of printed documents including books, newspapers, magazines, text books, etc. just as the idea of OCR originated primarily for the purpose of Assistive Technology, some of it's most important breakthroughs were driven by this same incentive.

Figure 7 The IBM 1401 System (Optical Character Reader not shown here). From left to right, the punch card reader/writer, mainframe, printer, and magnetic tape units.

2.1.2 Omnifont

A major commercial breakthrough in the field of OCR came with the introduction of Ray Kurzweil's Reading Machine in 1976 [9]. Up until this time, all OCR systems were tailored to a specific font, or perhaps a specific set of fonts. This font limitation can be attributed to the template matching algorithms commonly used at the time, which would compare each incoming character image to a library of bit-mapped images. Although recognition of a larger set of fonts can be made possible through the addition of more templates into the library, too many templates would cause the processing speed of each character to decrease significantly. Although it would be ideal to have a set of fonts which could encompass all possibilities in the

template library, this would prove unfeasible as there would be such a wide range of possibilities.

Omnifont recognition is characterized primarily by its use of sophisticated feature extraction techniques. As opposed to the brute force character-by-character template matching algorithms utilized in earlier systems, feature extraction enables recognition of characters irrespective of the font or typeset they are in. These techniques find properties which are relatively invariant for the same character with respect to the kinds of changes that occur across different typestyles. These properties can often include line segments (vectors), concavities, and loops. For example, the properties of a standard capital "B" include two loops on top of one another. Although the number “8” has this same feature, it does not have a straight edge on the left side as does the “B”. Furthermore, it is often that the two characters can also be disambiguated based on contextual analysis. For instance, if a character with the two vertically adjacent loops is detected at the beginning of a word, this character is far more likely to be a letter than a number.

The Kurzweil Reading Machine, used feature extraction and could be trained on any number of fonts. Once the system was trained on a given font (a process taking several hours), the knowledge would be stored on disk so that retraining would no longer be required. The system could be trained to handle up to nine fonts simultaneously [10]. If the page contained pictures or multiple columns, the user would be required to specify their locations manually. While sophisticated techniques have been developed to address the problems of document analysis, the following subsection section will focus on work which has been done to prevent any retraining from being required on new fonts. With the enhancements in processing speed and more abundant memory attributed to the advent of microprocessors, it became possible to implement much more intelligent systems utilizing complex pattern recognition approaches, as will be discussed in the following section.

2.2 Pattern Recognition Techniques in OCR

As with all pattern recognition applications, in OCR some combination of feature selection, extraction, and classification is essential. In general, a statistical classifier will observe the features of its input and, based upon those features, choose the optimal class label to which the input should be associated. For a given problem, there are often many combinations of features and classifiers from which acceptable results

may be obtained. The choice of classifier and feature set is largely application dependent, and, as of yet, no “one fits all solution” has been found. For OCR there are many such combinations which have been proven to yield near perfect results. This, of course, is to be expected, in that OCR is one of the most historically well-studied areas in the field of Pattern Recognition. Not only are pattern recognition techniques fundamental to character-by-character classification, but they are also essential for the detection of merged or broken characters, text lines, word recognition and linguistic analysis, and, as will be discussed in Section 2.3, document layout analysis. While a broad overview of all techniques utilized for OCR would be outside of the scope of this thesis, some of the most fundamental and important ones will briefly be discussed.

2.2.1 Text Line Finding

Character and word classification algorithms typically operate under the assumption that the unknown text to be recognized is already in the fully upright position. This is an unrealistic assumption given the many possible angles of skew with which the text may have been scanned. Skewed text, as illustrated in Figure 7 [25], is commonly encountered by most OCR systems. Assuming that page layout analysis has already extracted all of the columns and text blocks, it is then necessary to recognize angle of skew for each block. This is essential, not only so that characters may be rotated to their upright positions prior to classification, but also to prevent words and characters in vertically adjacent rows from becoming mangled inappropriately. Individual character classification algorithms will often utilize a character's positional information within a row as a distinguishing feature. As illustrated by Figure 8 [26], there is much information about a character which can be derived from where it's top, middle, and bottom portions reside within a row. In order to have access to such information, accurate text line finding algorithms are essential. Some of the most important techniques are briefly discussed.

Horizontal Projection Profile. One of the most straightforward methods for determining the skew angle of a document image uses horizontal projection profiles. When the horizontal projection profile is applied to an M x N pixel image, a column vector of size M x 1 is obtained. Elements of this column vector are the sum of pixel values in each row of the image [27]. The contents of this vector are at maximum amplitude and frequency when the text is skewed at zero degrees since the number of

co-linear black pixels is maximized in this condition. One way in which the horizontal projection profile can be utilized is by rotating the input image through a range of angles while calculating the projection profile for each one [28]. Each projection profile is then compared to determine which one has the maximum amplitude and frequency. Although much work has been done in order to optimize this approach, there are still more efficient and accurate methods which can be utilized [25].

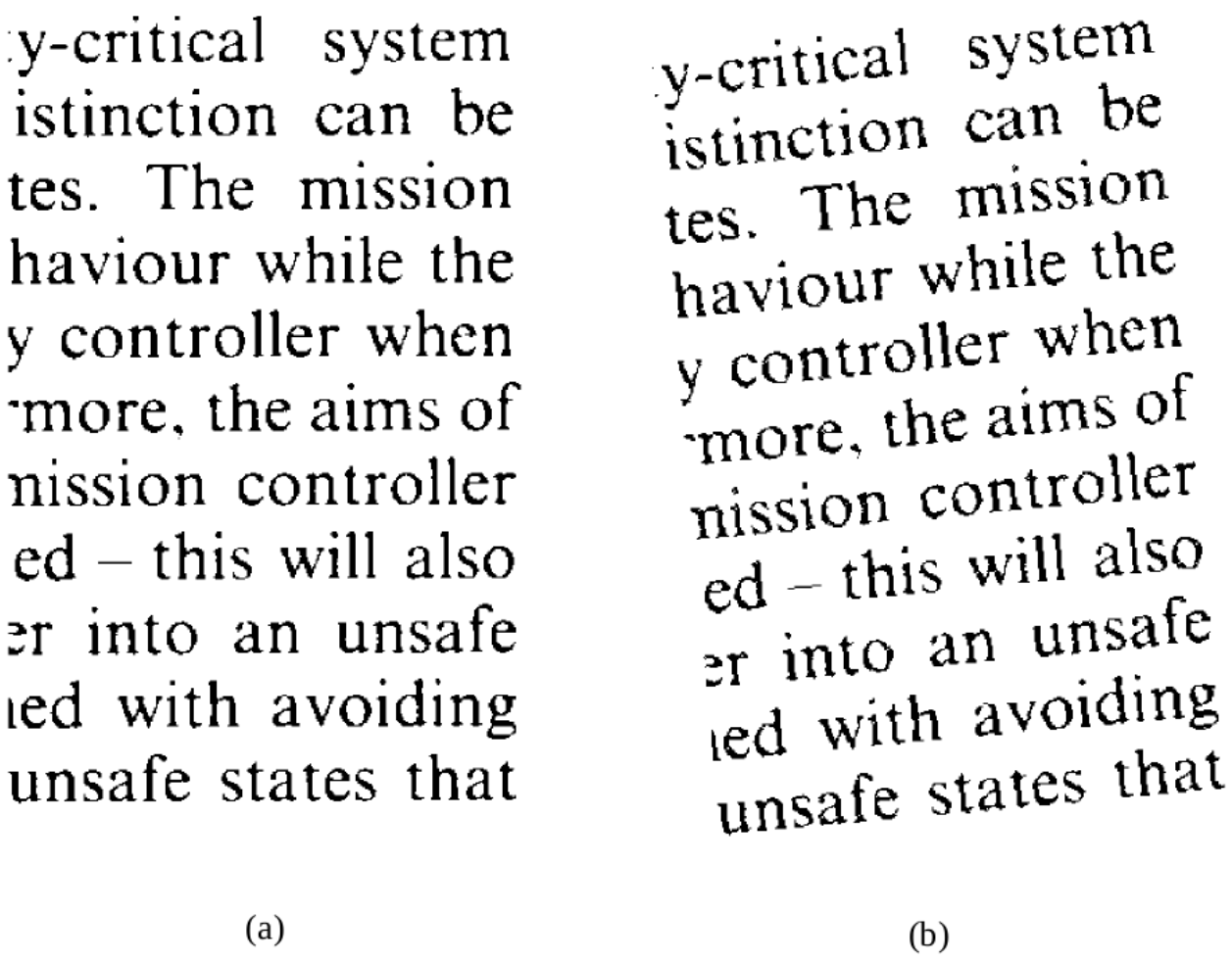


Figure 8: Original image in correct alignment (a) and skewed by 5 degrees (b). Image borrowed from [25].

Hough Transform. The Hough transform, a well known feature extraction technique in computer vision, can be utilized in order to detect, not only the skew angle of a document image, but any mathematically tractable shape of interest. This technique, when applied to 2D images, will take a series of (x, y) coordinates (for the case of document images this will likely correspond to groups of connected pixels) and

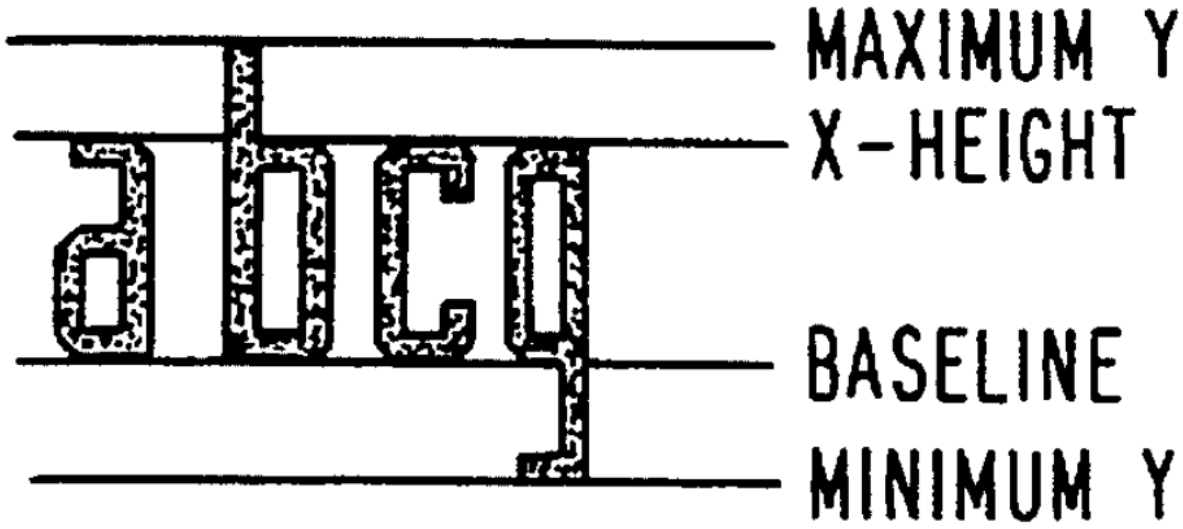


Figure 9: For printed text, a given character often has a precise position within the text line which can be useful for classification purposes [26].

transform them into a new coordinate space. While the coordinate space will vary depending upon the desired shape to be detected, for straight lines the x and y coordinates will be converted to the (ρ, θ) coordinate space using the following equation:

$$\rho = xcos(\theta) + ysin(\theta)$$

Where ρ is the distance of the (x, y) point from the origin (usually at the upper left-hand corner of the image), and θ varies between -90° and 90°. The (ρ, θ) parameter uniquely represents a given line in the image by specifying its perpendicular angle and distance with respect to the origin as shown in Figure 9.

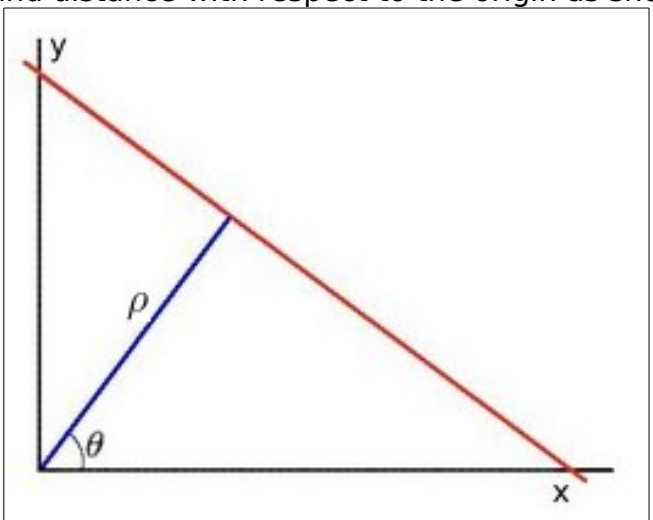


Figure 10: Mapping from (x,y) to Hugh space

For each chosen (x, y) coordinate within the image, the Hough transform algorithm will calculate the ρ values corresponding to some subset of the possible θ values between -90° and 90°. There can be an infinite set of lines going through a given point, thus the amount of lines required per point depends upon the desired accuracy of the system as well as desired overall computational speed. The set of chosen lines per point, each represented in Hough parameter space (ρ, θ), is represented by an accumulator array [29], each entry of which corresponds to a unique line in the image. Each time that a line is found to go through an (x, y) point of interest, it's corresponding entry in the accumulator array is incremented. When the process is completed, the accumulator array entries with the highest increments will correspond to the lines which intersect the most points in the image.

For OCR purposes, lines of text may be found within the image based upon this operation. The (x, y) coordinates of interest typically correspond to the centroids of connected components (groups of connected foreground pixels which often correspond to individual characters). When several parallel lines are found to have very high entries in the accumulator array, this will often mean that the page was scanned at the skew angle corresponding to these lines, and that they are likely to represent individual lines of text within the document.

Geometric Distribution of Connected Components. The Hough Transform has been utilized in various techniques to achieve accurate skew results. For a more complete survey of past techniques the reader is referred to [25]. These techniques, for the most part, vary, not by their use of the Hough Transform, but by their method for determining connected components which are of interest and most likely to correspond to rows of text. In [30], Smith utilizes an efficient and simple algorithm which, unlike previous methods, finds lines of text independently of the page's skew. The connected components of the image are extracted and filtered such that the remaining components are most likely to represent a body of text. The connected components are then sorted based on their positions in ascending order from left to right and iterates through them. Each connected component is added to a row of text to which it is most likely to belong based on vertical overlap. If no such row exists then it is created. Based upon which connected components are added to which rows, a running average is kept on the slope of the text rows. This process is continued in an iterative fashion until all connected components have been associated with rows. This algorithm has been found to achieve accurate results while proving to be more efficient than corresponding Hough Transform based algorithms.

Curved Text Line Detection. Even when text lines are accurately found, it is often the case that the lines will need to be fitted to the text more precisely due to scanning artifacts which may give the text a curved appearance as depicted in Figure 10. Among the techniques utilized for this problem are quadratic or cubic spline modeling via least square fitting techniques [31] as well as active contour tracing via snakes [32]. Smoothing techniques are often applied in order to simplify the input for curved line detection. The optimal technique to be applied largely depends upon the type of document fed into the OCR system. Thus document understanding at early stages in the OCR process is of great importance in achieving accurate results. For a more complete account of text line detection in various documents, the reader is referred to [33].

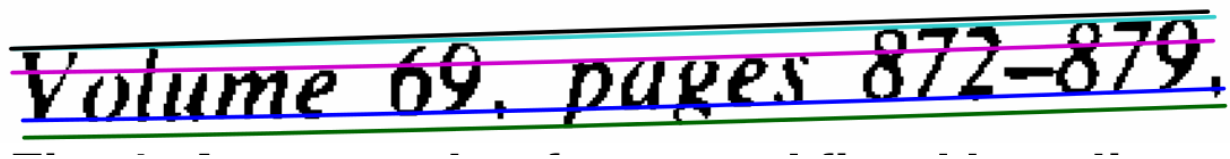


Figure 11: An example of skewed and slightly curved text borrowed from [31]. Close inspection shows that the cyan/gray line is curved relative to the straight black line above it.

2.2.2 Character Feature Extraction

The problem of feature extraction for optical character recognition, although a difficult task, has been extensively studied in the literature. Techniques vary based upon their application, with handwritten recognition often requiring different techniques from printed character recognition. As with the rest of this thesis, the focus here will be on techniques pertaining to printed character recognition. Techniques utilized by Google's OCR System, Tesseract, will be emphasized and discussed primarily since they are used within this thesis project.

Edge Extraction. After text lines have been located as discussed in the previous section, the next task for a typical OCR system will be to perform some image processing operations on the input in order to make features more easily and efficiently recognizable. In Tesseract [26], utilizes a novel edge operator which can take advantage of grayscale values if they are available to achieve robust character segmentation results. Text and non-text can often also be distinguished based on contextual evidence as well as using basic height/width filters. Furthermore, the edge

extraction algorithm will inherently filter out a significant amount of noise since it will disregard any portions of the image which do not form closed loops. The term “closed loop” is used here to describe a contour which, after being followed a certain amount of time will return to its starting position.

Also of importance is preserving the relationships between the inner and outer portions of characters. Take, for instance, the character “o” depicted at the left on Figure 11 [34]. Since the edge detector will find the inner and outer portions of this character as separate, simple data structures must be implemented which store the relationships among overlapping edges. In Tesseract, a 2D bucket sorting technique is utilized in order to store all of the inner portions of characters as enclosures or “holes” within them. The results of edge extraction are stored in chain code format as illustrated by Figure 12 [35].

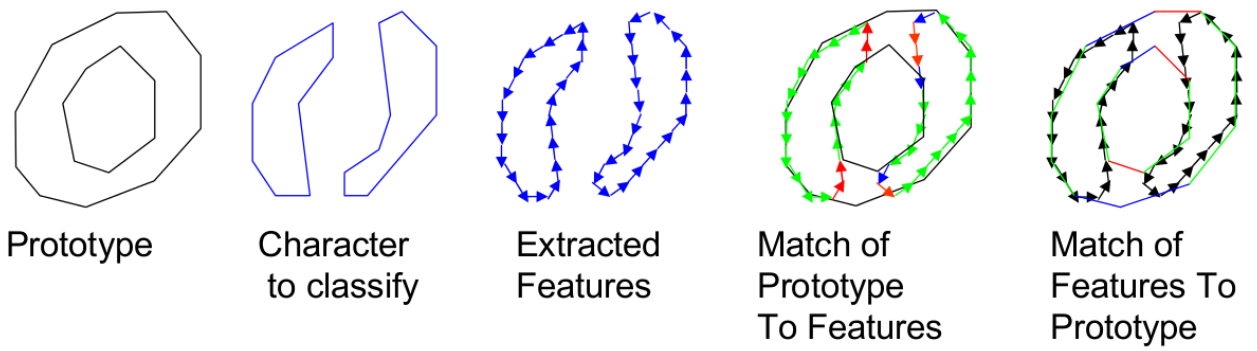


Figure 12: Comparison of a prototype for “o” to an unknown character by template matching. [34]

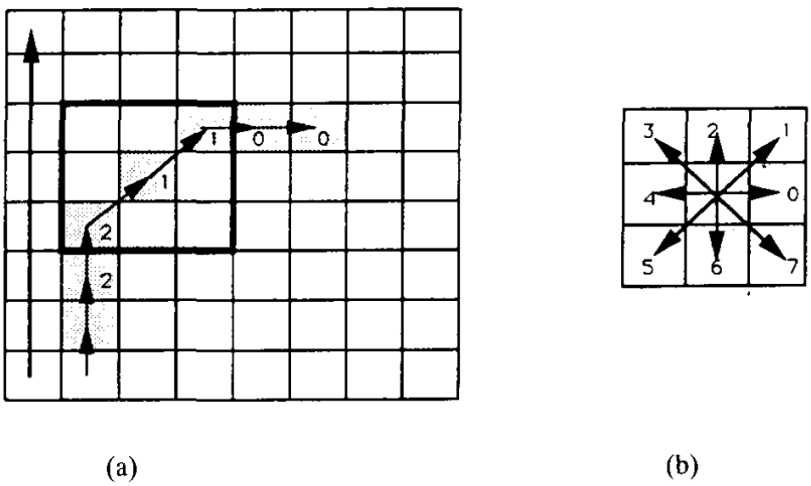


Figure 13: (a) Example of chain-code (b) Chain-code directions. [35]

Polygonal Approximation. In Tesseract, the process of polygonal approximation is required in order to optimize the efficiency and accuracy of subsequent feature extraction techniques. Polygonal approximation of a character image, if done effectively, results in an output whose data is neither too fine or course for purposes of feature extraction [36]. It becomes easier to detect global convexes and concavities as well as character enclosures, which are very important features.

The process of polygonal approximation utilized by Tesseract analyzes the chain code output of the edge extractor in order to locate simplifications which can be made, which will enhance the robustness of subsequent feature extraction techniques. The process begins by first breaking up the character into directional segments, separated by 90 deg or two subsequent 40 deg transitions [26]. The second stage involves further analysis of these segments and subsequent approximations being made between the end points of each segment. The process is repeated iteratively until certain criterion are met.

Normalization and Template Matching. After polygonal approximation and prior to feature extraction, normalization is applied to the input in order to eliminate some of the complexities which may come about from various font differences. For an in depth discussion on normalization techniques the reader is referred to Chapter 3 of [36]. Normalization is very important in accounting for character font transformations which may occur in terms of size, perspective, and rotation with respect to the features of prototypes used in training the system. Normalization techniques can, in general, be separated into categories using either linear or nonlinear methods. While linear methods account for affine transformations often found in printed characters, nonlinear techniques are generally geared more towards handwritten character recognition wherein much more drastic variation is to be expected.

Normalization can be performed either before or after feature extraction. If done after feature extraction, then the process is carried out within the feature space rather than directly on the character's pixels. In the case of Tesseract, normalization is carried out on the feature space of the character's polygonal approximation, which can be viewed as a vector of 3D features, the dimensions of which are simply x position, y position, and direction within the range of [0, 2pi] [37]. Figure 13 gives an example of how Tesseract will normalize the features of unknown characters while matching them to those of character prototypes.

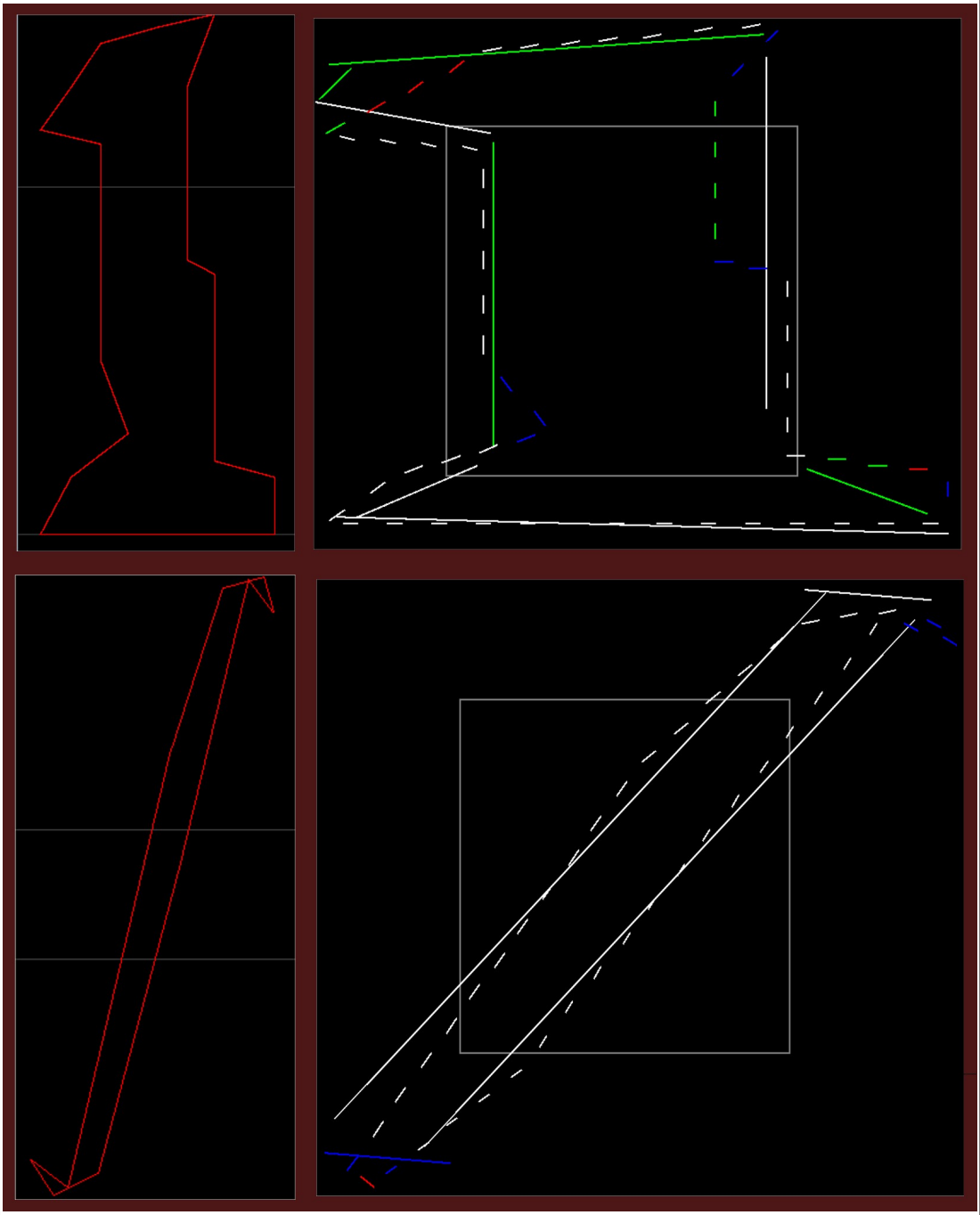


Figure 14: (top) The polygonal approximation features of a "1" followed by those same features after normalization with respect to the prototype of "1" to the right. (bottom) Features of an "integral," a character for which there is currently no valid template in Tesseract. To the right is the integral after normalization with respect to the prototype of "/". For both normalized pictures, the solid lines represent the prototypes while the dotted lines represent the normalized unknown character. Lines are colored from best to worst match: white, green, red, blue. These images were taken from Tesseract's debugger.

As illustrated by Figure 13, Tesseract normalizes a feature vector by each character prototype to which it is compared. For instance, assume that the character “8” is fed into the system. Based upon a coarse shape analysis of the character a subset of the the total prototypes may be chosen as potential candidates. For instance “B” may be chosen since it has two enclosures, and “0” may also be chosen based upon its convex top and bottom regions. Assuming that only these characters are chosen as candidates, the feature vector for “8” will be subsequently normalized based upon both of these prototypes prior to the respective template matching. The process of normalization begins by isotropically scaling the bounding box to a fixed height and width. The feature vector is then centered and scaled anisotropically based upon the second moments of the prototype to which it is being compared [37]. Moment-based character normalization has been studied extensively in the literature dating back to even before the advent of microprocessors. For some examples of in-depth studies the reader is referred to [38] [39].

During the classifier training, the training data is grouped into clusters based upon certain important features. These feature clusters are then utilized during classification to reduce computation time with very little loss in accuracy. The five most important features utilized by Tesseract will be herein briefly discussed.

Concavities. One of the most important features in character recognition are concavities. By definition, a concavity is part of an outline which does not lie on it's convex hull (the smallest convex region enclosing the outline as illustrated by Figure 14 [40]). In Tesseract, a concavities are characterized by the direction of their hull line, their centroid [26], shape, skew, and area.

Figure 15: Red outlines represent convex hulls [40].

Functional Closures. Character closures are common features which can be useful in distinguishing characters regardless of their font. For instance the “e” and “o” characters will both always consist of a single closure no matter what. The term functional closure is useful when a character's closure may be slightly degraded somehow, such that there there may be an unintended opening. In Tesseract [26], each concavity is tested for functional closure. Based open the location of the concavity within the character (i.e. upward facing, downward facing, etc.) a threshold is assigned for the maximum character to concavity width ratio expected for a functional closure. If the ratio is below the appropriate threshold then a functional closure will be detected.

Axes. Tesseract defines axis features only on characters for which no concavities or closures are detected. Characters including commas, periods, quotations, etc. fall under this category. The axis feature measures a character's length to width ratio. The length of a character is determined by finding a point on the outline whose distance from the character's centroid is maximum. The vector going from the point to the centroid is said to be the character's major axis. The character's width is then calculated as the sum of the maximum perpendicular distances from the major axis to the character outline on either side of the axis. The major axis length to character width ratio can be useful in disambiguating commas, periods, quotes, etc.

Lines. As illustrated by Figure 13, lines are useful features in template matching. Line features are only used by Tesseract for unknown characters which closely match more than one of the prototypes, as measured with concavities, closures, and axes [26]. The degree to which a line in the unknown character matches a line in a prototype is measured based upon the normalized position of the center of the line, its quantized direction, and its scaled length.

Symmetry and Detection of Italicized Characters. Vertical as well as horizontal symmetry can be a very useful measure in discriminating certain characters. For instance, the character “C” and “G”, “j” and “/”, “j” and “]”, “T” and “1”, etc. can often be disambiguated through their respective measurements of symmetry. The main difficulty in symmetry measurement is not in measuring the degree of symmetry about an axis, but rather in locating the axis of interest. While the problem is trivial for vertical text (simply drawing a vertical line through the center will suffice), italicized text is much more difficult since the axis is rotated slightly and may be difficult to locate. Tesseract utilizes two methods to determine a character's axis of

symmetry. Once this axis is found it is then easy to determine whether or not the character is italicized.

The first method used by Tesseract searches the character's outline for a vector which passes from the bottom to the top half of the character's bounding box. The direction of this vector may be a good indication for the direction of the axis of symmetry. For round characters such as “o” and “e” and those which contain vertical lines such as “H” and “p”, this method is useful. However, for angular character such as “X” or “8”, a valid result is not produced. The second method finds the rightmost point on the outline then calculates the most clockwise line which can be drawn through this point, without intersecting any other point on the outline. This operation is repeated on the leftmost point of the outline as well. The line which was least clockwise from the vertical becomes the axis of symmetry.

After the axis of symmetry found, the outline is searched around the axis for points of reflection. Symmetry testing is commenced at a point where the axis intersects the outline and works in opposite directions simultaneously. The points are tested for being in the same locality of the point on the opposite side of the axis. Symmetry is only measured for certain character candidates and typically only in one direction (either vertical or horizontal).

2.2.3 Character Classification

The line finding, edge extraction, polygonal approximation, and feature extraction techniques discussed thus far would be of little to no value without an effective classifier. In pattern recognition, a classifier will take a set of feature measurements as input and, using these measurements, choose from a finite set of classes, the class to which the unknown input is mostly likely to belong. In the case of OCR, the classes will often correspond to individual characters. Tesseract employs two separate classifiers, one is termed the static classifier while the other is the adaptive classifier. In order to save computational time, a class pruner is utilized first to narrow down the number of candidate classes for an unknown character.

Tesseract Class Pruner. In the first stage of classification, Tesseract will employ its class pruner in order to reduce the number of potential candidates to which an unknown character is to be compared. The class pruner uses a fixed quantized version of the 3D feature space wherein each of the 3 dimensions (x, y, theta) are quantized into 24 cells. After the unknown character's features are quantized, they are

indexed to the quantized feature space in order to obtain a set of classes which allow the given features. The number of feature hits for each class is summed and the best few matching classes are then fed into the next stage of classification [37].

Tesseract Static Classifier. Both Tesseract's adaptive and static classifiers are unique when compared to more standard techniques in that they operate on a variable number of features. While standard classifiers such as neural networks, support vector machines, etc will work in a feature space of fixed dimension, Tesseract has a variable number of features for each class of interest. The classifer can be regarded as an optimized K-Nearest-Neighbor (KNN) classifier where the character class, k, with minimum distance from the unknown character is computed as follows:

$$argmin(k)\frac{1}{M+J_k}\left(\sum_{l,i}\left(x_{il}-\mu_{ijk}\right)^2+\sum_{j,i}\left(x_{il}-\mu_{ijk}\right)^2\right)$$

Where the variables are as follows: i is the current feature dimension (either x position, y position, or θ); j is the cluster; k is the character class; and l is the unknown's feature. x_{il} is the feature dimension (either x position, y position, or θ) of the unknown character x's feature at index l. M is the total number of feature vectors in the unknown character, x (this varies depending upon the character of interest). J_k is the total number of character clusters which the training set was divided into. μ_{ijk} is the mean feature value for the i^{th} feature, j^{th} cluster, and k^{th}class calculated during training.

While the left-most summation measures the distance between each feature dimension and its corresponding average clustered prototype value, the right-most summation measures the distance between the average value in each cluster to the corresponding feature dimension. The result of these summations is then divided by the total number of features in the unknown character and training set. A key advantage to this approach is it's symmetry. The nearest matching features between both the unknown and prototype and the prototype and the unknown are effectively found. Say, for instance that the unknown character is “e” and the prototype to which it is compared is “c”. Since most of the features in “c” are allowed by “e”, it becomes possible that the “e” will be misclassified as “c” if only the distance between the “e” and “c” is computed. When the distance between the “c” and “e” is added into the

classification, the lack of crossbar in the “c” will incur a penalty, thus lowering the risk of misclassification.

Tesseract Adaptive Classifier. After word recognition, as will be briefly discussed in section 2.2.5, a second pass is made by Tesseract's classifier. This time the classification is considered to be adaptive in that it utilizes the extra information obtained after word recognition in order to better train the classifier to the current font. After word recognition is carried out, there may be several characters which can be disambiguated and thus used to better train the classifier on the second pass and increase accuracy. The adaptive classifier is essentially the same as the static one except that it applies a different type of normalization to the unknown character prior to comparing it to the prototype. While, for the static classifier, the centroid of the unknown character is centered in the feature space and then scaled anisotropically to normalize the second moments of the outlines, the adaptive classifier will normalize the unknown by centering the horizontal centroid of the outline and scaling isotropically to normalize the x-height of the character. This normalization retains font differences, which, at this stage of OCR, is very important [37].

2.2.4 Detection of Merged or Broken Characters

While some of the first OCR systems would only recognize each individual character independently, more sophisticated systems such as Tesseract, Omnipage, and Abby Fine Reader, ReadIris, etc. analyze inter-character relationships in order to increase their systems' robustness in the presence of noise. In Tesseract, while the results of word recognition (described in Section 2.2.5) are unsatisfactory, a character merger/segmenter module is utilized in order to test the word on new potential character candidates in areas with low character recognition confidence. The merger/segmenter module will locate concave vertices of a questionable character's polygonal approximation and attempt to separate the character in those locations to test for a possible merged character as illustrated by Figure 16. Likewise, potentially broken characters are attached to their neighbor and tested if their combined width is within an acceptable range.

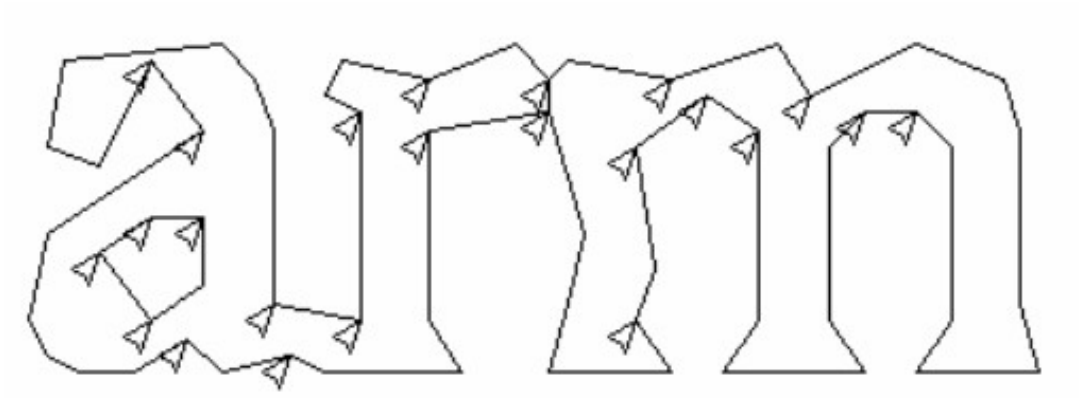


Figure 16: Example of merged letters with candidate chop points (denoted by triangles) [31].

2.2.5 Word Recognition and Linguistic Analysis

Individual words within Tesseract are detected based upon the distribution of space between characters found on a text line. Characters which are within an appropriate horizontal distance parallel to the text line are considered to be within the same words, while groups of such characters are considered to be separate words. The word recognition module looks up recognized words in a dictionary to make sure they are valid. This information is also vital to detecting broken or merged characters and training the adaptive character classifier.

Some basic linguistic information can be important for increasing accuracy. For instance, in Tesseract's English word recognition module, numeric characters are not allowed to exist in alphabetic words, uppercase characters cannot follow lower case ones, and the only punctuation allowed within a word are apostrophes. Markov Methods are also very useful in OCR due to spelling conventions (such as u following q) and the need for words to be pronounceable (i.e. g is unlikely to follow j). By modeling each individual character as a possible state and each character occurence as the next element in a markov chain, it is possible to use a transition matrix (whose width and height are 26, the number of characters in the English alphabet) to help in selecting a word's next character [26]. While it is possible to make choices based upon multiple characters, the transition matrix for making a choice based upon the previous m-1 characters would require a transition matrix of size $26^{(m-1)}x26^{(m-1)}$. Rather than using large values for m, Tesseract employs dictionary methods.

Strings of characters can be reduced into words which are either in a dictionary or can be generated through the use of various production rules [41]. For each string

of characters a set of candidate words are derived using the dictionary. The word which has the highest overall rating based upon the recognition confidence of its individual characters is chosen. The word recognition result can then be utilized in order to boost the adaptive character classifier's accuracy since certain characters which had low confidence in the static classifier may now be confirmed.

2.3 Document Layout Analysis Techniques

The improvements made in the field of commercial OCR throughout the 80's and early 90's are primarily attributed to enhanced processor and digitizer technologies rather than to improved classification techniques for individual patterns [42]. By the early 90's there had been significant progress already made toward the study of OCR and pattern recognition techniques which are still largely in use to this day. A significant amount of the more recent progress made in the state-of-the-art has been due to improvements in document layout analysis and understanding as opposed to the much more mature character-by-character feature extraction and classification algorithms.

While inflexible hardwired classification engines once dominated the market for OCR, the computational advancements of the 70's and 80's allowed for more intelligent systems take hold. While systems became robust against multiple fonts, merged/broken characters, and document skew as discussed in the previous section, the need also arose for systems which could recognize pages from a wide variety of document types. While an OCR system may be predominantly exposed to documents like newspapers, magazines, letters, etc., it is also often necessary to process such “special” documents as electronic circuit diagrams, envelopes, checks, tax return forms, music notations, etc. [43].

The importance of document layout analysis techniques is made apparent in the presence of both the former and latter document types as they may contain complex backgrounds, lines with drop-caps, mathematical formulas, various symbols, imagery, tables, graphs, multiple columns, titles, headings/subheadings, etc. Therefore, it becomes important, not only to recognize the individual words and characters, but to also interpret and preserve the layout and spatial context of a document's components. Such details as spatial context and document structure are vital in conveying a document's message as it is intended to be perceived, as well as for understanding how exactly the document needs to be processed in order to

achieve the optimal recognition accuracy. Figure 17 [44] shows two examples of document images with complex layouts.

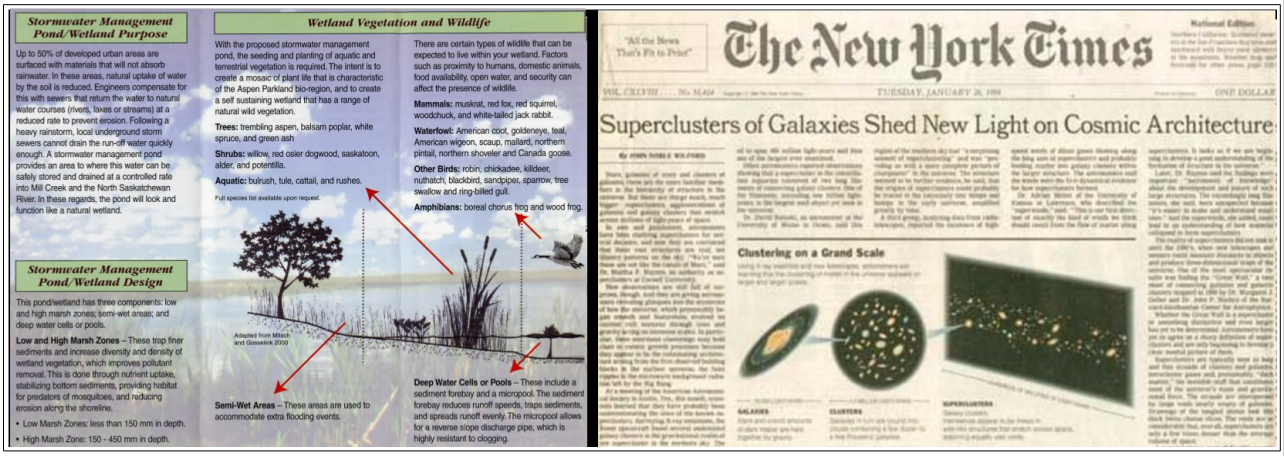


Figure 17: Two document images with complex layouts [44]. On the left, an intricate background as well as complex column structure is observed. On the right is a newspaper article with a complex layout of titles and columns along with imagery and captions. Both document images will require sophisticated image processing and layout analysis techniques in order to achieve both OCR accuracy as well as efficient data storage and indexing within computer systems.

Document layout analysis is a very important design component for any OCR system and has been extensively studied in previous literature [45][43][44][46][47][48][49][50]. Not only is document layout analysis often essential for obtaining correct OCR results, it can also provide the means for computer systems to use logical information such as titles, footers, authors, captions, abstracts, page numbers, etc. to more efficiently store and index a document image's information [51]. This contextual information is also essential for Assistive Technology purposes, in enabling blind individuals to have an understanding of the same spatial and logical cues afforded by the document's visual layout [52]. This section will discuss how the field of document analysis is divided into various sub-problems by existing literature and then compare and contrast various techniques which address these problems. The problem of document analysis can be broadly divided into its five most important interdependent components: image preprocessing, document structure analysis, document content representation, training set development, and finally performance evaluation as illustrated by Figure 18 [44]. After a brief overview of what each stage entails along

with some introduction of terminology, various techniques found in the literature for each stage will be discussed.

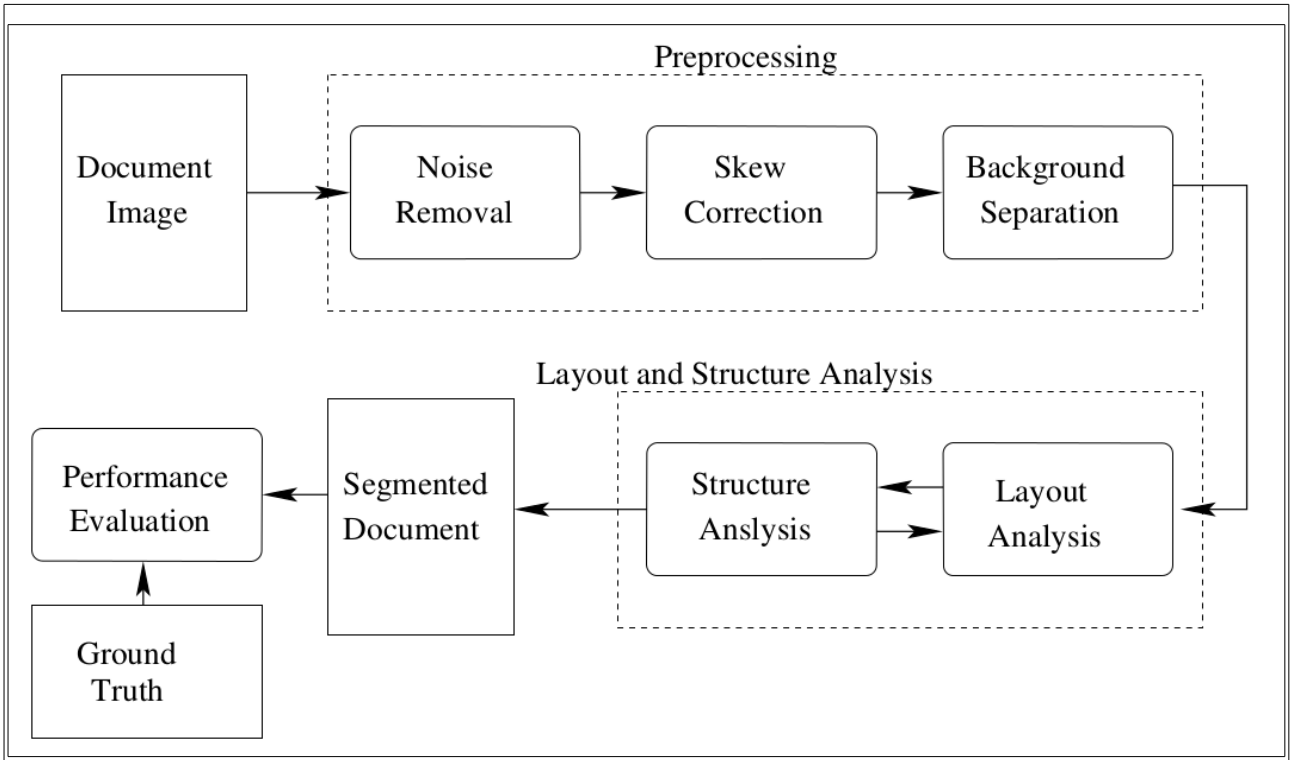


Figure 18: The five most important inter-dependent components of document analysis involve preprocessing, document structure analysis, document content representation (not illustrated here), training set development (ground truth), and performance evaluation. Each module is described as interdependent because the performance of the overall system really depends on each component. For instance, if preprocessing is not effective, then structure analysis will likely fail. If the document content representation is not consistent, then ground truth and performance evaluation will yield insignificant results. Diagram borrowed from [44].

Firstly, the most common problems addressed by image preprocessing (Section 2.3.1) in document analysis involve noise removal, separation of background and foreground regions, and skew correction. Secondly, after any necessary preprocessing is carried out on the document image, the modified image is fed into the system's document structure analysis module. From a broad perspective, document structure analysis involves first extracting the document's geometric structure and then mapping that structure into a valid logical one which can be understood by computer systems. A document is thus considered as having both a physical (geometric-based) structure and a logical (content-based) structure. Thus document structure analysis is commonly divided into two distinct phases: physical layout analysis and logical layout

analysis. Each distinct phase of document structure analysis will be further discussed in Section 2.3.2.

Thirdly, an important question which must be asked prior to the design of any structural layout system, is how exactly a given document should be represented by a computer internally. This brings about the problem of document content representation (Section 2.3.3) which also has been extensively addressed by the literature. A standard format for all documents is of course desired, however the problem of finding a standard format which could accommodate all possible document image layouts has been no trivial one. A variant of SGML or HTML is commonly employed, however the tradeoffs between different representations as well as grammars will be briefly reviewed. Fourthly, given the knowledge of how a document should be represented internally, an important question is what training sets and groundtruths are available for evaluating performance (Section 2.3.4). Since building a training set manually is an extremely expensive and time consuming process, the ideas of creating synthetic training data and/or training a system to automatically generate training data from documen[64]ts while requiring little to no human correction, have been explored. Finally the issue of performance evaluation has been explored through various techniques which will be discussed in Section 2.3.5.

2.3.1 Preprocessing

The most common problems addressed by image processing involve noise removal, separation of background and foreground regions, and skew correction. Since skew correction was already covered in great detail by Section 2.2.1, it will not be further discussed in this section. Noise removal in image processing is a well studied field and advanced techniques have been developed to cope with white noise, salt and pepper noise, quantization artifacts, etc. Such noise sources are often compensated for by using techniques such as median filtering, dithering, low pass filtering, etc. [49]. An in depth overview of noise reduction methods in image processing can be found in [53]. For purposes of document layout analysis, one of the more important noise removal tasks involves the detection and filtering of half-tones. This discussion will be followed by a brief overview of preprocessing tasks for background and foreground separation.

Noise Removal: Dealing with Half-tones

Halftones, as illustrated by Figure 19 [54], utilize variably sized or space dots in order to create the optical illusion of an infinite range of colors while, in actuality, only printing a limited amount. Half-tones are utilized by color and grayscale printers in order to reproduce imagery while requiring few colors of ink. Figure 19, for instance, creates the illusion of grayscale while only requiring black dots. When scanned at high resolution, the halftones in a document become a significant noise artifact, as an image's connect components clearly should not be divided into such small dots for document analysis purposes. Halftones can be detected through the use of various filtering techniques [55], whose accuracy often depends upon the dot sizes and spaces of the halftone in question. Once detected, the halftones can be converted into continuous grayscale by applying an appropriate low-pass filter to smooth out all of the dots, followed by a sharpening technique which will reduce the blur.

Figure 19: An example of a halftone image [54]. Notice that, when looking at the image from a distance, the illusion is created that the image is in grayscale, when, in fact, it is actually printed with only black dots of varying sizes.

Background and Foreground Separation

Although the problem of foreground detection is often very simple in the case of the most typical black text on white background, the problem becomes much more complex when faced with intricate backgrounds which are overlayed with text in varying color, size, and font as depicted in Figure 20. In the former case it is possible to use thresholding techniques like Otsu's Method [56]. An alternative method which could work for varying background and foreground color schemes would be to find the outline of characters through edge detection [26]. In the presence of complex backgrounds, however, more sophisticated background and foreground separation techniques may be required. A common approach is to compute statistical properties of image patches and assign them as either foreground or background using a trained classifier such as a neural network [44]. Through a combination of edge detection and a trained classifier it becomes possible to detect foreground text of varying colors on a complex background with a certain degree of confidence as demonstrated in Figure 20.

Figure 20: (Left) Part of a document image with complex background. (Right) The same image with foreground separated from background [44].

2.3.2 Document Structure Analysis

A primary component of any document analysis system is the document structure analysis stage itself. As previously indicated in Figure 18, however, the steps of preprocessing, document content representation, training set development, and performance evaluation also play a crucial role. In this section the term “document structure analysis” is used to refer to the broad class of both physical and logical document structure analysis methods which will be explored in this section. In general, physical layout analysis techniques are one of the first steps of an OCR system and will initially divide the document image into areas perceived as text and non-text, as well as splitting multi-column text into columns [57]. In this literature review an important distinction between physical and logical layout analysis techniques is made such that, while logical layout analysis techniques make final classification decisions on blocks, physical layout analysis techniques extract and evaluate the geometry of blocks without necessarily reaching any final conclusions on their syntactic meaning. While the physical layout analysis stage looks for geometric patterns, the logical layout analysis stage will utilize this and other information in order to infer a document's meaning from a syntactic perspective (i.e. the type of document and the location and functional purpose of its “zones” which may include titles, headers, footers, math equations, imagery, etc). This logical understanding is very important both for indexing and storage purposes as well as for Assistive Technology applications as previously mentioned.

A document analysis system must be able to understand, not only how a document can best be partitioned into its logical sections, but also the role that physical geometry plays in conveying information effectively. It is important for a document recognition system to move back and forth between physical and logical analysis in an intelligent manner which may vary significantly depending upon the aspects of what is being recognized. This concept is illustrated by the bidirectional arrows seen in Figure 18. Although it is possible for a very specific physical layout to match to only a single logical structure (i.e. in the case of a very complex and unique form), there is never a guaranteed one-to-one mapping between any physical and logical layout or vice versa. In creating a system that can generalize to a wide variety of document structures while minimizing overfit, it is thus important not to make assumptions too early based solely upon geometric information. A system may require to make “fuzzy” decisions which, in later steps, can be further refined to reach an

appropriate solution. For instance, if text is found to be centered within a column this could open up many distinct possibilities based upon the contents of the text itself as well as its context within the entire page. It could, for instance, be the title of a new subsection, new chapter, a mathematical formula, a quote, image caption, or any number of other possibilities. Thus, while an understanding of the geometric structure of a block of text is important, there is more information required in order to understand the block's logical structure. If an OCR algorithm yields results with low enough confidence then various alternatives can be tested (i.e. for mathematical formulas, musical notation, other languages, etc.).

Document Physical Structure Analysis

Physical layout analysis, an essential step for all OCR and document analysis systems, localizes individual blocks of text and imagery while leaving assignment of logical meaning of these blocks as well as final classification of text/nontext regions to later stages in processing which will be discussed in the Document Logical Structure Analysis Section. Methods for physical layout analysis fall into roughly three categories: top-down, bottom-up, and hybrid, each of which will be discussed in turn by this section. An important distinction between algorithms involves the types of physical layouts which they can handle. The following three types of physical layout patterns are commonly defined: these include Manhattan, rectangular, and arbitrary layouts [58]. A document's Manhattan layout can be viewed as the document divided into a grid, which may be horizontally or vertically split recursively into smaller components in any given region. For a Manhattan layout, if a region overlaps another then it must be entirely covered by that region (i.e. there is no partial overlap). Rectangular layouts consist of several rectangles arbitrarily spaced apart or which could be partially overlapping. Arbitrary layouts, on the other hand, are formed by unconstrained polygonal shapes as demonstrated by Figure 21 [59].

Top-down physical layout analysis techniques recursively segment the document into smaller rectangles which are expected to correspond with image, column, paragraph, or other text block boundaries [51]. Bottom-up techniques, on the other hand, analyze individual pixels or connected components, recursively merging them together into larger regions. While bottom-up techniques can handle arbitrary physical layouts, top-down methods are constrained to only handling rectangular regions. A disadvantage of bottom-up techniques, however, is that they may result in over-fragmented regions. For instance, a bottom-up technique will be more likely to

properly segment small structures like individual paragraphs of text than to properly segment entire columns. Due to these trade-offs it is often that hybrid techniques, which combine top-down and bottom-up ideas, are employed [60]. Starting with top-down methods, variants of each broad category of physical layout analysis will be reviewed by this section.

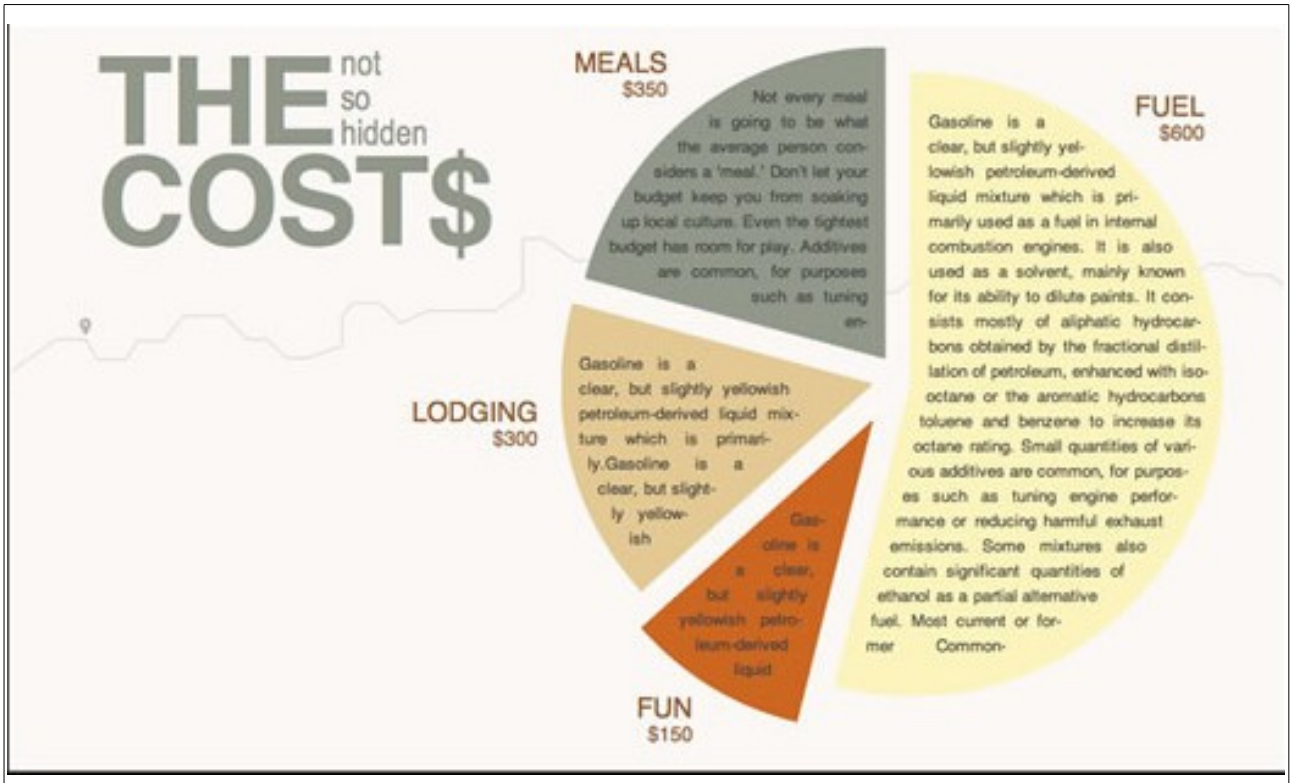


Figure 21: An example of an arbitrary physical layout wherein blocks of text are fit to the shape of a pie chart. A layout analysis system should ideally be able to segment text blocks into the appropriate shape, which sometimes may be more complicated than simple rectangular layouts. For this figure, a document layout analysis system which can only handle rectangles would be insufficient, and would likely result in a mangled output. Image borrowed from [59].

Top-Down Physical Structure Analysis

The first physical layout analysis technique to be reviewed here is the “top-down” method. Top-down strategies segment blocks based upon interpretations of the document from a high level (i.e. by first looking at a representation of the entire document and recursively splitting it into smaller components). Top-down strategies will then typically attempt to verify each segmentation by visiting each node down the

to the terminals (the lowest levels, corresponding to individual connected components or pixels) [61]. For documents having a complex layout, top-down methods are often more robust but slower than bottom-up ones. Typical bottom-up algorithms are faster, but can be less reliable since they may greedily over-segment blocks without regard to all of the available contextual information.

X-Y Cut Algorithm. The X-Y cut algorithm [62] is a top-down approach which has been utilized extensively over the past several decades. The technique analyzes vertical and horizontal projection profiles of the image to find regions of low pixel density, often termed as “valleys” [43][44]. Assuming that the document has a white background and Manhattan layout, its X and Y valleys are likely to correspond with horizontal and vertical text block boundaries respectively. For instance, these could be divisions between paragraphs and columns. The X-Y cut algorithm will start with the horizontal and vertical projection profiles of the entire image and use the largest valley (or valleys) in either direction as the first splitting point. After having made the first split(s), the algorithm will then recursively make further splits within each sub-region using the same methodology. The document's physical layout is represented by an X-Y tree data structure wherein each node represents a split region. If the algorithm is correct, then the terminal nodes of the tree will correspond to the individual text blocks. Once the terminal nodes have been located, the algorithm will backtrack through the tree structure to ensure that the physical structure is appropriate based upon some preconceived notions of expected document structure. A possible result of the algorithm is illustrated by Figure 22. In order for the X-Y cut algorithm to work correctly, it is vital that the document first has its skew corrected. If, for instance, the horizontal projection profile is taken for a document that has been rotated by several degrees, then many of the “valleys” will not be found correctly and thus the algorithm will fail.

Shape-directed Cover Algorithm. In an attempt to combine the strengths of top-down and bottom-up methods (i.e. faster run time for the more greedy bottom up methods but more global knowledge for top-down), Baird et al. [63] proposed a “global-to-local” strategy which first finds the rectangular coordinates of all foreground connected components and then finds all of the maximal white space rectangles surrounding them. A white space rectangle is considered maximal if it contains only white pixels and cannot be further expanded while staying entirely white. The white space rectangles are then sorted into a binary tree structure where

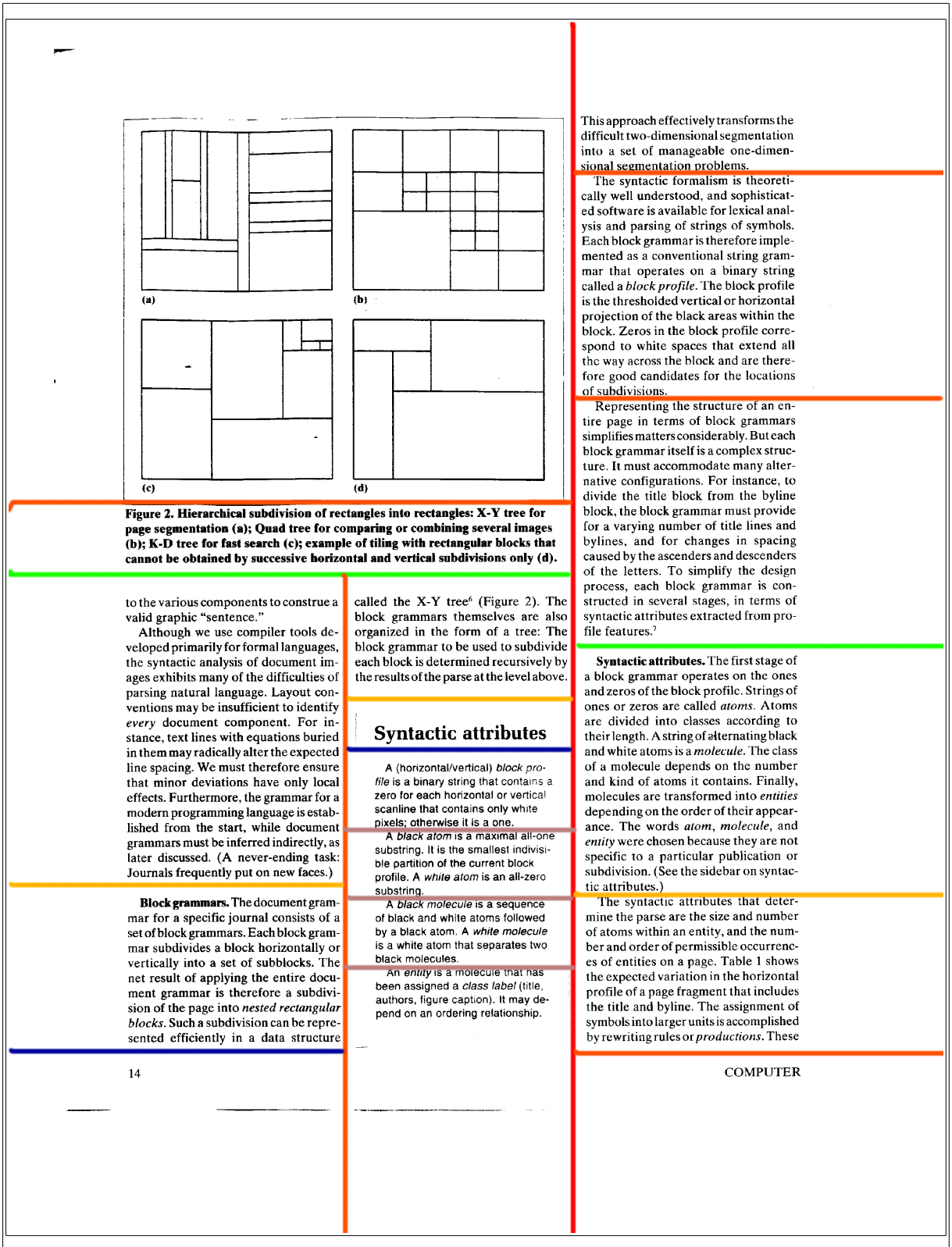


Figure 2. Hierarchical subdivision of rectangles into rectangles: X-Y tree for page segmentation (a); Quad tree for comparing or combining several images (b); K-D tree for fast search (c); example of tiling with rectangular blocks that cannot be obtained by successive horizontal and vertical subdivisions only (d).

to the various components to construe a valid graphic "sentence."

Although we use compiler tools developed primarily for formal languages, the syntactic analysis of document images exhibits many of the difficulties of parsing natural language. Layout conventions may be insufficient to identify *every* document component. For instance, text lines with equations buried in them may radically alter the expected line spacing. We must therefore ensure that minor deviations have only local effects. Furthermore, the grammar for a modern programming language is established from the start, while document grammars must be inferred indirectly, as later discussed. (A never-ending task: Journals frequently put on new faces.)

Block grammars. The document grammar for a specific journal consists of a set of block grammars. Each block grammar subdivides a block horizontally or vertically into a set of subblocks. The net result of applying the entire document grammar is therefore a subdivision of the page into *nested rectangular blocks*. Such a subdivision can be represented efficiently in a data structure called the X-Y tree[6] (Figure 2). The block grammars themselves are also organized in the form of a tree: The block grammar to be used to subdivide each block is determined recursively by the results of the parse at the level above.

Syntactic attributes

A (horizontal/vertical) *block profile* is a binary string that contains a zero for each horizontal or vertical scanline that contains only white pixels; otherwise it is a one.

A *black atom* is a maximal all-one substring. It is the smallest indivisible partition of the current block profile. A *white atom* is an all-zero substring.

A *black molecule* is a sequence of black and white atoms followed by a black atom. A *white molecule* is a white atom that separates two black molecules.

An *entity* is a molecule that has been assigned a *class label* (title, authors, figure caption). It may depend on an ordering relationship.

This approach effectively transforms the difficult two-dimensional segmentation into a set of manageable one-dimensional segmentation problems.

The syntactic formalism is theoretically well understood, and sophisticated software is available for lexical analysis and parsing of strings of symbols. Each block grammar is therefore implemented as a conventional string grammar that operates on a binary string called a *block profile*. The block profile is the thresholded vertical or horizontal projection of the black areas within the block. Zeros in the block profile correspond to white spaces that extend all the way across the block and are therefore good candidates for the locations of subdivisions.

Representing the structure of an entire page in terms of block grammars simplifies matters considerably. But each block grammar itself is a complex structure. It must accommodate many alternative configurations. For instance, to divide the title block from the byline block, the block grammar must provide for a varying number of title lines and bylines, and for changes in spacing caused by the ascenders and descenders of the letters. To simplify the design process, each block grammar is constructed in several stages, in terms of syntactic attributes extracted from profile features.[7]

Syntactic attributes. The first stage of a block grammar operates on the ones and zeros of the block profile. Strings of ones or zeros are called *atoms*. Atoms are divided into classes according to their length. A string of alternating black and white atoms is a *molecule*. The class of a molecule depends on the number and kind of atoms it contains. Finally, molecules are transformed into *entities* depending on the order of their appearance. The words *atom*, *molecule*, and *entity* were chosen because they are not specific to a particular publication or subdivision. (See the sidebar on syntactic attributes.)

The syntactic attributes that determine the parse are the size and number of atoms within an entity, and the number and order of permissible occurrences of entities on a page. Table 1 shows the expected variation in the horizontal profile of a page fragment that includes the title and byline. The assignment of symbols into larger units is accomplished by rewriting rules or *productions*. These

Figure 22: A possible result of the X-Y Cut algorithm on a page taken from [51]. Here the entire page is cut vertically (red) and then each sub-region is cut horizontally (green). The splitting order from this point becomes rather complex but is color coded as follows: orange, yellow, blue, and pink. Notice that a single node may have more than two children, which is the case for sections with multiple paragraphs, columns, etc.

the right-most white space rectangles are at the root, and the left-most are the leaves. Multi-way branches which occur when there is more than one maximal white space rectangle at a given X coordinate, are handled using singly linked lists as entries in the binary tree. Unlike most of the previous top-down physical layout analysis research, Baird focuses intently on algorithmic complexity. When he denotes the number of maximal white space rectangles as m and the number of foreground rectangles as n, he found his algorithmic complexity for sorting the white space rectangles to be $O(nlogn + m)$.

Once the rectangles are sorted, a subset of these rectangles denoted as the "cover set" is chosen. Any regions of the image not covered by the union of this cover set will define the segmented text blocks. In order to speed up processing time, the rectangles in the cover set are chosen in a greedy fashion using the high level information available in the binary tree. In terms of processing speed, this can prove advantageous over the X-Y Cut algorithm which uses extensive backtracking. The cover space is chosen based upon domain specific information. For instance, in Manhattan layouts, the white space rectangles between columns will typically have a high (but not too high) aspect ratio. Baird et al. thus assign shape scores to the rectangles in order to favor the most significant and choose the cover space based upon these scores. Experiments were run on over 100 Manhattan layouts which included typewritten and printed pages from letters, magazines, books, journals, and newspapers, which included complex layouts consisting of headers, footers, embedded mathematical equations, graphs, multiple columns, etc. The authors reported near perfect results for large column structures but would observe errors for smaller blocks of text especially in the presence of noise.

White space cover algorithm by Breuel. Breuel presents a variation of Baird's white space analysis algorithm which is simpler to implement (requires less than 100 lines of java code) [64]. The algorithm starts by picking one of the black rectangles, called the "pivot", toward the center of the image. Since the maximal rectangle cannot contain the pivot, there are now four distinct possibilities for the maximal rectangle's location: above, below, to the right, and to the left of the pivot. Each sub-rectangle is then evaluated using a quality measure to determine which is most likely to contain the maximal rectangle. After the sub-rectangles and their respective quality measures are inserted into a priority queue, the above steps are repeated. This process continues until a fully white-space region is detected. The rectangle

corresponding to this region is the optimal solution. The results of this algorithm were described as favorable when run on the same dataset as Baird (the UW3 Database [65]), with no errors observed on 223 pages. An in-depth evaluation, however, was not provided.

Run-Length Smoothing Algorithm (RLSA). It is typically unnecessary to perform processing on all pixels of the document image. For the top-down algorithms previously described, which use either maximal white space rectangles or projection profiles, the document image is usually reduced in size during a preprocessing stage. By reducing the size and complexity of the input image, both the efficiency and accuracy can be enhanced assuming that only insignificant data is reduced. For instance, when detecting entire columns of text, the spacing between individual characters, words, and lines is unnecessary. One way to reduce the amount of data is to use a run-length smoothing algorithm (RLSA) [66] which will be discussed further in the Bottom-up Physical Structure Analysis section. This method can merge characters into words, words into text lines, and text lines into paragraphs by “smearing” the text to join characters into blobs. This is done by inspecting white spaces between foreground pixels and, if their width is below some threshold, setting them to black.

Template Techniques. “Template” techniques which have been observed in the literature [67][68], are labeled as top-down even though they often rely on a combination of both logical and physical document structure analysis [47]. These methods require a significant amount of knowledge about the expected document structure on which they are trained and may not generalize well to new types of documents. An effective way in which document structure can be described is through the use of a Form Description Language (FDL) [67]. The basic concept of FDL is that both the logical and physical structures of a document can be described in terms of a set of rectangular regions. The FDL specifies how a document should be processed based upon various aspects of its physical layout. Systems which utilize an FDL typically operate on a limited assortment of document types, thus its use is very application specific.

Dengel et al. present a technique which they call “Discriminating Attribute Values in uncertain Object Sets (DAVOS) [68]. By “object sets”, Dengel et al. are referring to sets of regions on a document image along with their appropriate logical labels. The attributes (geometric features) of these objects may not be limited to single values but could cover a range of possible values and are thus considered as

“uncertain.” The DAVOS system analyzes business letters and builds a decision tree where each level corresponds with an increasing level of document type specificity. The terminals on the tree specify the entire logical layout of the document. Just as with FDL, the ability of the DAVOS system to generalize to new document types is limited. DAVOS was only tested on business letters and was evaluated against a bottom-up technique (which utilized merging of connected components) and shown to have similar but “more balanced” results (i.e. logical labeling errors were more distributed among the various labels).

Bottom-Up Physical Structure Analysis

While top-down approaches start with the complete document image, repeatedly splitting it into smaller regions, bottom-up approaches carry out the inverse operation. Starting with the document image's primitives (i.e. individual pixels, connected components, words, etc. depending upon the application) bottom-up techniques repeatedly merge smaller regions into larger ones. While allowing more flexibility over top-down techniques, bottom-up techniques often result in greedy over-segmentation of regions. This section will briefly review work that has been done for bottom-up techniques, starting with a discussion of connected component analysis techniques.

Connected Component Analysis. As discussed previously, connected components are sets of foreground pixels such that a four or eight-connected path exists between every pixel pair in the the set. While text usually consists of connected components with a relatively consistent size and spacing, graphics generally tend to consist of larger connected components with more sparsely distributed positions. By analyzing these spatial properties of connected components, it becomes possible to identify and group text and graphics separately. Connected component generation involves grouping all four or eight-connected foreground pixels together in the document image. The components are then grouped based upon their bounding box location. The output of a connected component (cc) generation algorithm is a list of cc's where each entry contains the bounding box coordinates, shape of the region, number of black pixels, an image of the region itself, etc. The cc's are typically sorted by their bounding box position, and can be then filtered based upon height and width to determine regions more likely to be text vs those which are more likely to be graphics [43].

An example of a bottom-up technique which utilizes connected component analysis is Bixler et al.'s text extraction algorithm [69]. Bixler demonstrates his algorithm by extracting and recognizing the text from a map as shown in Figure 23. His technique first uses a standard recursive (stack-based) flood fill algorithm in order to find the connected components [70]. After finding an initial starting foreground pixel, the flood fill algorithm can be described simply as follows: (1) If the current pixel is not foreground then return, (2) Set the current pixel to a replacement color in order to mark it as processed (3) Recurse to the function in each direction in turn (4) Return from the function. The aforementioned algorithm is then repeated for each unmarked foreground pixel of the image in turn, until all connected components are found and all marked pixels grouped into their constituent connect components are stored in memory, along with their bounding boxes, and any other relevant information.

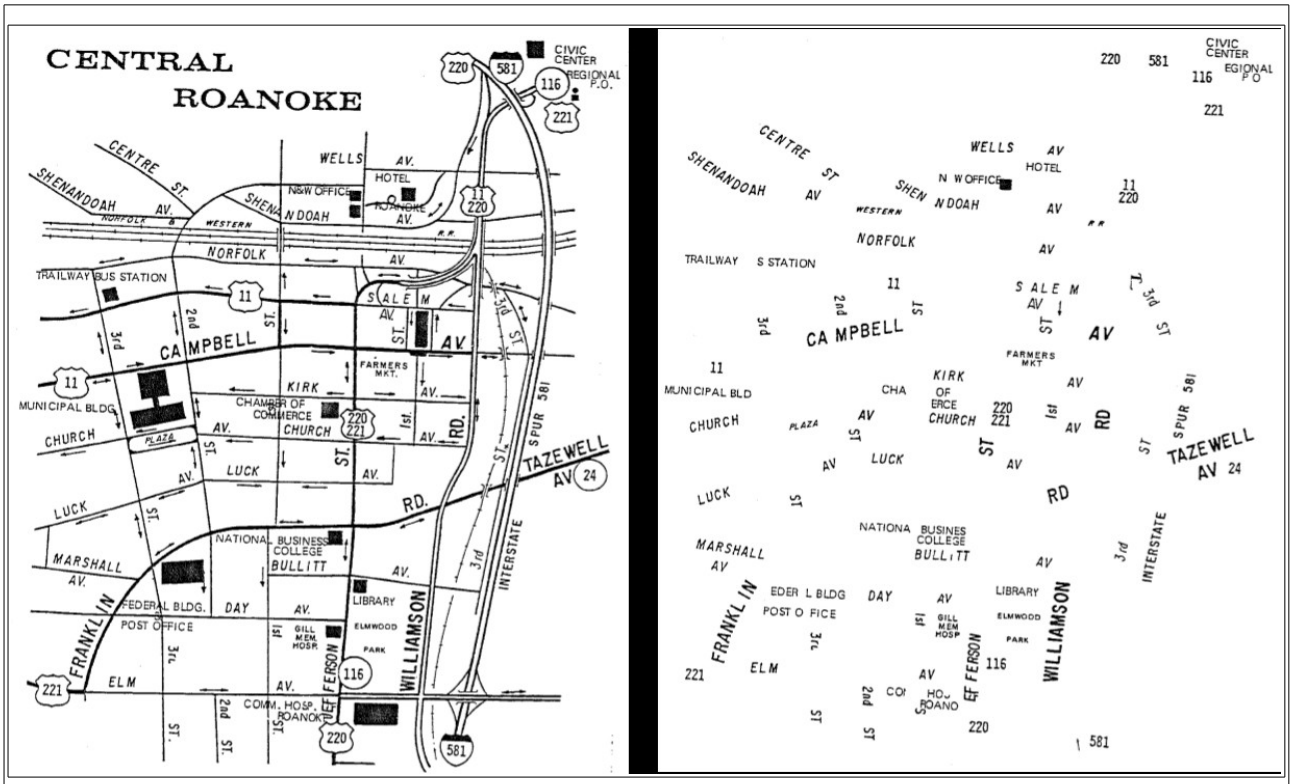


Figure 23: On the left is a map, and on the right is the map's extracted text. Notice there are some dependencies where the foreground text was confused with the imagery of the map. For instance one of the "f's" in the word "Post Office" is missing because it overlaps with a road.

With the connected components found, Bixler then determines which ones are text and which are graphics based on a simple height and width thresholding technique. Once the components have been segmented into text and graphics, those

identified as graphics are subtracted from the image to leave only the text. The resolution of the image is then reduced based upon the size of the character components. A connected component tracking algorithm is then utilized in order to find words which could be potentially in any direction (i.e. vertical, diagonal, horizontal, etc). The algorithm scans the reduced document image from left to right, top to bottom looking for a starting connected component, then does a nearest neighbor search in each direction to find the closest character. Information about the spacing and direction between the first two characters is then utilized to track the location of the next character until entire words are detected. The procedure is repeated for each unique starting point until all words are found. The technique achieved near perfect results for a complex map, with only those words which significantly overlapped graphics being missed.

Document Spectrum Analysis ("Docstrum"). The Document Spectrum (Docstrum) proposed by O'Gorman [71], is a representation of a document which describes global structure features and can be useful for page analysis. The technique takes the document's connected components and utilizes a k-nearest-neighbor clustering technique in order to segment the document into words, text lines, paragraphs, etc. As described in [71], the algorithm recognizes five nearest neighbors for each connected component, where closeness is measured by Euclidean distance in the image. Each nearest neighbor pair is described by a 2-tuple, (d, θ), which is the distance and angle between the centroids of the two connected components. The "Docstrum" is the plot of (d, θ) for all connected components in the image as illustrated by Figure 24. The text's spacing between characters and words as well as the line angles can be estimated by summing up the distance and angle values in the docstrum plot. The distances and angles are converted to respective histogram representations. The nearest neighbor angle histogram is smoothed and the peak found. The angle of the peak value gives a rough estimate of the text line orientation. This rough estimate is then used to determine intra-line and inter-line spacing by analyzing histograms of the nearest-neighbor distance values. The histogram for inra-line spacing filters out all distance values that are not within a tolerable range of the textline orientation estimate. This histogram thus represents the distribution of inter-character and word spacing within each text line. The second distance histogram filters out all values outside of a tolerable range of the textline orientation estimate's perpendicular. This histogram, therefore, represents the distribution of the document's inter-line spacing.

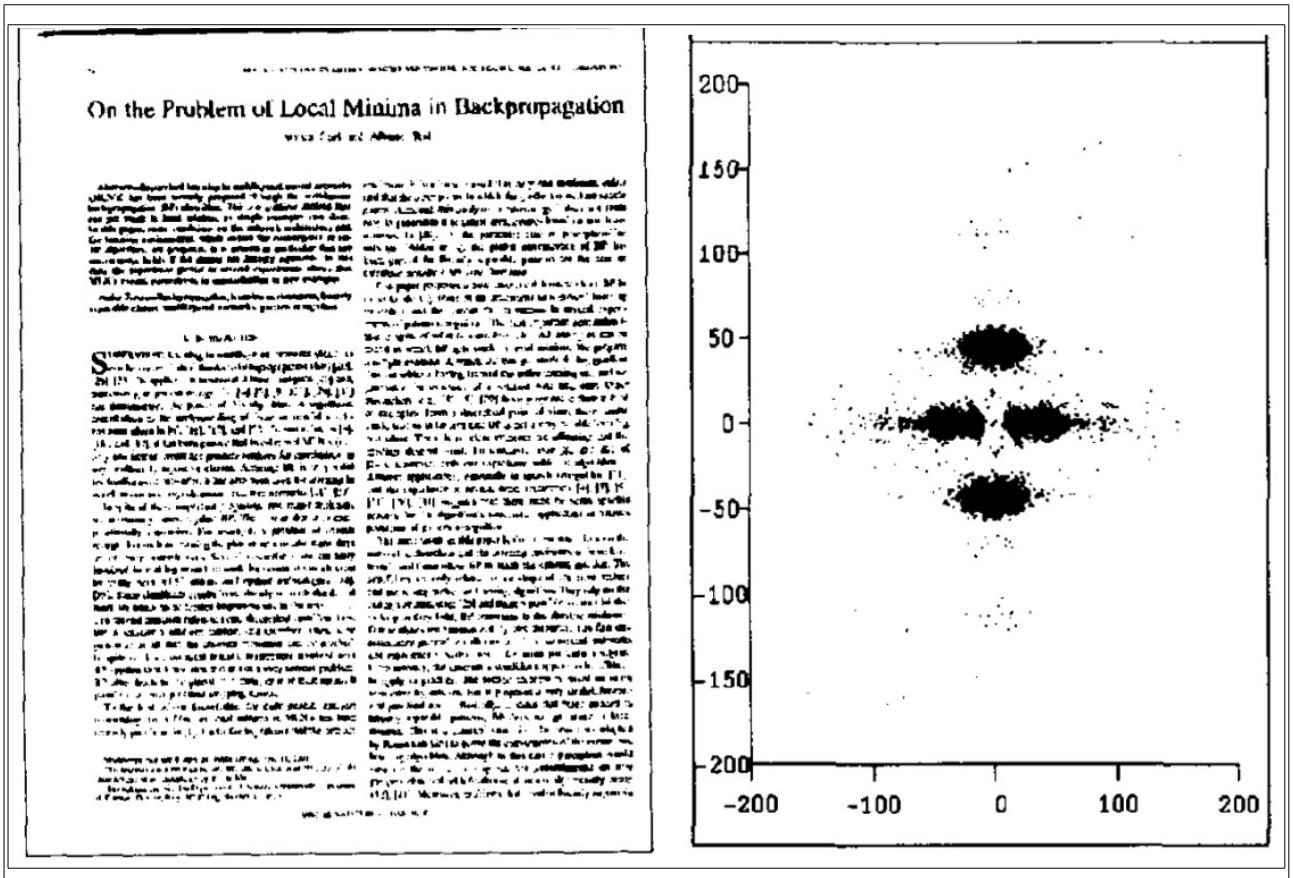


Figure 24: On the left is a document image and on the right is its corresponding Document Spectrum representation [71].

Nearest neighbors on each line are merged into words and then a regression fit is made to the centroids of the words in order to locate text lines. A straight line is fitted to the centroids in each group by minimizing the sum of square errors between centroids and the line. From these text lines a final estimate is made of the page's skew. An issue with this method is that text line descenders and noise could reduce the accuracy of the initial estimate and cause problems with reaching the right conclusions. It is important to have the correct threshold values and to smooth the histograms appropriately in order to get successful results. After the text lines are estimated, larger structures (like paragraphs or other text blocks) are then detected. The blocking technique examines pairs of text lines to determine whether or not they meet certain criteria to be considered part of the same text block. If the two lines are approximately parallel, close enough in perpendicular distance, and/or horizontally overlap to some degree then they are said to meet the criteria of belonging to the same block.

One of the benefits of this algorithm is that it does not assume that each component of the document has the same skew angle. Thus it is possible to indiscriminately segment lines and/or blocks of text in any direction. This may be useful for a variety of circumstances including analysis of magazines or journals with sporadically appearing vertical text, scans of several credit cards or business cards each on the same page but at arbitrary angles, maps with text overlayed over imagery in arbitrary directions, etc. The technique was tested on hundreds of scanned journal pages, however no comprehensive performance evaluation is given.

Voronoi Diagram. Given a set of points and a subset of these points called sites (or generators), the Voronoi diagram is the partition of the entire set into convex cells, such that each cell is the region consisting of all points that are closer to a particular site than to any other. Voronoi diagrams are among the most fundamental and well-studied objects in computational geometry [72]. An *ordinary* Voronoi diagram, as illustrated by Figure 25 [73], is one which uses Euclidean distance as its metric and can be described as the set of Voronoi regions which correspond to the convex shapes created by the partition.

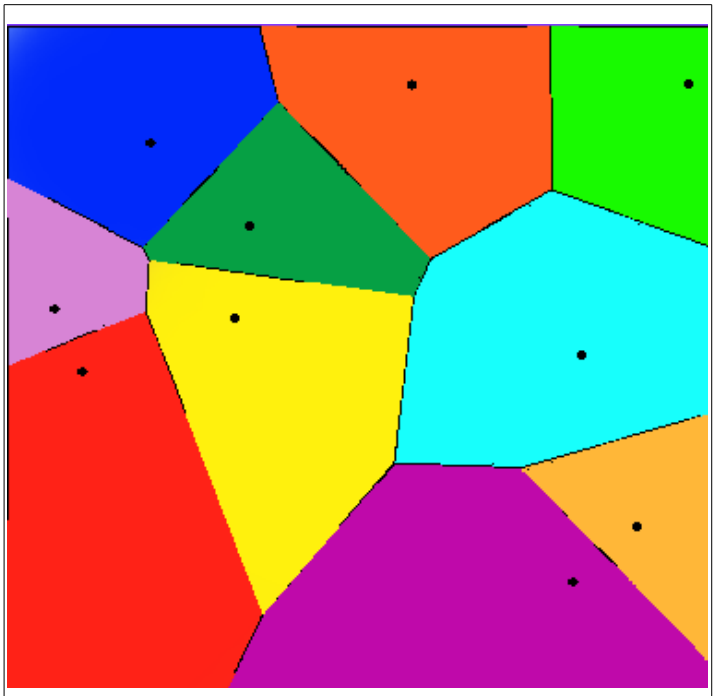


Figure 25: Illustration of an ordinary Voronoi diagram [73].

An *area* Voronoi diagram is a generalization of the *ordinary* Voronoi diagram depicted by Figure 25 which uses the Euclidean distance between the areas of connected components as a metric rather than the distance between points. The *area* Voronoi diagram for a document image can be found by the following procedure: (1) Sub-sample every connected component in the image such that all that remains is a subset of the points on the outter edge; (2) Generate an *ordinary* Voronoi diagram using this subset of points in the image; (3) Remove all edges of the Voronoi diagram which both belong to points of the same connected component. This process is illustrated by Figure 26 [74].

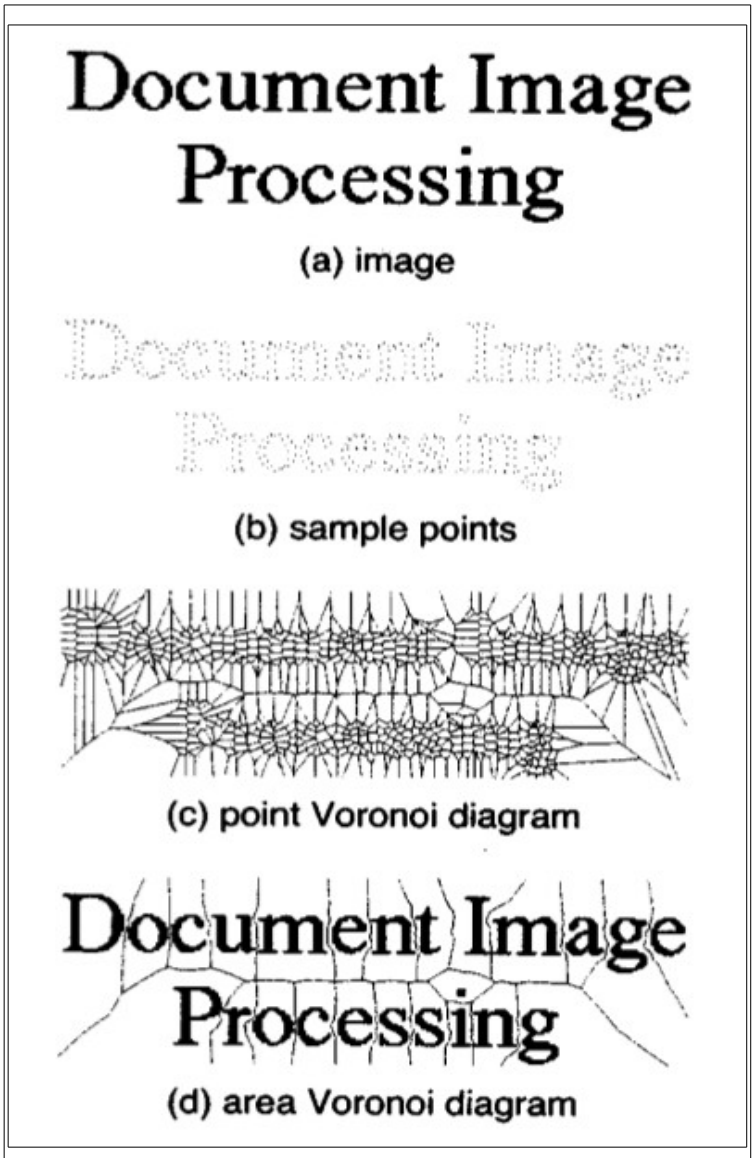


Figure 26: The process of segmenting an image into an area Voronoi diagram as illustrated by Kise et. al.

Kise et. al [74] formulate the problem of physical page as that of determining which edges of a document's *area* Voronoi diagram best represent the boundaries of document components. By analyzing various features in the document image, superfluous edges of the area Voronoi diagram can be removed, thereby leaving only the edges corresponding to document boundaries. Superfluous edges would, for instance, correspond to the space between characters, words, text lines, etc. when a division of the page into separate paragraphs, columns, imagery, title, etc. is required. For each edge, all of its line segments are evaluated in order to determine the minimum distance between the two points on the connected component which were used to generate the Voronoi line segments in the first place. If this minimum distance is below a given threshold for any line segment of the edge, then the entire edge removed. Likewise, the area of connected components are divided by these edges are compared and if the distance between the connected components is small enough in relation to the area ratio of the two connected components, then the corresponding edge is removed.

Kise et. al evaluate their algorithm on 16 document images at two resolutions, 90 DPI and 300 DPI having a non-manhattan layout each at 4 different skew angles to test for robustness (thus a total of 128 with non-Manhattan layout when counting the resolutions and skew). In order to test the applicability with Manhattan layouts, the algorithm was also evaluated on 98 images from the University of Washington database (UW1) all at 300 DPI. In evaluating the algorithm on these datasets, the percentage of the “body” text, “auxiliary” text, and “non-text” document zones which were over and under fragmented is evaluated respectively. The algorithm performed best on the body text of the non-Manhattan documents scanned at higher resolution where only 2.1% of zones were over-fragmented and only .4% under-fragmented. The algorithm faired poorly for the segmentation of non-text zones of all document types, but especially poorly for Manhattan documents where it resulted in a 98% over-fragmentation rate.

Run Length Smearing Algorithm (RLSA). Proposed originally in 1974 by Johnston [75] in order to separate text blocks from graphics, the Run Length Smearing Algorithm (RLSA) has been frequently used to obtain basic features for document analysis [43]. RLSA, in its most basic form, transforms a binary image as follows: (1) For each background pixel, if the number of neighboring foreground pixels is above a certain threshold, then the pixel is changed to foreground; (2) All foreground pixels are left unchanged. When applied horizontally or vertically to the rows or columns of

an image respectively, RLSA has the effect of linking together neighboring background pixels that are separated by a number of pixels below the given threshold (illustrated by Figure 27 [66]). With an appropriate choice of threshold, it is possible for the linked areas to correspond with separate document zones. The threshold is typically set based upon the character height, gap between words, and interline spacing [43].

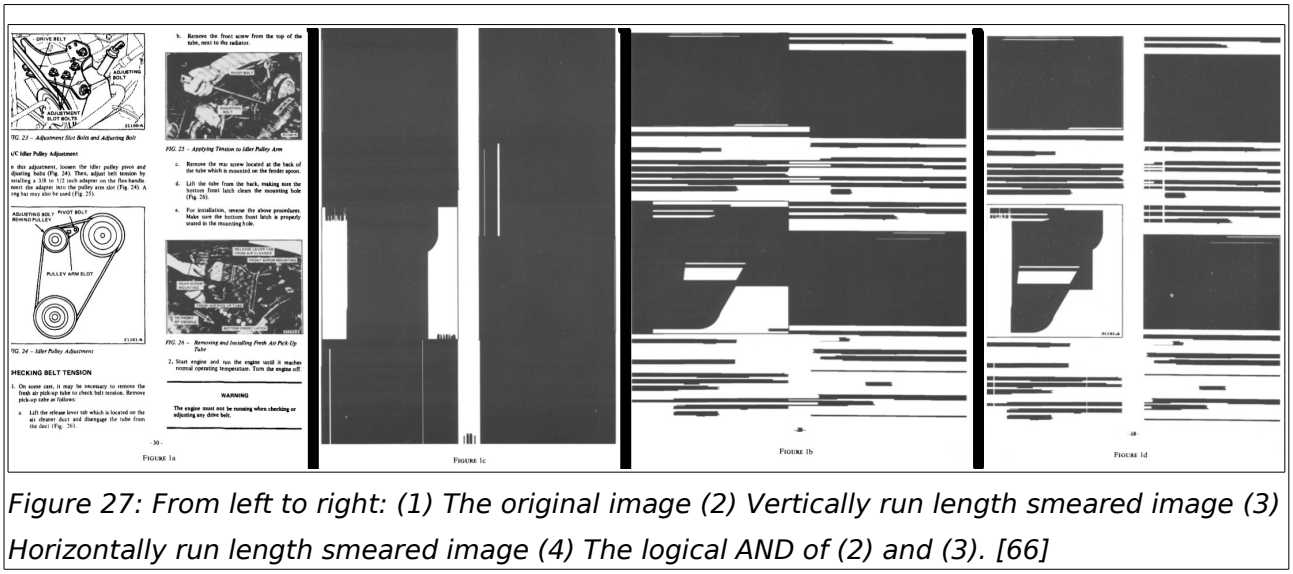


Figure 27: From left to right: (1) The original image (2) Vertically run length smeared image (3) Horizontally run length smeared image (4) The logical AND of (2) and (3). [66]

Multiresolution Morophology. Bloomberg [76] discusses an approach to document image analysis which uses morphological operations

closing: dilation followed by erosion by the SE

opening: erosion followed by dilation by the SE

...confused by intended meaning of subsampling.. try reviewing dip book???

threshold reduction -> combination of threshold convolution (either closing, dilation, opening, or erosion operations) and subsampling (reduction in resolution). This yields similar results to doing threshold convolution with a large structuring element to locate non-text regions but at reduced computational complexity.

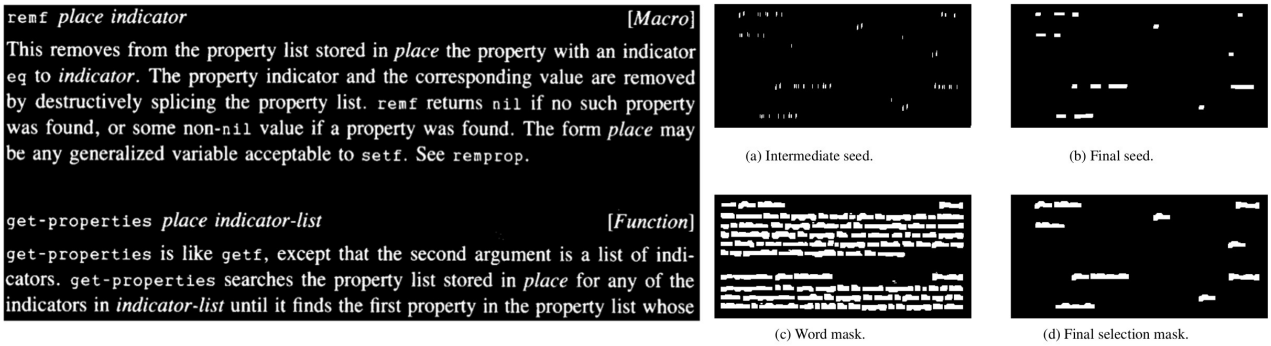
((A brute-force morphological approach might be to close the image

with sufficiently large SEs to solidify the halftone parts, and then open the image with even larger

SEs to remove the (somewhat blocked up but smaller) text parts.)

briefly explain what opening and closing is... (refer reader to reference for more info on dilation/erosion.. show image to illustrate finding of italics using morphological operations... multiresolution morphology.... this could be helpful for seed location...

uses morphological operations on the document image at various resolutions to determine identify font style for each word. Class labels include bold, italic, and normal. The method employs a small vertical dilation followed by a close open sequence t o remove noise followed by a hit and miss transform to identify seed points of characters in the italic class or bold class. Then the words which are in italic or bold can be delineated by conditionally dilating the seed with a precalculated word segmentation mask. No accuracy performance results are given.



(a) Intermediate seed.

(b) Final seed.

(c) Word mask.

(d) Final selection mask.

just list these last two at the intro.....

Neural networks [23].

C.L. Tan, Z. Zhang Text block segmentation using

pyramid structure. SPIE Document Recognition and

Retrieval, Vol. 8, January 24-25, 2001, San Jose, USA,

pp. 297-306.

communication theory approach to page segmentation [**T. A. Tokuyasu and P. A. Chou, “Turbo recognition: a statistical approach to layout analysis,” in Proceedings of SPIE Conference on Document Recognition and Retrieval, (San Jose, CA), January 2001.**]. regular grammars are used to describe the structure of document page images in terms of axis-parallel rectangles obtined by subdividing the image vertically and horizontally, and they used a turbo decoding approach to estimate the 2d image from the observations. very limited experimental verification provided. <--- just reference at the end as an example of a communications theoretical approach.....

(1994 - document processing for automatic knowledge aquisition)

Neighborhood-Line density [19]

O. Iwaki, H. Kida and H. Arakawa. A character / graphics segmentation method using neighborhood line density. Trans. of Institute of Electronics and Communication Engineers of Japan, 1985, Part D J68D, 4, pp. 821 828.

[49] - **O. Iwaki, H . Kida, and H. Arakawa, "A character/graphic segmenta tion method using neighbourhood hne density," Trans. In st . Electron. CommUll. Engineers of Japan , Part D, vol. J68D, no. 4, pp. 82 1 -828, 1 985,**

[50]

O. Iwaki, H. Kida, and H . Arakawa, "A segmentation method based on office document hierarchical struclure," in Proc. IEEE Int. Conf Syst. Man. Cybernetics, Alexandria, VA, Oct. 1987, pp. 759-763.

Hybrid Physical Structure Analysis

..need to abbreviate significantly...

2007 - Document Structure and Layout Analysis - definition of layout analysis::::The process of document layout analysis decomposes a document image into a hierarchy of maximally homogeneous regions, where each region is repeatedly segmented into maximal sub-regions of a specific type. :::::: very good description of the different types of top-down or bottom-up algorithms and why they fall under each category. This is very good stuff. It's basically one giant lit review!!!!!!!!!!!!!!!

Two notable open source systems which perform document layout analysis are Tesseract and OCR-Opus. Both systems utilize a combination of bottom-up and top-down physical layout analysis techniques

2008---- "The OCRopus open source OCR system,"

in [Document Recognition

------- discusses ocropus algorithms "it is currently the best available open source OCR system for English, when using the combination of RAST-based layout analysis with the Tesseract text line recognizer. "

2009 - Document Image Analysis with OCRopus

^ This makes me want to look into ocropus a little more. Talks about how pluggable the architecture of it is. But not very much about how things are actually implemented....

- Gaussian pyramids method. [77] (steerable pyramid....)

2003 “document analysis structure algorithms” survey..

hybrid

split-and-merge strategy by pavlidis and zhou [T. Pavlidis and J. Zhou, “Page segmentation and classification,” Graphical Models and Image Processing 54, pp. 484–496, 1992.]

texture:::::

D. Chetverikov, J. Liang, J. Komuves, R.M. Haralick.

Zone classification using texture features. Proc. Of Intl.

Conf. on Pattern Recognition, Vol. 3, 1996, pp. 676-680.

- 2-D Gabor filters method <- talk about this in 1996 survey!! [78] <- this is hybrid since it uses top-down to filter the image then uses clustering on the features of the filtered image...

(**1994 - Document Image Understanding: Geometric and Logical Layout)**

Work 4::

Lebourgeois et.al. (1992) sample the document image by a factor of 8 vertically and 3 horizontally. Each pixel on the sampled image corresponds to an 8x3 window on the original image. If any pixel on the 8x3 window of the original image is a binary one then the sampled image has a binary one in the corresponding pixel position. Then the sampled image is dilated by a horizontal structuring element to effectively smear adjacent characters into one another. Each connected component is then characterized by its bounding rectangle and the mean horizontal length of the black runs. Connected components having a vertical height within given bounds and mean

horizontal run length within given bounds are then labeled as a text lines and outside the given bounds are labeled as a nontext lines. Components assigned as text regions are then vertically merged into larger blocks using rules taking into account alignment. **Blocks are also subdivided to separate their horizontal peninsulas.** N o measure of performance is given but an indication that the method needs improvement was stated.

Work 6:::;;

Saitoh and Pavlidis (1992) proceed sampling by 8 vertically and 4 horizontally and then extracting the connected components. They then classify each component into text, text o r noise, diagram or table, halftone image, horizontal separator, or vertacal separator, using block attributes such as block height, height to width ratio, and connectivity features of the line adjacency raph, and whether there are vertical or horizontal rufings. Page rotation skew is estimated from a least squares line fit to the the center points of blobs belonging to the same block. Blocks are subdivided based on the vertical distance between lines in a block, and the height of the lines in a block. The technique was tried on 52 Japanese documents and 21 English documents. No quantitative measure of performance was given.

Work 11::

O'Gorman (1992) discusses what he calls the docstrum technique for determining geometric page layout. This technique involves computing the k-nearest neighbors for each of the black connected components of the page. Each pair of nearest neighbors has an associated distance and angle. By cluster the components using the distance and angle features, the geometric regions of a page layout can be determined.

Work 13::

Hirayama (1993) develops a technique for determining the geometric layout structure of a document which begins by merging character strings into text groups. Border

lines of blocks are determined by linking edges of text groups. Then blocks which have be oversegmented are merged and a projection profile method is applied to the resulting blocks to differentiate text areas from figure areas. Hirayama reports that on a data set of 61 pages of Japanese technical papers and magazines 93.3%of the text areas and 93.2% of the figure areas were correctly detected.

Work 8::

Pavlidis and Zhou (1991) determine the geometric page layout by analyzing the white areas of a page by computing the vertical projection and looking for long white intervals from the projections. Then the column intervals are converted into column blocks, merging small blocks into larger blocks. Blocks are clustered according to their alignments and the rotation angle estimated for each cluster. The column blocks are then outlined. Finally, each block is labeled as text or nontext using features such as ratio of the mean length of black intervals to the mean length of white intervals, the number of black intervals over a certain length, and the total number of intervals. No performance results are given.

Hybrid

2009-Hybrid Page Layout Analysis via Tab-Stop Detection

^ THIS IS THE TESSERACT ARTICLE.. i've already read this... might be worth a breif reskimming with my newer perspective after reading so many other articles...

Constrained Text-Line Extraction by T. Breuel [Bre02c] <-???? I'd guess this one to be hybrid....

1993 - hybrid segmentation method.pdf

Document Logical Structure Analysis

2002 Structural Extraction from Visual Layout of Documents

^ brings up pdf extraction……

2008 Dolores: An Interactive and Class-Free Approach for Document Logical Restructuring

^ good info on PDF!!!!!

2008-Geometric Layout Analysis of Scanned Documents

2010-Document Image Segmentat using Discriminative Learning over Connected Com

^ Here we train a self-tunable multi-layer perceptron (MLP) classifier for distinguishing between text and non-text connected components using shape and context information as a feature vector. Experimental results prove the effectiveness of our proposed algorithm. We have evaluated our method on subset of UW-III, ICDAR 2009 page segmentation competition test images and circuit diagrams datasets and compared its results with the state-of-the-art leptonica's 1 page segmentation algorithm.

---------->by leptonica are they referring to tesseract????????!!!!!!!!!!!1 No this is just for segmenting text and half-tone components…… has nothing to do with math detection or other zone classification it seems….

2008-Geometric Layout Analysis of Scanned Documents

2011-Boosting based text and non-text region classification

^ just distinguishes text from non-text (nothing about further zone classification it seems) does a good job though (99.5% accuracy for their data). Used uwIII dataset and also their own indpendent one (had to train the classifier differently for each).... this article uses a top-down approach (x-y cut)...

2012 - Ensemble methods with simple features for document zone

^ same guys as 2011 boosting article!!!! Talks a lot about different classifiers and comparison... less focus on segmentation more on simply classifying the zones... this relies on correct page segmentation in a separate step and seems to be a weak design approach to me....

> **Very important::::::::::::Discusses how Ocropus does layout analaysis!!!!!!!!!! lassification ------- focus on classifying blocks after they have already been extracted. We present a comparative evaluation of three ensemble based classification algorithms (boosting, bagging, and combined model trees) in addition to other known learning algorithms. Experimental results are demonstrated for a set of 36503 zones extracted from 416 document images which were randomly selected from the tobacco legacy document collection. The results obtained verify the robustness and effectiveness of the proposed set of features in comparison to the commonly used Ocropus recognition features. When used in conjunction with the Ocropus feature set, we further improve the performance of the block classification system to obtain a classification accuracy of 99.21%.**

2011 - Page layout analysis and classification for complex scanned documents

^ detects text, photo, and strong line/edge regions... first module uses wavelet analysis, second uses vector projections and then markov random field based on MAP optimization with ICM (iterated conditional mode), third module uses houghe transform...

2007 - DOCUMENT IMAGE ZONE CLASSIFICATION

A Simple High-Performance Approach

2008-Structure Extraction in Printed Documents Using Neural Approaches

^ mainly talks about logical analysis..... LLOOK INTO THIS FOR NEURAL NET APPROACHES..... this is good for lit review on nueral net approaches. Not too sure about how good their results were however. I don't see any mention of equation detection anywhere.. and they have it set up so its kind of hard to see what their results are.. i mean you really have to dig for it... Will relook at it after i cateogize all my articles for organizatiuon...

2005 Document zone content classification and its performance evaluation

^ this one looks really good...even talks about math zones!!!!!!!!!!!!!!! A+... also has a very nice but brief literature review!!!!!!!!!!!!!!!!!!!!!!!!!!!!1 (not covered by surveys, but the concept is)

> This is a really good article::::::::::::::
>
> **There is a constant interest in the document image analysis field on document zone content classification problem. However, most of common approaches focus on specific type zone extraction and recognition. For example, Xiao and Yan [9] worked on text region extraction problem, Zanibbi et al. [10] on Mathematics Expression**

recognition, Hu et al. [3] on table extraction problem, Chen et al. [11] and Pham [12] on logo detection, Li et al. [13] on image (halftone) extraction problem, Futrelle et al. [14] on diagram (drawing) extract and classification problem, etc. To the best of our knowledge, our group is the first group who systematically study the zone content classification and conduct experiments on a large data set, UW Document Image Database III [8]

1994 - document processing for automatic knowledge aquisition

according to the international standard iso ISO/IEC 8613-1:1994 -> logical structure is the result of dividing and subdividing the content of a document into increasingly smaller parts on the basis of the human-pereptible meaing of the content- for example, into chapters, sections, subsections, and paragraphs.

Logical object - an element of the specific logical structure of a document. Logical structure can be represented in a tree form... (is there an example thats any good available of this??????????) see 1996- using image domain knowledge -> [79]

2007 – a survey of document image classification ----

very detailed.....explores classification of entire pages, emphasizes algorithms that don't use ocr results, explores algorithms that classify individual blocks.... good

Logical Layout Analysis

Should this be divided into separate sections for page classification and zone classification??? depends on what I have to write about.. if enough content on page classification then use, if not then no..... also consider type-specific.... but really i would want to focus on that in the next part. Type-specific is the focus of this thesis because it is entirely necessary, since there is no “one fits all” solution for every document analysis problem...

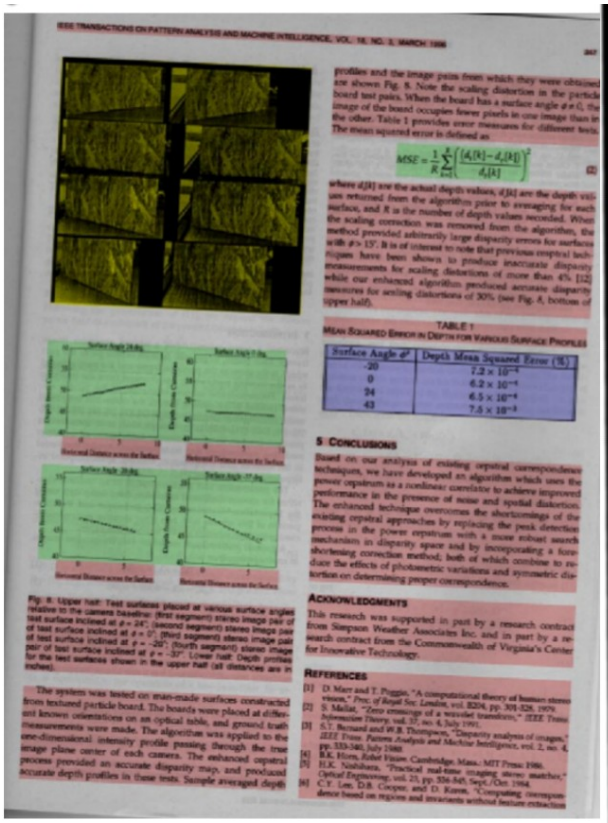
The logical layout analysis assigns a meaning to the regions identified by the physical layout analysis. The physical and logical analysis can be performed together [nagy non-survey article!!!] so as to assign a meaning to blocks during their segmentation. In most cases this is not feasible, since the class can be defined only after analyzing the region position with respect to other parts of the page (or even after the reading of its content).

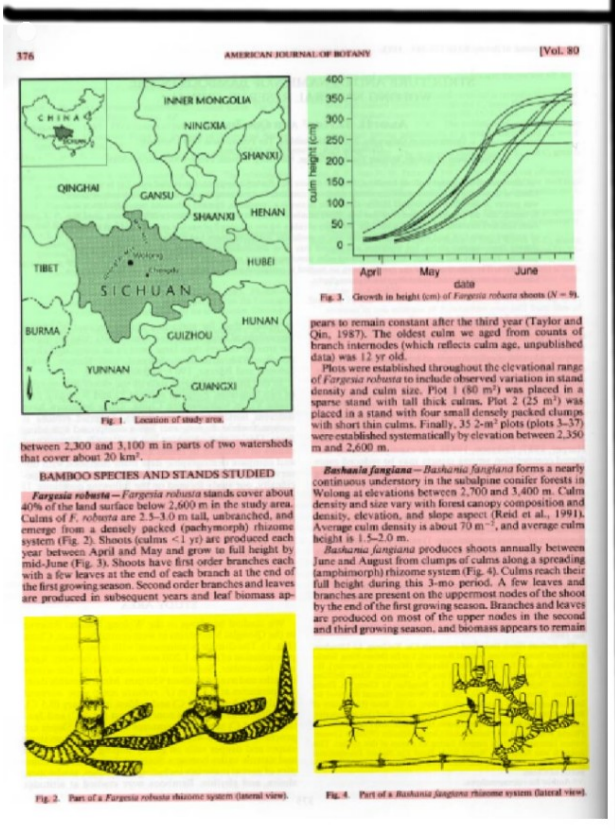
need to address how it is often the case that you must transition between logical and physical frequently to reach the correct conclusion.... <- bring in other resources in here to back this up <------ from the 2007 article with the pictures....The analysis of logical structure of a document is usually performed on the results of the layout analysis stage. However, in many complex documents, layout analysis would require some of the logical information about the regions to perform correct segmentation.

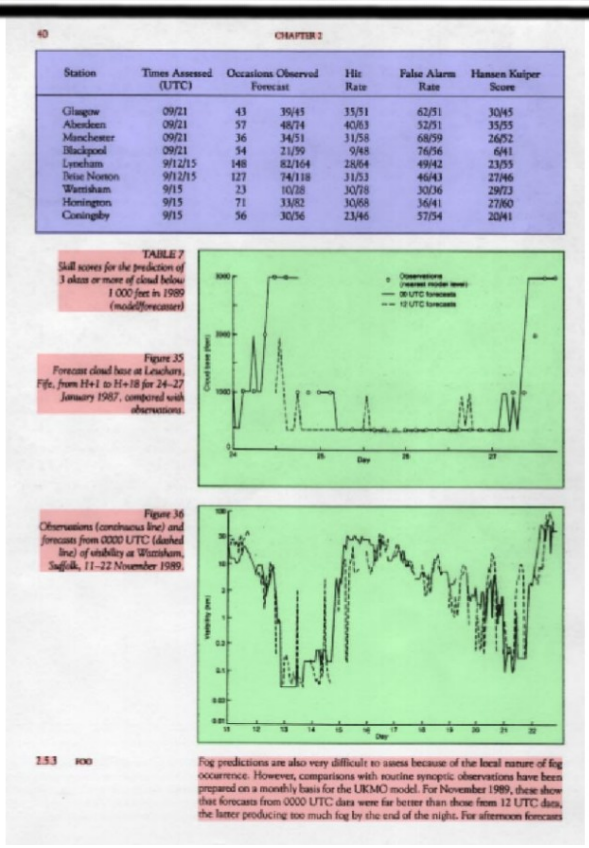
"seperate and conquer"(i.e. rule based learning) vs "divide and conquer" (i.e. decision trees) is a good paradigm to go into detail on for thesis

1998 - Document Representation and Its Application to Page Decomp

^ this has some good imagery.. has a great survey, very consise for both physical and logical!!!! recognizes some mathematical formulas as drawings and others as text...







....forgot which article this is but won't be hard to find it...

type specific is just focused on a single type ... for instance my research is just focused on finding math zones... this is how logical layout analysis is subdivided.....

Broadly speaking, content analysis algorithms can be classified as one of three main approaches – (1) type-specific detection, (2) page classification and (3) zone classification. Type-specific approaches, emphasize finding specific types of zones, such as text regions [9], logos [2], mathematical expressions [10] and tables [4]. Page classification approaches, assume the content of the entire page is of a single type (e.g. title page or index page) and a classifier is used to determine the page content [1][3]. Finally, zone classification approaches assume that the page is segmented into zones with independent content types. Low level image features are extracted from each zone and a statistical classifier is used to label the different zones into one of possible content types (e.g. text, math, etc.) [7].

!!!!!!!!!!!!!!!!1This paper uses SVM's!!!!!!!!!!!!!!!!!!!!!!!!!!

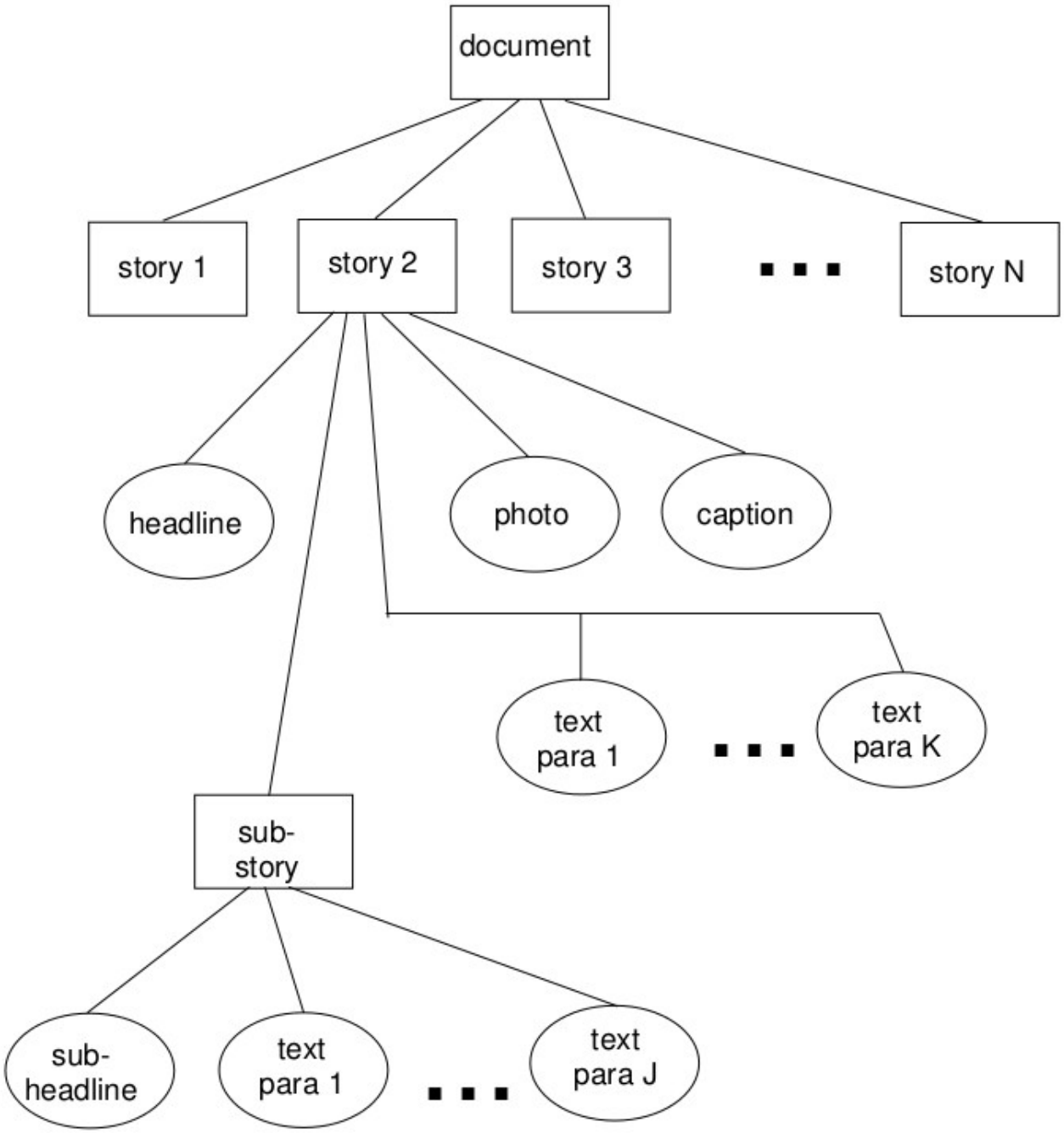
!!!they use the UW dataset.....

Break this down into decision tree, rule-based, etc.....

1.

1996 - using domain knowledge (stole an image from this one)

good depiction of logical structure in tree format....



(1994 - document processing for automatic knowledge aquisition)

---Form Definition Language (FDL):

Since logical structure can correspond to a variety of geometric structures, the generation of logical structure from the geometric structure is difficult. One of the

promising solutions to this problem is use of formatting knowledge. For documents of different kinds and of different languages the formatting rules will be different. For a specific kind of document, once its formatting knowledge is acquired, its logical structure can be deduced. This is often carried out in the literature using rule-based approaches derived from statistics on a large database of documents of particular types.

FDL::

Another method for detecting the logical structure of a document makes use of the knowledge rules represented by a form definition language (FDL). The basic concept of the FDL is that both the geometric and logical structures of a document can be described in terms of a set of rectangular regions. See "A knowledge-based segmentation method for document understanding" for more details....

A method is here discussed which uses statistical models of various document types in order to register an unknown page to a particular document form type for recognition.

Reference [40) proposed a top-down document analysis method where the document layout knowledge is effectively utilized to parse the two-dimensional physical document structure. It devised a knowledge representation called form definition language (FDL) to describe the generic layout structure of the document.

The structure can be represented in terms of rectangular regions, each of which can be recursively defined in terms of smaller regions. An example is given in Fig. 1 0. These generic descriptions are then matched to the preprocessed input document images . Various document blocks can be located. This image analysis method is powerful, but it is complicated to implement. [321 developed a simplified version of FDL so that it may be implemented more easily. This method will be described in greater detail in the following section.

::::::::::::::::::::::::::::::::::::

Document understanding can be viewed as a transformation that converts a geometric structure into logical structure [104]. A document has an obvious

hierarchical geometric structure, represented by a tree as shown in Fig. 2(b). Also, the logical structure of a document is also represented by a tree that is illustrated in Fig. 2(c). This method defines document understanding as the transformation of a geometric structure tree into a logical structure tree. In this example, three kinds of blocks are defined: H (head), B (body) , and S (either body or head). During the transformation, a label is attached to each node. Labels include title, abstract, subtitle, paragraph, header, footnote, page n umber, and caption.

^ welll... duhhhh...

Generation of logical structure from geometric structure involves labeling the geometrically separated objects into sections classified by their purpose (ie paragraph, column, title, etc)

2003 “document analysis structure algorithms” survey..

Document logical strucutre representations and analysis algorithms:

Logical labels either derived from a set of rules

([**J. Kreich, A. Luhn, and G. Maderlechner, “An experimental environment for model based document analysis,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 50–58, (Saint-Malo, France), September 1991.**]

[**C. C. Lin, Y. Niwa, and S. Narita, “Logical structure analysis of book document images using contents information,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 1048–1054, (Ulm, Germany), August 1997.**]

[**Y. Ishitani, “Logical structure analysis of document images based on emergent computation,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 189–192, (Bangalore, India), September 1999.**]

[**J. Kim, D. X. Le, and G. R. Thoma, "Automated labeling in document images," in Proceedings of SPIE Conference on Document Recognition and Retrieval VIII, pp. 111–122, (San Jose, CA), January 2001**.]

[**K. Summers, "Near-wordless document structure classification," in Proceedings of International Conference on Document Analysis and Recognition, pp. 462–465, (Montreal, Canada), August 1995.**].

relations between logical components are expressed through trees that are derived either from a set of rules

[**D. Niyogi and S. N. Srihari, "Knowledge-based derivation of document logical structure," in Proceedings of International Conference on Document Analysis and Recognition, pp. 472–475, (Montreal, Canada), August 1995.**]

[**S. Tsujimoto and H. Asada, "Understanding multi-articled documents," in Proceedings of International Conference on Pattern Recognition, pp. 551–556, (Atlantic City, NJ), June 1990.**]

[**A. Yamashita, T. Amano, I. Takahashi, and K. Toyokawa, "A model based layout understanding method for the document recognition system," in Proceedings of International Conference on Document Analysis and Recognition, pp. 130–138, (Saint-Malo, France), September 1991.**]

[**J. L. Fisher, "Logical structure descriptions of segmented document images," in Proceedings of International Conference on Document Analysis and Recognition, pp. 302–310, (Saint-Malo, France), September 1991.**]

[**D. Derrien-Peden, "Frame-based system for macro-typographical structure analysis in scientific papers," in Proceedings of International Conference on Document Analysis and Recognition, pp. 311–319, (Saint-Malo, France), September 1991.**]

or from formal grammars

[**M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan, "Syntactic segmentation and labeling of digitized pages from technical journals," IEEE Transactions on Pattern Analysis and Machine Intelligence 15, pp. 737–747, 1993.**]

[**R. Ingold and D. Armangil, “A top-down document analysis method for logical structure recognition,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 41–49, (Saint-Malo, France), September 1991.**]

[**A. Conway, “Page grammars and page parsing: A syntatic approach to document layout recognition,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 761–764, (Tsukuba Science City, Japan), October 1993.**]

[**Y. Tateisi and N. Itoh, “Using stochastic syntactic analysis for extracting a logical structure from a document image,” in Proceedings of International Conference on Pattern Recognition, pp. 391–394, (Jerusalem, Israel), October 1994.**]

the document is regarded as a sentence which can be either a string of logical labels or a string of observed features of document physical components..

2003 “document analysis structure algorithms” survey..

deterministic parsing algorithms

[**M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan, “Syntactic segmentation and labeling of digitized pages from technical journals,” IEEE Transactions on Pattern Analysis and Machine Intelligence 15, pp. 737–747, 1993.**]

[**R. Ingold and D. Armangil, “A top-down document analysis method for logical structure recognition,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 41–49, (Saint-Malo, France), September 1991.**]

[**A. Conway, “Page grammars and page parsing: A syntatic approach to document layout recognition,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 761–764, (Tsukuba Science City, Japan), October 1993.**]

augmented parsing algorithm with cost attributes [**Y. Tateisi and N. Itoh, “Using stochastic syntactic analysis for extracting a logical structure from a document image,” in Proceedings of International Conference on Pattern Recognition, pp. 391–394, (Jerusalem, Israel), October 1994.]**

2001 ML Textbook (Duda, Hart)

for growing decision trees, CART (classification and regression trees)

p.429 of text::::

they start talking about rule-based learning on p 429 (Grammatical Inference)

rule-based methods have the advantage that they are easily interpretted and can be used in database applications where information is encoded in relations. A drawback

is that there is no natural notion of probability and it is somewhat difficult, therefore, to use rules when there is high noise and a large Bayes error.It is often

very difficult to accurately evaluate the predicates

p.433-434!!!

seperate and conquer technique: (aka sequential covering)

From a set of positive and negative examples is to learn a single rule, delete the examples that it explains, and iterate.

--- the following is basically copied right out of the book.....

This is distinguished from decision tree methods (which after being pruned can be converted into rules) in that this approach can learn sets of first order rules containing variables. This leads toa disjunctive set of rules that "cover" the training data. After such training it is traditional to simplify the resulting logical rule by means of standard logical methods. The designer must specify predicates and functions based on prior knowledge of the problem domain. The algorithm begins by considering the most general rules using these predicates and functions, and it finds the "best" simple rule. Here "best" means that the rule describes the largest number of training examples. Then the algorithm searches among all refinements of the best rule, choosing the refinement that too is "best". This process is iterated until no more refinements can be

added, or when the number of items described is maximum. In this way a signle, covering algorithm iterates this process and returns a set of rules.

First order learning::::

i think that this is referring to first-order logic (learning which uses first-order logic in some capacity)

http://en.wikipedia.org/wiki/First-order_logic

First order logic is similar to propositional logic, but involves quantifiers (i.e. for every literal some condition is satisfied)

---

what is a "clause", "literal", machine learning? --- these are all terms in mathematical logic literal - simplest well-formed logic (for instance a bool variable)

clause - a finite disjunction of literals (i.e. lit1 V lit2 V lit3 etc...)

First Order Inductive Learner

http://en.wikipedia.org/wiki/First_Order_Inductive_Learner

Developed in 1990 by Ross Quinlan,[1] FOIL learns function-free Horn clauses, a subset of first-order predicate calculus.

Horn clause - a clause with only one literal that isn't negated.

Uses "seperate and conquer" rather than "divide and conquer"

"seperate and conquer" basically, a separate-and-conquer algorithm searches for a rule that explains a part of its training instances, separates these examples, and recursively conquers the remaining examples by learning more rules until no examples remain

"divide and conquer" - In computer science, divide and conquer (D&C) is an important algorithm design paradigm based on multi-branched recursion.

A divide and conquer algorithm works by recursively breaking down a problem into two or more sub-problems of the same (or related) type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

From "1999-Seperate and Conquer Rule Learning"

seperate and conquer - also called "sequential covering" or just "covering"

concept-learning problem - "the search for and listing of attributes that can be used to distinguish exemplars from non exemplars of various categories."

concept-learning problem can be tackled with divide and conquer techniques (examples of these are decision tree techniques which use "divide and conquer" strategies.

Much of the popularity of decision tree

learning stems from its efficiency in learning and classification (Bostr¨ mo 1995). Moreover, decision trees can easily be turned into a rule set by generating one rule for each path from the root to a leaf. However, there are several aspects which make rule learning via the separate-and-conquer strategy attractive:

Decision trees are often quite complex and hard to understand. Quinlan (1993) has noted that even pruned decision trees may be too cumbersome, complex, and inscrutable to provide insight into the domain at hand and has consequently devised procedures for simplifying decision trees into pruned production rule sets (Quinlan 1987a, 1993). Additional evidence for this comes from Rivest (1987) who shows that decision lists (ordered rule sets) with at most k conditions per rule are strictly more expressive than decision trees of depth k. A similar result has been proven in (Bostr¨ m 1995).

---- terms ---

hill-climbing - In computer science, hill climbing is a mathematical optimization technique which belongs to the family of local search. It is an iterative algorithm that starts with an arbitrary solution to a problem, then attempts to find a better solution by incrementally changing a single element of the solution. If the change produces a

better solution, an incremental change is made to the new solution, repeating until no further improvements can be found.

beam-search - In computer science, beam search is a heuristic search algorithm that explores a graph by expanding the most promising node in a limited set. Beam search is an optimization of best-first search that reduces its memory requirements. Best-first search is a graph search which orders all partial solutions (states) according to some heuristic which attempts to predict how close a partial solution is to a complete solution (goal state). But in beam search, only a predetermined number of best partial solutions are kept as candidates.[1]

Beam search uses breadth-first search to build its search tree. At each level of the tree, it generates all successors of the states at the current level, sorting them in increasing order of heuristic cost.[2] However, it only stores a predetermined number of best states at each level (called the beam width). Only those states are expanded next. The greater the beam width, the fewer states are pruned. With an infinite beam width, no states are pruned and beam search is identical to breadth-first search. The beam width bounds the memory required to perform the search. Since a goal state could potentially be pruned, beam search sacrifices completeness (the guarantee that an algorithm will terminate with a solution, if one exists) and optimality (the guarantee that it will find the best solution). The beam width can either be fixed or variable. One approach that uses a variable beam width starts with the width at a minimum. If no solution is found, the beam is widened and the procedure is repeated.

best-first search - a search algorithm which explores a graph by expanding the most promising node chosen according to a specified rule. Judea Pearl described best-first search as estimating the promise of node n by a "heuristic evaluation function f(n) which, in general, may depend on the description of n, the description of the goal, the information gathered by the search up to that point, and most important, on any extra knowledge about the problem domain."[1][2]

breadth-first search - In graph theory, breadth-first search (BFS) is a strategy for searching in a graph when search is limited to essentially two operations: (a) visit and inspect a node of a graph; (b) gain access to visit the nodes that neighbor the currently visited node. The BFS begins at a root node and inspects all the neighboring

nodes. Then for each of those neighbor nodes in turn, it inspects their neighbor nodes which were unvisited, and so on.

depth-first search - Depth-first search (DFS) is an algorithm for traversing or searching a tree, tree structure, or graph. One starts at the root (selecting some node as the root in the graph case) and explores as far as possible along each branch before backtracking.

1999-Logical Structure Analysis of Document Images Based on Emergent Computation

^ this is a really good one... this even has formula detection with evaluation!!!!!!!!!! uses emergent computation, a concept wherein several different modules cooperate in order to reach some sort of optimal state (ie to segment and categorize the logical regions of a document image correctly)SEEE “1999-Logical Structure Analysis of Document Images [80]

1999 – Wisdom++ /2000..... <<<< these aren't cited in any of the big surveys I've found... however... the 1990 article is

wisdom++, intelligent document processing system

http://www.di.uniba.it/~malerba/wisdom++/... wisdom++!!!!!!!!!

WISDOM++ (www.di.uniba.it/~malerba/wisdom++/)

(Manages printed documents such as letters and journals)

Knowledge is represented by means of decision trees and firstorder rules automatically generated from a set of training documents. In particular, an incremental decision tree learning system is applied for the acquisition of decision trees used for the classification of segmented blocks, while a first-order learning system is applied for the induction of rules used for the layout-based classification and understanding of documents. Issues concerning the incremental induction of decision trees and the handling of both numeric and symbolic data in first-order rule learning

are discussed, and the validity of the proposed solutions is empirically evaluated by processing a set of real printed documents.

In this article **(the 2000 article)** we advocate an extensive application of machine learning techniques and tools in order to solve the knowledge acquisition (logical layout analysis) bottleneck problem. This approach has been pursued in the design and development of an intelligent document processing system, named WISDOM++, which is a newer object-oriented version of the system WISDOM (Windows Interface System for DOcument Management) (Malerba et al., 1997b), originally written in C and used to feed a digital library (Esposito et al., 1998). The two main requirements considered in the design of WISDOM++ are real-time user interaction and adaptivity. The former involves choosing fast algorithms for document image analysis, while the latter requires the application of machine learning techniques.

In our work, three problems have been identified in the whole document process:

1. Classification of blocks defined by the segmentation algorithm, in order to separate text from non-text areas in the document image.

2. Assignment of documents to one of a pre-defined set of classes (document classification).

3. Association of semantic (or logic) labels to some layout components (document understanding).

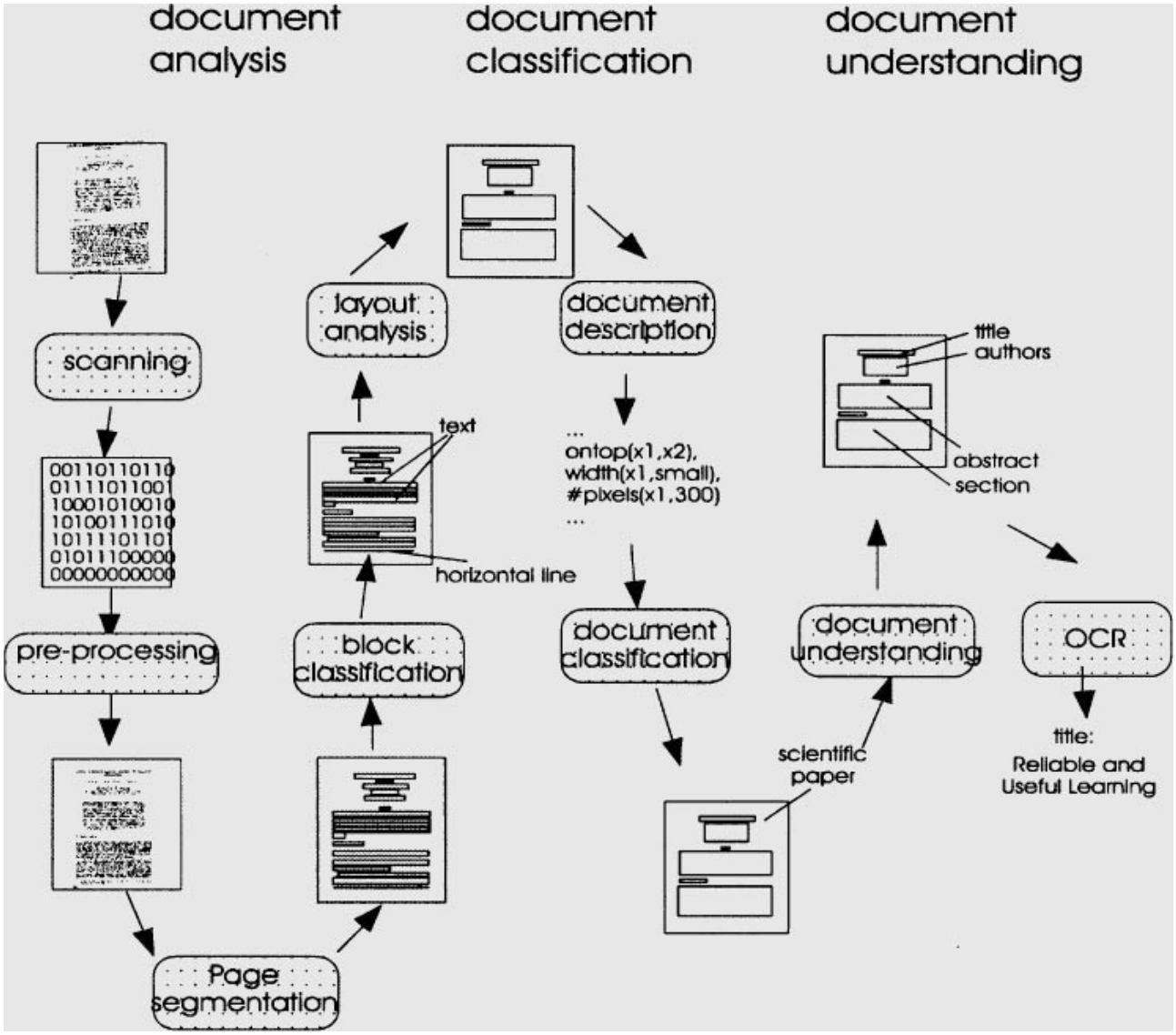
Our approach is based on the idea that humans are generally able to classify documents (invoices, letters, order forms, papers, indexes, etc.) from a perceptive point of view, by recognizing the layout structure of a form. This means that documents belonging to the same class have a set of relevant and invariant layout characteristics, called page layout signature, which can be used for classification. However, the representation of such a page layout signature requires first-order logic

formalisms or equivalents (e.g., attributed graphs), due to the presence of geometric relations among layout components. The same applies to the document understanding problem, since we assume that the identification of layout components

with a logical meaning can be based on their geometric properties and relations. Other representation issues regarding the document processing domain are the accurate preprocessing of document images in order to reduce the effect of noise and the consideration of both numeric and symbolic descriptions, since the former increase the sensitivity while the latter increase the stability of the internal representation of images (Connell and Brady, 1987).

The fourth aspect of the application is the choice of appropriate machine learning tools and techniques. In previous works on block classification, linear discriminant analysis techniques were used (Wang and Srihari, 1989; Wong et al., 1982). It is well-known that these parametric statistical techniques cannot handle complicated non-linear interactions among features. On the contrary, the top-down induction of decision trees does handle such interactions and produces results that are simple to interpret. In fact, this is the approach followed in our work (see Section 3). To solve document classification and understanding problems, it is necessary to resort to first-order learning systems, suitably extended in order to handle both numeric and symbolic attributes and relations (see Section 4).

!!!!!!!!!!!!!111has a really good image ill steal and reference!!!!!!!!!!!!!!!!!!!!!!!!1


document analysis
document classification
document understanding
scanning
pre-processing
Page segmentation
block classification
layout analysis
text
horizontal line
document description
...
ontop(x1,x2),
width(x1,small),
#pixels(x1,300)
...
document classification
scientific paper
document understanding
title
authors
abstract
section
OCR
title:
Reliable and Useful Learning

(2000 Malbera) <- this one uses decisions trees for the block classification and also uses rule-based learning for the page segmentation (physical analysis) <<<<<<<<<<<<<<<<==== block classification uses decision tree methods, organizing the blocks after classification involves rule-based (figuring out the logical meaning of a block, ie title, caption, etc... this is based on geometric relations between blocks and content of blocks?)....the main advantage i see from these articles is that they do a good description of decision tree methods.... but this will be more applicable in the theory section anyway.... moving it to there...

this statement here sums up the 2000 Malbera article:::::::::

Experimental results reported above prove that machine learning techniques can be helpful to acquiring specific knowledge for intelligent processing of printed documents. In particular, two decision tree learning systems have been successfully applied to the problem of classifying blocks defined by the segmentation algorithm, while a first-order rule learning system has been effectively used for document classification and understanding.

so they use decision trees to determine whether a block is text or non-text then use rule-based learning to assign logical meaning to the image/text blocks....

The 1997 one implemnted the rulebased method:::

Preliminary results obtained with the integrated approach were encouraging, but not totally satisfying. This prompted the investigation of an extension of the first-order learning method implemented in INDUBI/CSL (Malerba et al., 1997c), a general-purpose learning system used in this application.

At the high level INDUBI/CSL implements a separate-and-conquer (or sequential covering (Mitchell, 1997)) search strategy to generate a rule. With reference to figure 5, the separate stage corresponds to the external loop that checks for the

completeness of the current rule.2 If this check fails, the search for a new consistent clause is begun.3 The search space for the separate stage is the set of rules, while the search space of the conquer stage is the set of clauses. The conquer stage performs a general-to-specific beam-search to construct a new consistent, linked and range-restricted clause. The separate-and-conquer search strategy is adopted in other well-known learning systems, such as FOIL (Quinlan and Cameron-Jones, 1993). On the other hand, INDUBI/CSL bases the conquer phase on the concept of seed example, whose aim is that of guiding the generalization process.

Preliminary results obtained with the integrated approach were encouraging, but not totally satisfying. This prompted the investigation of an extension of the first-order learning method implemented in INDUBI/CSL (Malerba et al., 1997c), a general-purpose learning system used in this application.

seperate and conquer search strategy implemented .

1999 Wisdom++ Article

WISDOM++ (Windows Interface System for DOcument Management) is a document analysis system that operates in four steps: document analysis, documentclassification, document understanding, and text recognition with an OCR

deskews the document, then uses a top-down segmentation technique.

4 users are allowed to train the system online iti 2.0 is the decision tree used by the system

basically the "head" of a clause would be the name of the production rule in a grammar.

need to talk some about ilp in lit review. this is very relavent

http://archive.ics.uci.edu/ml/datasets/Document+Understanding

2010 - Improved CHAID Algorithm for Document Structure Modelling

^ ..... CHi-squared Automatic Interaction Detection .... this falls under the category of decision tree based approaches.. .so goes alongside malerba stuff....

2008-Incremental Machine Learning Techniques for Document Layout Understanding

^ (me of several weeks ago->) I don't like this one very much in that it involves trying to classify the entire document rather than its individual components. Could mention it in brief as one the ones falling under that category.... nagy references it... so maybe its worth something.... just doesn't seem that relavent to what i'm doing... neeed to take another look.... i recall having mixed feelings about this one..... **did I lchange this part at home?!! No... Maybe this relates to Wisdom++?? Well it's a “knowledge based approach” so I would say so....**

2003 “document analysis structure algorithms” survey..

[**S. Tsujimoto and H. Asada, “Understanding multi-articled documents,” in Proceedings of International Conference on Pattern Recognition, pp. 551–556, (Atlantic City, NJ), June 1990.**] tree representation of document layout structure. document understanding posed as the transformation of a physical tree into

a logical one using a set of generic transformation rules and a virtual field separator technique.

[**A. Yamashita, T. Amano, I. Takahashi, and K. Toyokawa, “A model based layout understanding method for the document recognition system,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 130–138, (Saint-Malo, France), September 1991.**] model-based method for logical structure analysis. describes each document object's logical label, tree level, separator location, minimum and maximum numbers of constituent character strings, as well as its successor's orientation.

[**J. Kreich, A. Luhn, and G. Maderlechner, “An experimental environment for model based document analysis,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 50–58, (Saint-Malo, France), September 1991.**] experimental environmnet called SODA (system for office document analysis) for model-based document analysis. first use a bottom-up approach to group connected components into text blocks, then found lines within each text block and words within each line. no quantitative performance data reported

[**J. L. Fisher, “Logical structure descriptions of segmented document images,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 302–310, (Saint-Malo, France), September 1991.**] Fisher presented a rule-based system for recognizing the physical layout and logical structure of a document image without prior nformation about the document's format or content. automatically extracts the general physical layout of the document and transforms it into a logical strucutre. three rules used: location cues, format cues, and textual cues. reconstructs paragraphs broken during formatting, determine the read order of text blocks, and express its results in a document markup language. text and nontext assumed to be already identified. no experiemntal results given

D. Derrien-Peden, “Frame-based system for macro-typographical structure analysis in scientific papers,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 311–319, (Saint-Malo, France),

September 1991.: frame-based system for analyzing document physical layout and logical structure. document layout structure obtained in three steps. recursive x-y cut. lines extracted using special rules. physical zones obtained from topographical features. reading order obtained from depth-first search of the layout structure. logical recognition done in 2 steps: 1. paragraphs with the same features are grouped into classes, 2. logical labels are assigned to each class using a set of general alyout rules. no experimental results were reported

R. Ingold and D. Armangil, "A top-down document analysis method for logical structure recognition," in Proceedings of International Conference on Document Analysis and Recognition, pp. 41–49, (Saint-Malo, France), September 1991.: document logical structure recognition method using a formal description of each document class that includes composition rules and presentation rules. composition rules define the generic logical structure. presentation rules define the pysical characteristics of the logical entities. composition rules represented by extended backus-naur form grammars. no experimental results reported

R. Brugger, A. Zramdini, and R. Ingold, "Modeling documents for structure recognition using generalized n-gram," in Proceedings of International Conference on Document Analysis and Recognition, pp. 56–60, (Ulm, Germany), August 1997.: document logical structure model based on a statistical representation of patterns in a document class, ie on generalized n-grams. the tree structure of document logical components is represented by the probabilities of local tree node patterns similar to n-grams. logical tree is constructured from the physical entities in conformity with the given model. tree with optimal conformity selected. 5 memo pages used in the experiments, one for training the model and the remaing four for testingl the model.

A. Conway, "Page grammars and page parsing: A syntatic approach to document layout recognition," in Proceedings of International Conference on Document Analysis and Recognition, pp. 761–764, (Tsukuba Science City, Japan), October 1993.: page grammars and page parsing techniques used to recognize document logical structure from phsycial layout. physical layout described by set of grammar rules, each of which is a string of components specificed by a

neighbor relationship. possible neighbor relationships include above, left-of, over, left-side, and close-to, so that the layout is two dimensional. context-free string grammars are used to describe logical structure. both grammars are deterministic. the physical layout grammar has attached constraints to incorporate information such as font size, style, alignment and indentation. no quantitative experimental results reported.

M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan, “Syntactic segmentation and labeling of digitized pages from technical journals,” IEEE Transactions on Pattern Analysis and Machine Intelligence 15, pp. 737–747, 1993.: document logical structure recognition method which recursively applies grammars to horizontal and vertical projecction profiles of the page. parsing process is divided into four stages. 1. lengths of runs of zeros or ones in the thresholded projection profiles are thresholded into atoms. 2. atoms are grouped into molecules. 3. logical labels are assigned to the molecules. 4. contiguous entities of the same type are merged. results saved in a labeled x-y tree. transforms a two dimensioal segmentation and labeling problem into a one-dimensional segmentation and labeling problem in an x-y tree. trained on 21 ibm journal pages and tested on twelve ibm/pami pages. the algorithm performance was reported in terms of percentage of labeled area and missed labels.

T. Saitoh, M. Tachikawa, and T. Yamaai, “Document image segmentation and text area ordering,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 323–329, (Tsukuba Science City, Japan), October 1993.: system for document segmentation, text area classification and ordering. independent of the shapes of the physical blocks and robust to document skew. connected components extracted and classified, then merged into lines which are merged into zones. the extracted zones are classified into body, caption, header, and footer. a tree structure is generated from the classified zones using text area inflence ranges. order of text obtained by pre-order traversal of the tree. the experimental dataset included 131 japanese and english documents which were scanned with skew. size of the final dataset was 393 images. authors used three criteria to evaluate their segmentation and classification results, and three other criteria to evaluate their text ordering results.

Y. Tateisi and N. Itoh, “Using stochastic syntactic analysis for extracting a logical structure from a document image,” in Proceedings of International Conference on Pattern Recognition, pp. 391–394, (Jerusalem, Israel), October 1994.: posed document logical structure analysis as a stochastic syntactic analysis problem. document modeled as a string of text lines and graphic objects. text lines and graphic objects segmented and classified using preprocessing step, and the string is parsed using a stochastic regular grammar with attributes. characters recognized and font lines determined. parser retains possible parsing results in order of their total cost. algorithm tested on sevventy pages of japanese text taken from books and magazines. 86% average markup acuracy on manuals and 82% average markup accuracy on technical papers foor the parsing result with the least cost.

D. Niyogi and S. N. Srihari, “Knowledge-based derivation of document logical structure,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 472–475, (Montreal, Canada), August 1995.: system called DeLoS for document logical structure derivation. In this system, a computational model is developed based on a rule-based control structure as well as a hierarchical multi-level knowledge representation scheme. knowledge about the physical layouts and logical strucctures of various types of documents are encoded into a knowledge base. three levels of rules: knowledge rules, control rules, and strategy rules. doc image first segmented using bottom-up algorithm. segmented blocks are then classified. logical tree structure derived by classified blocks after they are input in to the system. was tested on 44 newspaper pages.

K. Summers, “Near-wordless document structure classification,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 462–465, (Montreal, Canada), August 1995.: algorithm for automatic derivation of logical document struccture from genericc physical layout. algorithm divided into segmentation of text into zones and classification of these zones into logical components. the document logical structure is obtained by computing a distance measure between a physical segment and predefined prototypes. for each logical clabel, a set of prototypes is specified. these include contours, context, successor, height, symbols, and children. algorithm tested on 196 pages of computer science technical reports. accuracies about 85% were reported.

A. Dengel and F. Dubiel, “Computer understanding of document structure,” International Journal of Imaging Systems and Technology 7, pp. 271–278, 1996.: DAVOS system capable of learning and extracting document logical structure. learns document structure concepts by detecting distinct attribute values in document objects. structural concepts are represented by relation patterns defined by a cut-and-label language. a gtree (geometric tree) is used to represent the conept language. unsupervised decision tree based learning techniques are used to build the gtree. bottom-up and top-down approaches are compared. trained on 40 letters and then tested on a different 40 letters.

C. C. Lin, Y. Niwa, and S. Narita, “Logical structure analysis of book document images using contents information,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 1048–1054, (Ulm, Germany), August 1997.: method of analyzing logical structure of books pages using contents page information. the contents page of a book contains a concise and accurate logical structure description of the whole book. text lines are first extracted from the contents page, and ocr is then performed for each text line. the structures of the page number, head, foot, headline, chart and main text of the text page are analyzed and matched with information obtained from the contents page. the algorithm was tested on 235 pages. the exxperimental results were reported in terms of two labeling errors and the logical labeling identification rate.

Y. Ishitani, “Logical structure analysis of document images based on emergent computation,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 189–192, (Bangalore, India), September 1999.: document logical structure analysis system based on emergent computation. includes five interacting modules: typography analysis, object recognition, object segmentation, object grouping, and object modification. doc image is first segmented into text lines, which are then classified into different types using special rules. the classified text lines are then grouped and classified into logical components using heuristic rules. the system was tested on 150 documents taken from various sources. the author reported 96.3% average rate of correct logical object extraction.

J. Kim, D. X. Le, and G. R. Thoma, "Automated labeling in document images," in Proceedings of SPIE Conference on Document Recognition and Retrieval VIII, pp. 111–122, (San Jose, CA), January 2001.: rule-based automated labeling module for medical article records, extracting bibliographic records from medline database. 96% accuracy reported.

(1994 - Document Image Understanding: Geometric and Logical Layout)

Logical Works:::::::::::;;;

Logical Page Structure involves determining the type of page (page classification) assigning functional labels to each block of the page, and ordering the text blocks according to their read order.

Work 1::

Esposito et. al. (1990) develop a syriibolic learning approach and compare it with a statistical approach to accomplish page classification. They used a set of 231 pages and a seven page t es and obtained 100% accuracy for 5 classes and 95% a n d 96% accuracy for the other 2 classes by a combined statistical and symbolic classification method.

Work 2::

Fujisawa and Nakano (1990) classified 81 Japanese Patent Disclosure Bulletin 200dpi document image pages into one of the three classes front page, text page wiihout figures, or text page with figures with 100% accuracy. In a second test utilizing 106 Japanese Patent Application 200dpi document image pages, they had six classes: separator sheet, front page type-1, front page type-2, claim page, text page, or figure page. The recognition rate was 98%.

Work 3::

Tsujimoto and Asada (1990) assume that each block of the geometric page layout contains exactly one logical class. They organize the geometric page layout as a tree. Each new article in a document such as a newspaper begins with a headline which is in the head block. They find the paragraphs which belong to the head block by rules relating to the order of the geometric page layout tree and are able to assign logical structure labels of title, abstract, sub-title, paragraph, header, footer, page number,

and caption. They worked on 106 document images and correctly determined the logical structure for 94 document images.

Work 4::

Visvanathan (1990) employs an X - Y tree t o represent the geometric layout and then employed a regular grammar scheme t o label the document image blocks. Block labels included: title, author, abstract, section titles, paragraphs, figure, table, footnote, footers and page numbers. No performance results were given.

Work 5::::::::::::::::::::::::::::

Fisher (1991) is an extension of Fisher (1990) and describes a rule based system to identify the geometrical and logical structure of document images. No performance results are given.

Work 6::::::::::::::::::::::::::::::;;

Ingold and Armangil (1991) describe a formal top-down method for determining the logical structure. Each document class has a formal description that includes composition rules and presentation rules. They have utilized the technique on legal documents. No performance results are given.

Work 7::::::::::::::::::::::::::::

Chenevoy and Belaid (1991 use a blackboard system for a topdown method o logical structure analysis of a document image. The system is defined in a Lisp formalism and has a hypothesis management component using probabilities. No performance results are given.

Work 8::::::::::::::::::::::::::::

Kreich et. al. (1991) describe a knowledge-based method for determining the logical structure of a document image. To obtain the blocks they search for the largest text blocks because these are the most characteristic elements in the document layout. The search consists of grouping together the connected components which are close enoough t o each other. Once text blocks are determined, lines are found within each of the text blocks and words within the lines. The determination of document layout structure is based on interpreting documents and their parts as instance of hierarchically organized classes. They have defined over 300 classes for a document image and its parts. No performance results are given.

Work 9::::::::::::::::::::::::;

Derrien-Peden (1991) describes a frame-based system fo the determination of structure in a scientific and technical document image. The basis of this system is a macro-typographical analysis. The idea is that in a scientific and technical documents, changes of character size or thickness of type, white separating spaces, indentation etc. are used to make visual searching for information easier. So the technique searches for such typographical indications in the document and recovers its logical organization without any interpretation of its semantic content. The first step is the determination of the geometric page layout keeping a part ofrelationship between blocks. The logical structure determination removes running headers and footnotes and searches for the text reading order. Text blocks are then compared t o logical models of classes and each text block is then assigned a class. No performance results are given.

Work 10::::::::::::::::::::::::

Yamashita et. al. (1991) use a model-based method. Character strings, lines, and half-tone images are extracted from the document image. Vertical and horizontal field separators (long white areas or black lines) are detected based on the extracted elements, then appropriate labels are assigned t o character strings by a relaxation method. Label classes included: header, title, author, a m i a t i o n , abstract, body, page number, column, footnote, block and figure. The technique was applied to 77 front pages of Japanese patent applications. They reported that the logical structure for 59 were determined perfectly.

Work 11::::::::::::::::;

Dengel (1993) discusses a technique for automatically determining the logical structure of business letters. He reports that on a test set of 100 letters, the recipient and the letter body could be correctly determined.

Work 12::::::::::::::::::::::::::;

Saitoh et. al. (1993) determine logical layout with text block labels of body, header, footer, and caption. They tested the technique on 393 document images of mainly Japanese and some English documents. To characterize performance they measured the average number of times per document image an operator has to correct the results of the automatically produced layout. They report that on the average 2.17 times per image areas not suitable for output have t o be discarded, .01 times per image mis-classified areas have to be correctly labeled, and 1.09 times per image does a text area have to be reset. With respect to text ordering they report that it

required moving connections .47 times per image, on the average, making new connections .11 times per image, and re-assigning type of test .36 times per image.

2.3.3 Representing Document Structure and Layout

Sub-sec 3:: Representing Document Structure and Layout <--- 2007 article talks about formal grammars, hocr (derived from html)....etc. Etc.

This'll be a pretty brief section. Representing the recognized text in a manner that is spatially accurate to the orignal document. This is where the document understanding and page layout analysis done in earlier stages is put to use. The information about the document, also often considered as metadata, is used to dictate the output format, in a similar fashion to html or css style sheets. HOCR (PAMI article should be referenced some too, does a good job of describing this)

tree structured

formal grammar, context free grammer, etc etc

2007 - Document Structure and Layout Analysis --- this has a very nice high level description of algorithms.....discusses how context free grammars can be useful for describing document logical structure (also stochastic grammar where there is a probability associated with each

production rule..and attributed grammars, basing prodcution rules on attributes...)...,,,

1995 represenation of doc structure... lots of good stuff here

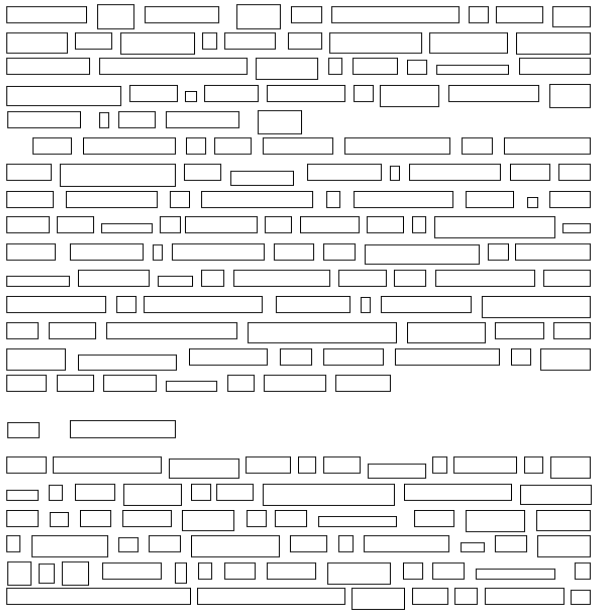
surfaces (see Section 2.3), the combination of local planarity and rigidity is used. For arbitrary motion, rigidity between environmental points is used to recover motion parameters from a small number of image locations (See Section 2 and Section 3.1).

The reminder of this section introduces the notation used throughout this paper. Section 2 describes how the local direction of translation is estimated from a flow field and cases of motion for which this is particularly robust. Section 3 describes how the parameters of relative sensor motion can be recovered from the estimated local directions of translation. Section 4 discusses computing the local translational decomposition directly from real image sequences without the initial extraction of optic flow and other areas for future work.

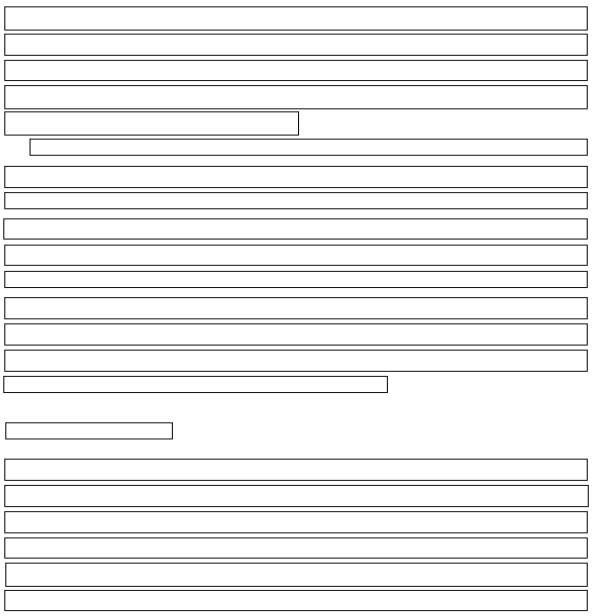
1.1 Notation

The coordinate system used in this paper is shown in Figure 1. The origin of this right-handed coordinate system lies at the focal point of the camera. The image plane is parallel to the xy-plane and is centered on the point $(0, 0, f)$, where f is the focal length of the camera. A three-dimensional environmental point will be referred to

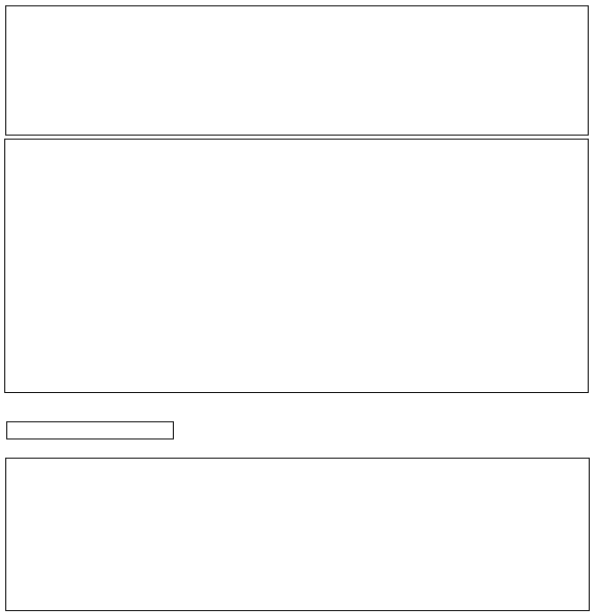
(a) original image



(c) word bounding boxes



(b) text line bounding boxes



(d) text block bounding boxes

Standards for representing the logical structure of documents:::: SGML is an example. Need to enumerate the others..

Logical structure reflects semantics which are part of the author's intention. While there may exist multiple physical representations of a document, there should be only one logiccal representation.

Document creation vs Document "reverse encoding". During reverse encoding process there may be varying levels of uncertainty in the interpretation of aspects within the document. In the document creation process this ambiguity is not present, since the document is usually encoded by the same person who created the document represenation.

In document image reverse encoding, it is important to move back and forth between the interpretation of the physical and semantic structure of the document. Although they rely heavily on one another and may share common aspects, they are still two different ways of perceiving the structure of a document. The fact that typiucally there is not a one-to-one mapping between the physcial and logical structure complicates this requirement.

Throughout the process of document analysis (aka document image reverse encoding, document image decoding.. (need to enumerate the list of terms used to describe this same concept))... the ultimate goal is to use physical attributes to obtain a consistent and valid interpretation of the semantic attributes. For instance, the text-blocks should be semantically ordered by the "reading order" of the text units. Each text unit should be assigned a semantic label (ie for a business letter there might be a sender and receiver address, date, opening salutation, body, closing, and signature... Likewise for a technical article, the title, author(s), abstract, keywords, sections, displayed equations, tables, graphs, illustrations, footnotes, page numbers, reference list, and other logical components can be deduced by their locations and/or sequencing as well as the fonts, styles and sizes of the characters that make them up. The resulting recognized text strings should be formatted such that their two-dimensional layout, deduced from the 2-D layout analysis is recorded along with the text itself.

Resulting complex data structure should capture the entire semantics of the original document (assuming it was constructed correctly).

Geo Layout section::

Many of the algorithms for determining geometric layout employ the operations of mathematical morphology

The lit review in this section is pulled directly from the '94 heralick article.....

:::::Logical Layout Analysis

Skipping thisl..... in favor of the heralick part..... thats good enough for now...

::::Logical and Physical Document Description

In order to describe a document's physical and logical characteristics consistently, it is advantageous to first distinguish between the document's content and its structure.

:::::Doc Content vs Doc Structure

The content is the information contained in the document, not boudn to any representation format.

The physical structure is how the document's content is laid out on the physical medium. The same content can be orgtanized in a variety of ways and therefore can have many physical layouts which stem from different values of the attributes of the physical components (point size, line spacing, page size, etc).

The logical structure ofa document's contents are how its content is organized, prior to the enforcement of a particular physical structure.

::::Realizing Doc Structures

Determining the type of document is an important step for any document analysis system. For purposes of this thesis, this will be only briefly discussed... In order to properly extract and classify the zones of a document it is often necessary to have a some knowledge on what category the document falls under (ie business letter, technical article, check, form, etc)

:::::Generic (ie could apply to many different types of documents) Physical Structure

Some definitions related to the generic physical structure of a document:

Frame - an area within a page of a document (may consist of a collection of lower level frames and/or blocks)

Root Frame - a frame which encompesses the entire physical document

Page - a frame which occupies a rectangular region of a page, on which the document is physically recorded. This is the basic physical document object (ie the paper it was printed on originally)

Page Set - a frame which consists of multiple pages. An instances of a page set may be a single volume or chapter for example.

Block or Simple Frame - a terminal, lowest level frame, which, at the given level of granularity, need not or cannot be further decomposed. This could be a paragraph for instance

A frame is defined to be a recursive component, as it may consist of a set of one or more frames. A block is the terminal frame, which defines a region on the page and has content. Depending on the desired level of granularity, a block's content may correspond to a column, line, word, or character, for example.

Each type of frame has an associated set of attributes. Information about a frame's location, position on the page, justification, etc. is defined by its attributes. For example, a character is characterized by such attributes as font, boldness, point-size, inclination; a line, by maximal length; etc.

Types of blocks:

- character - a block containing the image of a symbol in the document's language

- word - a frame containg a group of one or more aligned characters, separated by white space. the specializations of a word are as follows:

  - subword - a frame containing a group of one or more aligned characters, which make up the first part or the last part of a word

  - preword - a subword containing the first part of a word. a preword is located at the end of a line and ends with a hyphen

  - postword- a subword containing the last part of a word. A postword is located at the beginning of a line and completes its preceeding preword to a whole word.

- line - a frame containing 1) a collection of one or more aligned words and 2) at most one preword and at most one postword

- stack - a frame containing a collection of one or more lines stacked on top of each other and possibly separated by a non-empty white space, such that its logical content is one paragraph at most

- column - a frame containing a collection of one or more stacks on top of each other. A page may contain one or more columns. In structured documents this number is generally fixed throughout the document.

Generic logical structure::::::::::

texton is definied to be the logical analog of "frame". could include chapter, section, paragraph, etc....

texton - a logical component of a text-intensive document, which consists of one or more (lower level) textons or simple texton.

root texton - a textn which is the entire document (ie book, dictionary, journal, magazine, etc.)

simple texton - (usually a character is the base logical unit....)a logical component of a text-intensive document, which is not further divided. Instances of a simple texton are paragraph, sentence, phrase, word, and character. If, for example, word is the simple texton in a particular document, then any subcomponent such as a character is a primitive texton document.

compount texton - a texton consisting of a distinct header, body, and optional trailer. One example is a section of a document, which has a header-the section head, a body - the set of one or more paragraphs, and no trailer. Another, less obvious example of a compound texton may be a signature block in a letter, which contains a header-the closing, a body- the signature and a trailer- the printed name

words and characters - the logical meaning of these is different from their physical in that the physical is the image represenation and logical is the linguistic meaning

phrase - a texton which is a meaningful collection of one or more words that do not necessarily form a complete grammatical sentence (ie a title, name of person, address, or a list)

sentence - a meaningful collection of one or more phrases which correspond to a valid grammatical sentence, complete with punctuation.

paragraph - a texton whihc is a generalization of a paragraph. It consists of a group of one or more sentences and/or phrases.

referenced texton - a graphic or textual texton which is referenced from the document. examples of referenced components include figures, appendices,

footnotes, citations, continuation text body (eg in newspapers) and even complete documents. The header of a texton is a label or identifier which is "referenced" by a pointer.

pointer - a referencing texton pointing from the main text of the document to the referenced texton.

graphon - a referenced texton in a document whose nature is mainly non-textual and whose funtion is to illustrate, explain or demonstrate text. examples include line drawings, half-tones, photographic images, maps, diagrams, charts, tables, etc.may contain a caption, with caption header and optional body.

reading order - the order in which the characters or symbols in a text-intensive document must be traversed for the document to be correctly understood. Logically, anything referenced (ie an image, equation, etc) will be visited in the appropriate reading order based upon where the corresponding pointers appear in the text.

Document Complexity::::::::::::;

Logical complexity is the number of levels for the document's tree representation (ie root->section->subsection->paragraph, sentence, phrase, word, character

Relating physical and logical structure::::::::::

Example: a a multiple page chapter may be considered a compound texton, with a header (title) and two body components (one abstract and one section). The abstract is a simple texton and teh section is a compount texton consisting of two simple textons (paragraphons). The physicial structure subdivides the document into rectangular blocks. The content is shared in both structures. .....

Document Attribute Format Specification (DAFS)::::::::

This describes the physical layout of the document

It was developed by the Document Image Understanding (DIMUND) project funded by ARPA and is meant to be the file format for all documents used whose content has been examined either manually or automatically and which form parts of DIMUND databases. In addition, DAFS-formatted documents are used in the Illuminator proejct, where they are employed for the training and testing document image understanding tools.

DAFS Design::::::::::::;;

Object Oriented Design - under DAFS an object is called an entity and is essentially one or more rectangular pieces of image from the document. The DAFS entity is analogous to the SGML element and has nothing to do with the SGML concept of entity. Examples are paragraph, character, and document.

Hi////.....

DAFS is heavily influenced by SGML. for more detailed description of DAFS see.....

2.3.4 Training Sets

Sub-sec 4:: Training sets./ Evaluation...

Training:: (((this is from the 2008 survey!!!!!!!!!)

UW databases (http://www.science.uva.nl/research/dlia/datasets/uwash3.html)

The UW databases are three document collections that have been gathered and manually annotated by the Intelligent Systems Laboratory, at the University of Washington (WA) in the late 1990's under the supervision of Prof. Robert Haralick. The databases were distributed as CD-ROMs containing document images and software for research and development.

The UW-III is the third in the series (it was published in 1996) and contains pages of chemical and mathematical equations, pages of line drawings and engineering drawings. There are also 33 pages containing English text with the corresponding character level groundtruth, 979 pages from UW-I, and 623 pages from UW-II corrected for skew, and the word bounding boxes for each word on these pages. These CD-ROMs, distributed for research purposes under the payment of a small fee, have been a reference for many years for the research on printed text.

NIST databases (not free!!!!)

In the 1990’s the National Institute of Standards and Technologies (NIST) produced several CD-ROMs aimed at supporting the research on OCR software and information retrieval systems. A fully-automated process developed at NIST was used to derive the groundtruth information for the document images. The method involves matching the OCR results from a page with typesetting files for an entire book. The databases included scanned images, SGML-tagged groundtruth text, commercial OCR results, and image quality assessment results.

NIST’s SD-3 (Special Database 3) and SD-1 contained binary images of handwritten digits. NIST originally designated SD-3 as training set and SD-1 as test set. However, SD-3 is much cleaner and easier to recognize than SD-1 and this fact is a limit for comparing different algorithms.

MediaTeam database (oulu) (http://www.mediateam.oulu.fi/downloads/MTDB/)

The MediaTeam Oulu Document Database4 is a collection of 500 scanned document images with related groundtruth information about the physical and logical structure of the documents. The images cover a broad range of document types including journal papers, maps, newsletters, form, music, dictionaries and can be used for comparing various tasks in DAR.

Infty project

The InftyProject is a voluntary R&D organization consisting of researchers from different universities and research institutes in Japan with the shared objective of investigating and developing new systems to process scientific information by computer. Starting from 2005 three datasets have been dis-

tributed5 . InftyCDB-1 [32] consists of mathematical articles in English containing 688,580 objects (characters and mathematical symbols) from 476 pages. The image of each object is recorded together with appropriate ground-truth information. InftyCDB-2 has the same structure of InftyCDB-1 and contains some documents in German and French, as well as many in English. InftyCDB-3 is a database of single alphanumeric characters and mathematical symbols. Unlike InftyCDB-1 and InftyCDB-2, word and mathematical expression structure is not included. The images are of individual characters only for a total of 259,389 symbols.

MARG database <------ very good look into more (http://marg.nlm.nih.gov/index2.asp) MARG7 is a freely-available repository of document page images and their associated groundtruth information on the textual and layout content. The pages are representative of articles drawn from the corpus of important biomedical journals. The database is suitable for the development of algorithms to locate and extract text from the bibliographic fields typical of articles within such journals. These fields include the article title, author names, institutional affiliations, abstracts and possibly others. Only the first page of each article is available, or the second page if the abstract runs over [34].

Generating training sets:::::

2012 - VeriClick, an efficient tool for table format verification

^ this is that Nagy article that references the previous one i had doubts on..... the concept of this is intriguing... to train a classifier to generate groundtruth data....

2.3.5 Performance Evaluation

2008-Geometric Layout Analysis of Scanned Documents (p. 67)

2007 - Document Structure and Layout Analysis - also some good tips on evaluation metrics. Also the following image is really good!!!!!!!!!

2003 “document analysis structure algorithms” survey..

rigorous empirical comparison of five document physical layout analysis using the pset software package [**S. Mao and T. Kanungo, “Software architecture of PSET: A page segmentation evaluation toolkit,” International Journal on Document Analysis and Recognition 4, pp. 205–217, 2002.**][**S. Mao and T. Kanungo, “Empirical performance evaluation methodology and its application to page segmentation algorithms,” IEEE Transactions on Pattern Analysis and Machine Intelligence 23, pp. 242–256, 2001.**]

Liang et al [**J. S. Liang, I. T. Phillips, and R. M. Haralick, “Performance evaluation of document structure extraction algorithms,” Computer Vision and Image Understanding 84, pp. 144–159, 2001.**] propose a performance metric for evaluating document structure extraction algorithms. They evaluated several document layout analysis algorithms on 1600 images from the uw-III dataset.

ocr zoning evaluation method [**J. Kanai, S. V. Rice, T. A. Nartker, and G. Nagy, “Automated evaluation of OCR zoning,” IEEE Transactions on Pattern Analysis and Machine Intelligence 17, pp. 86–90, 1995.**]

yearly conference for ocr evaluation [**S. V. Rice, F. R. Jenkins, and T. A. Nartker, “The fifth annual test of OCR accuracy,” Tech. Rep. TR-96-01, University of Nevada, Las Vegas, NV, 1996.**]

yanikoglu and vincent [**B. A. Yanikoglu and L. Vincent, “Pink pather: A complete environment for ground-truthing and benchmarking document**

page segmentation," Pattern Recognition 31, pp. 1191–204, 1998.] describe an environment (called pink panther) for ground-truthing and benchmarking document page segmentation. they use a bitmap-level region-based metric.

2003 "document analysis structure algorithms" survey..

Algorithm Performance Evaluation:

Important evaluation aspects: performance metric, experimental dataset, groundtruth specification, performance results, error analysis and comparative evaluation.

See [**J. Hu, R. Kashi, D. P. Lopresti, and G. T. Wilfong, "Evaluating the performance of table processing algorithms," International Journal on Document Analysis and Recognition 4, pp. 140–153, 2001.**] [**M. Hurst, "Layout and language: An efficient algorithm for detecting text blocks based on spatial and linguistic evidence," in Proceedings of SPIE Conference on Document Recognition, pp. 56–67, (San Jose, CA), January 2001.**] for document structure analysis of a particular type including performance evaluation...

a meaningful and computable metric is necessary for quantitatively evaluating the performance of any algorithm. it's a funcction of the dataset, groundtruth and algorithm parameters. a performance metric is typically not unique, and researchers can select particular performance metrics and study particular aspects of the evaluated algorithms.

[M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan, "Syntactic segmentation and labeling of digitized pages from technical journals," IEEE Transactions on Pattern Analysis and Machine Intelligence 15, pp. 737–747, 1993.] metric based on the percentage of area labeled and missed labels. [**T. Saitoh,**

M. Tachikawa, and T. Yamaai, “Document image segmentation and text area ordering,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 323–329, (Tsukuba Science City, Japan), October 1993.] used 3 criteria to show the results of their algorithm, based on three proposed ways of using their experminental results. [**D. Niyogi and S. N. Srihari, “Knowledge-based derivation of document logical structure,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 472–475, (Montreal, Canada), August 1995.**] reported results using 3 metrics: block classification, block grouping, and read-order accuracy. [**C. C. Lin, Y. Niwa, and S. Narita, “Logical structure analysis of book document images using contents information,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 1048–1054, (Ulm, Germany), August 1997.]** used two types of labeling errors and an identification rate to report the experimental results of their algorithm. a common aspect of these metrics is their lack of formal definitions; verbal descriptions are usedi nstead.

[**A. Yamashita, T. Amano, I. Takahashi, and K. Toyokawa, “A model based layout understanding method for the document recognition system,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 130–138, (Saint-Malo, France), September 1991.**] cost function based metric for selecting results with the least cost. [**J. Kreich, A. Luhn, and G. Maderlechner, “An experimental environment for model based document analysis,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 50–58, (Saint-Malo, France), September 1991.**] used a generalized hamming metric to compute a confidence measure for matches between a document physical layout and logical structure knowledge base and document object. [**K. Summers, “Near-wordless document structure classification,” in Proceedings of International Conference on Document Analysis and Recognition, pp. 462–465, (Montreal, Canada), August 1995.**] precise and generalized accuracy metrics and reported the performance of algorithm using these metrics. [**A. Dengel and F. Dubiel, “Computer understanding of document structure,” International Journal of Imaging Systems and Technology 7, pp. 271–278, 1996.**] used recall, precision, and F value to evaluate performance. [**Y. Ishitani, “Logical structure analysis of document images based on emergent computation,” in Proceedings of**

International Conference on Document Analysis and Recognition, pp. 189–192, (Bangalore, India), September 1999.] [**S. Tsujimoto and H. Asada, "Understanding multi-articled documents," in Proceedings of International Conference on Pattern Recognition, pp. 551–556, (Atlantic City, NJ), June 1990.**] [**Y. Tateisi and N. Itoh, "Using stochastic syntactic analysis for extracting a logical structure from a document image," in Proceedings of International Conference on Pattern Recognition, pp. 391–394, (Jerusalem, Israel), October 1994.**] experimental results were reported but with no clear definition of performance metrics. [12,34-36,38] no quantitatve experimental results reported. **<---look back at this.....**

evaluation basedo n large-scale experimental datasets is crucial for objectively evaluation the performance of algorithms and assessing the state of the art. in

T. Saitoh, M. Tachikawa, and T. Yamaai, "Document image segmentation and text area ordering," in Proceedings of International Conference on Document Analysis and Recognition, pp. 323–329, (Tsukuba Science City, Japan), October 1993.

C. C. Lin, Y. Niwa, and S. Narita, "Logical structure analysis of book document images using contents information," in Proceedings of International Conference on Document Analysis and Recognition, pp. 1048–1054, (Ulm, Germany), August 1997.

Y. Ishitani, "Logical structure analysis of document images based on emergent computation," in Proceedings of International Conference on Document Analysis and Recognition, pp. 189–192, (Bangalore, India), September 1999.

S. Tsujimoto and H. Asada, "Understanding multi-articled documents," in Proceedings of International Conference on Pattern Recognition, pp. 551–556, (Atlantic City, NJ), June 1990.

K. Summers, "Near-wordless document structure classification," in Proceedings of International Conference on Document Analysis and Recognition, pp. 462–465, (Montreal, Canada), August 1995.

more than 100 document images were used. in [6,11,30,37,39] tens of document images were used. other authors [5,12,38] however tested their algorithms on very

smalldatasets. in [34-36] no dataset was specified. none of the authors clearly specified the groundtruth of the datasets used for testing their algorithms. **<--- come back to this..**

comparative performance evaluations are necessary for comparing the performance of algorithms on some common ground and identifying state-of-the-art techniques. however, for most algorithms, there is a lack of comparative evaluation. [**A. Dengel and F. Dubiel, "Computer understanding of document structure," International Journal of Imaging Systems and Technology 7, pp. 271–278, 1996.**] performed a comparative evaluation of the bottom-up and top-down version of their algorithm through learning and testing procedures.

2.3.6 Conclusion

2003 "document analysis structure algorithms" survey..

---

summary

most the past work has been limited in one or more respects:

1. much of the owrk has not been based on formal models for document pages. here are some advantages of formal models:

(a) one can use a model that has an appropriate level of complexity for a given class of documents

(b) model parameters can be estimated from examples of a given document class

(c) formal models can be used for both analysis and sytnesis of documents. !!!!! model can be used to generate synthetic page image data which can be used in controlled experiments!!!!

2. much of the work on logical structure analysis assumes that physical layout analysis has already been performed

3. most of the work makes use of deterministic models. such models fail in the presence of noise or ambiguity.

4. in some of the work, quantitatvie performance evaluation issues have been neglected.

:::::formal models::::: importance of this is really stressed...

characterizing the complexity of document images in a given dataset, could allow choosing appropriate analysis techniques. if formal models are used, this issue could be addressed

the use of generative document models would enable simulation of document images and performing controlled experiments to evaluate algorithms and study their breakdown points.

A soundly designed experimental methodology should include: a meaningful and computable performance metric, large datasets with well-defined groundtruth, a training procedure and a testing procedure, a thorough error analysis and, finally, comparisons with other state-of-the-art algorithms. very few algorithms have used such complete experimental designs.

1997 geometric range searching and its relatives <--- ... a lot of theory in here. Maybe not so good for lit review.. but this and similar publications will be helpful as a reference while designing and implementing tthe software.....

To ignore::

1. 1999 A Document Classification and Extraction System with Learning Ability << this isn't very good.... i'm just going to ignore it then

2. 2004 Syntactic Modeling and Recognition of Document Images < this is useless... its mainly just for “handwritten forms”.....) so ignore

3. 2008-Improving State-of-the-Art OCR through High-Precision Document-Specific Modeling <-----This one uses tesseract but its completely irrelevant to document analysis. Its more about improving accuracy under extremely high amounts of noise.... **scrapping it.....**

4. 1994 - “Advancements in Computational Geometry for Document Analysis”

has some good references on nearest neighborhood decision rules for ocr. See “Geometric Range Searching and its Relatives....”. “Advancements” article discusses: for layout analysis: textline orientation estimation techniques, textline interference problem? This is where you have lines with different orientations like some horizontal some vertical etc. for ocr: polygonal approximation, the two fundumantal problems in ocr: feature extraction (shape analysis) – medial axis (skeleton) skelotonization/thinning algorithms (pixel based approach, and vectorized approach), proximity graph (a graph whose

vertices are only connected if certain geometric properties are met). **Line sweep technique** for shape recognition. classification - nearest neighbor rule, k-nearest neighbor rules based on proximity graphs?, nearest neighbor search, often using either bucketing techniques or kd-trees (for low dimension classification problems, fewer than 8 dimensions). **Scrapping for now.... talks mainly about OCR.. not layout analysis.....**

2.4 Detection of Equations in Scientific/Mathematical Documents

Garain and related – start with earliest stuff i've found. Also discuss google if possible.

2.5 Recognition of Scientific/Mathematical Documents

Things leading up to and including InftyReader. This section will have a lot of things I looked into as an undergrad. From the outset, indicate that this is a much more mature area of study than equation detection.

future work ideas:: handwriting detection (often there is handwritten notes in these documents which could cause further confusion). A handwriting detection module, may therefore, be of great importance to accuracy for many old books.

Theory section:::

SVM, ID3, C4.5, random forest, multi-class svm

3 Text styles

3.1 Text styles

Heading 4: This is a paragraph heading

Heading 5: This is a sub-paragraph heading

Heading 6: This is a sub-sub-paragraph heading

If you need more headings within the chapters of the document, use Heading 5 or, if necessary, Heading 4 and Heading 5 together.

Paragraph inline headings. An additional possibility is to use paragraph inline headings. Inline headings are emphasized titles at the beginning of a paragraph. It has been tried and tested to provide more clarity and navigability in a lengthy text with these inline headings.

Paragraphs without an inline header are continuations of paragraphs with an inline header. They use a different paragraph style, lacking indentation to support the continuation impression.

`Paragraph verbatim inline heading.` You would use this verbatim inline heading style for source fragments or verbatim quotes of computer-generated text such as filenames etc..

Emphasis and Deemphasis. It is quite usual to have a style for *emphasizing* text. In this template there is also a style for *deemphasizing* text. You might e.g. adopt the custom to deemphasize the words “et al.” when referring to multiple authors: John Curloe *et al.*, for example.

Mathematical formulas. You might use an where applicable or a stand-alone formula with its own numbering. Choose one of the dedicated paragraph styles for either right alignment of formulas:

(1)

or for centered alignment of formulas:

(2)

Source code. To place small chunks of soure code or verbatim quotes of computer-generated text inline into your text, use this `inline style for source and verbatim text`. For whole paragraphs of source or computer-generated text, use the dedicated paragraph style for pre-formatted text:

```
<source code>
<source code>
<source code>
<source code>
<source code>
```

Refer to ►appendix B (p. III) for a description how to place page-long listings of source code into your document together with source highlighting.

Draft mode. There are some text styles for special purposes. For example, while developing a thesis it is handy to mark paragraphs as “in draft quality”. For that purpose, two paragraph styles are provided, one for paragraphs with inline headers and one for those without them:

Draft mode paragraph. This is a paragraph style for draft quality paragraphs with inline heading.

This is a paragraph style for draft quality paragraphs without inline heading.

Todo items. While developing a thesis you will encounter the need to place todo items within your text. They should be marked out to be easily recognized lateron. For small inline todo notes and marks use the inline todo style, for whole paragraphs use the dedicated paragraph style:

This is the paragraph style for todo items.
As you see, consecutive todo item paragraphs are joined together.

1. It is also possible to mark numbered list items as todo items.
2. It is also possible to mark numbered list items as todo items.

■It is also possible to mark bulleted list items as todo items.
■It is also possible to mark bulleted list items as todo items.

1.1 Special text styles for patterns

Summary. Patterns are a special form of verbalizing content in computer science. The rest of this sub-chapter contains some paragraph styles that you can use to format a pattern. The subpart titles are chosen with reference to the PLML pattern format. The pattern title would appear in the chapter or sub-chapter that is reserved to contain the pattern.

<pattern synopsis>

Problem. Problem description.

Context. Context description.

Solution. Solution description.

Inline heading. Another paragraph of the solution description, with inline heading.

And another paragraph of the solution description, without inline heading.

Evidence: Rationale. <rationale description>

Related patterns.

- **Related Pattern.** Related pattern description.
- **Related Pattern.** Related pattern description.

1.1 Lists

Numbered lists. You might use numbered lists together with inline headings:

1. **List item 1 line header.** List item 1 text.
    1. **List item 1.1 line header.** List item 1.1 text.
    2. **List item 1.2 line header.** List item 1.2 text.
2. **List item 2 line header.** List item 2 text.
3. **List item 3 line header.** List item 3 text.

It is however no problem to do without inline headings of course. But remember to right-click the paragraph and choose “restart numbering”.

1. List item 1 text.
2. List item 2 text.
3. List item 3 text.

Bulleted lists. You might use bulleted lists together with inline headings or without them or in mixed form:

- **List item line header.** List item text.
  - List item text.
  - List item text.
  - List item text.
- **List item line header.** List item text.

Note the fully worked out hierarchy of bullets of this bullet list style:

- List item level 1.
  - List item level 2.
    - List item level 3.
      - List item level 4.
        - List item level 5.
          - List item level 6.
            - List item level 7.
              - List item level 8.
                - List item level 9.
                  - List item level 10.

Definition lists. The last list style available is the “definition list”. Something similar ist known from LaTeX and comes in very handy there:

<definition list term 1>

<definition list description 1>

<definition list term 2>

<definition list description 2>

1.1 Linking and referencing

URLs. You might use footnotes to mention URLs directly and not via bibliographic references, e.g. to mention the example.org site[3]. This does not clutter the text with URLs but is better than linking without mentioning the URL, as it preserves full functionality for printed versions.

Footnotes. And you might use footnotes for additional annotations[4] that have no place in the flow of thoughts. As you see, we place a special character in front of every footnote to mark out hyperlinks in PDF versions better.

Internal references. All hyperlinks (including document-internal references) are prepended with a link flag in this template. Link flags help detecting active elements in PDF documents but can become ugly if there are too much. The following basic styles are proven:

- (see ►p. 116)
- see ►object 1 (p. 118)
  see ►chapter Error: Reference source not found (p. Error: Reference source not found)
  see ►appendix B (p. III)
- (see ►object 1, p. 118)
  (see ►chapter Error: Reference source not found, p. Error: Reference source not found)
  (see ►appendix B, p. III)
- do not prepend glossary entries with a link flag

1.1 Citing and bibliography

Citing. Block quotes have their dedicated paragraph style and might span one or multiple paragraphs:

[3] ►http://www.example.org
[4] Such as this annotation.

> This is a paragraph where some other work is cited. Which means that this very text that you read is the tet of the citation, drawn from this very other work. For convenience, an unknown dummy work is cited.
> Another paragraph of the citation is appended using a forced linebreak, not by starting a real new paragraph. ►[p. 1$_{-198}$]

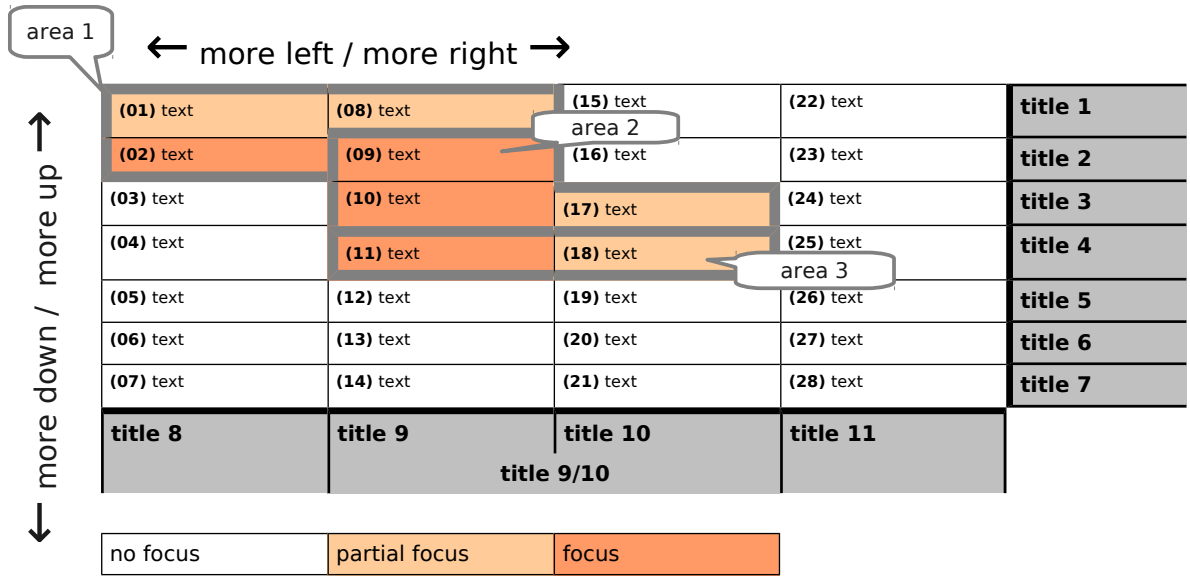
Bibliographic references Here are examples of bibliographic references of every type used in this template. See the bibliography and meta pages on how these bibliographic types differ. Note that these bibliographic references are hyperlinked in the PDF output though this is not natively supported by OpenOffice.org yet. The idea is to mark bibliography items as headings (menu “Extras :: Chapter numbering …”), then insert hyperlinks to headings with the bibliographic reference as link text. Do this just before finishing your document or, even better, implement it in OOo or bibus.

- ARTICLE: ►[, pp. 199-201$_{-198}$]
- BOOK: ►[, pp. 1.3]
- INBOOK: ►[, p. 543$_{+120}$]
- INCOLLECTION: ►[]
- INPROCEEDINGS: ►[]
- MASTERTHESIS: ►[]
- MISC: ►[]
- PHDTHESIS: ►]
- WWW: ►, ► chp. 4.1] (a bibliographic reference including a hyperlinked subpart marker)

1 Object demonstration

Summary. In this template, all framed content is referred to as “objects” regardless of the actual content (images, tables, diagrams etc.). So only one index of objects is necessary, which is far more clear than one index for each type of frames. Note that all frames are anchored to the paragraph whose text starts *below* the frame.

Table object. Here is a demonstration of a table within a frame. Note the additional OOo Draw elements placed over the table and anchored to the frame. For graphical tables such as this better use hard formatting than paragraph styles, to not clutter your style namespace.



Object 1: table with OOo Draw elements

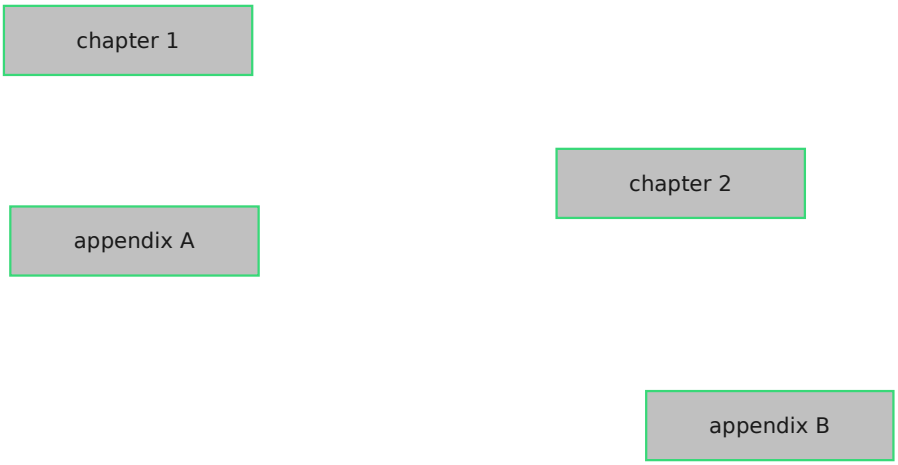
OOo Draw drawing with imagemap-style hyperlinks as object. The only way to include vector-oriented graphics in your document is to include OOo Draw objects. Now using OLE-objects for that purpose imposes cumbersome editing, placing and adjusting. Using the OOo Draw-like functionality of writer lacks Draw styles. The solution is to draw with styles in OOo Draw, group the whole drawing and than paste it into a frame here in OOo writer. This was done in the following example.

Another goodie of this example is that it shows a possibility to create imagemap-style hyperlinks within OOo Draw diagrams that are *usable* in the exported PDF documents.

1 Object demonstration

The green boxes are transparent non-printing copies of one PNG image which have been placed over the inserted OOo Draw diagram and are anchored to the frame. They can be hyperlinked to outline elements of this document or to other targets.

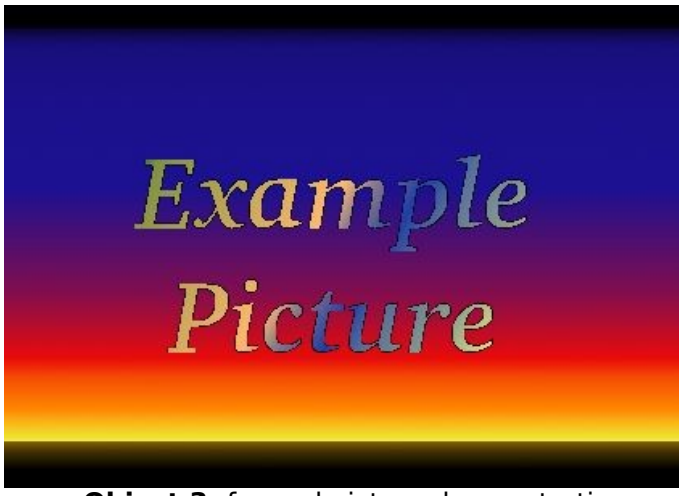
Note further that it is a good idea to place a white image into the background of the frame, which is aligned to the frame. This is to span the desired size of the frame, so lets you work around frame sizing problems and problems with the placement of the frame's title.



Object 2: OOo Draw drawing with imagemap-style hyperlinks

Picture object. This one is easy: a picture within a frame.

1 Object demonstration



Object 3: framed picture demonstration

A Glossary of terms and abbreviations

Summary. There have been special paragraph styles defined for the glossary. Do not use the styles for definition lists because the glossary styles have been adjusted to appear as PDF bookmarks in PDF documents exported from OpenOffice.org.

glossary term 1

This is the definition and explanation of glossary term 1.

glossary term 2

This is the definition and explanation of glossary term 1. The indentation of citations has been adjusted so that you can reasonable use it within the glossary, too:

> Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. Citation text. ►[, p.]

B Source listings

Summary. This appendix chapter will contain source codes developed during the diploma thesis, ordered by program modules. If you want to include long source listings here it might be a good idea to use source highlighting. The best way seems to use an editor which can export highlighted source to HTML (such as KDE's kate), to open the HTML document with OOo writer and then to copy it into your thesis. This does result in hard formatting (not style-based) but this does not hurt here. A short example done with this method is included here.

B.1 http_post()

```
<?php

// adapt these constants and variables to configure the script
// include the path of commands if they reside outside of PHP's PATH
define('LOGFILE_NAME', __FILE__ . '.log.txt');

/** perform a HTTP POST request using the cURL PHP extension
  * @param $server where to POST to, e.g. http://www.example.org
  * @param $path URL part after server name, e.g. '/foo/bar.php'
  * @param $vars array of key/value pairs, maybe nested; or an object
  * @return the content returned by the server, without headers
  */
function http_post($server, $path, $vars) {
  $ch = curl_init();
  curl_setopt($ch, CURLOPT_URL, $server.$path);
  curl_setopt($ch, CURLOPT_POST, 1);
  curl_setopt($ch, CURLOPT_POSTFIELDS, http_build_query($vars));
  curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
  $reply = curl_exec($ch);
  if (curl_errno($ch))
    error_log(
      "ERROR: curl_exec() error no ". curl_errno($ch) . " : " .
        curl_error($ch)."\n",
      3, LOGFILE_NAME
    );
  curl_close($ch);
  return($reply);
}

?>
```

Index of glossary items

Index of objects

Bibliography

Bibliography

[1] K. Wilcox and A. Stephen, "*Are Close Friends the Enemy? Online Social Networks, Self-Esteem, and Self-Control*," Journal of Consumer Research, Forthcoming Columbia Business School Research Paper No. 12-57, Date posted: October 3, 2012.

[2] D. A. Vise and M. Malseed, *The Google Story*. New York City: Dell Publishing, 2005.

[3] H. F. Shantz, *The History of OCR, Optical Character Recognition*. Manchester Center: Recognition Technologies Users Association, 1982.

[4] S. V. Rice, F. R. Jenkins and T. A. Nartker, "*The Fourth Annual Test of OCR Accuracy*," Technical Report 95-03, Information Science Research Institute, University of Nevada, Las Vegas, July 1995.

[5] L. Vincent, "Google Book Search: Document Understanding on a Massive Scale," *International Conference on Document Analysis and Recognition (ICDAR)*, 2007, pp. 819 - 823.

[6] R. Unnikrishnan and R. Smith, "*Combined Script and Page Orientation Estimation Using the Tesseract OCR Engine*," Submitted to International Workshop of Multilingual OCR, 25th July 2009, Barcelona, Spain.

[7] Z. Huang, M. Cmejrek and B. Zhou, "Soft Syntactic Constraints for Hierarchical Phrase-based Translation Using Latent Syntactic Distributions," *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010, p. 138–147.

[8] P. W. Handel, "*Statistical Machine*," United States Patent Office. 1,915,993, Jun. 27, 1933.

[9] A. Kleiner and R. Kurzweil*, A Description of the Kurzweil Reading Machine and a Status Report on its Testing and Dissemination*, Bulletin of Prosthetics Research, vol. 27, no. 10, pp. 72-81, Spring. 1977.

[10] M. Bokser*, Omnidocument Technologies*, Proceedings of the IEEE, vol. 80, no. 7, pp. 1066-1078, Jul. 1992.

[11] ABBY FineReader. "ABBYY FineReader for Personal Use." Internet: http://finereader.abbyy.com/, Date Accessed: April 3, 2013.

[12] Nuance Inc. "OmniPage Professional." Internet: http://www.nuance.com/for-business/by-product/omnipage/professional/index.htm, Date Accessed: April 3 2013.

[13] Iris Products and Technologies. "Introducing the new Readiris 14." Internet: http://www.irislink.com/c2-2115-189/Readiris-14--OCR-Software--Scan--Convert---Manage-your-Documents-.aspx, Date Accessed: April 3, 2013.

[14] Contributor: Bob Stein (uploader to http://archive.org). "New York Times August September 1901 Collection." Internet: http://archive.org/download/NewYorkTimesAugSept1901Collection/New_York_Times_August_September_1901_Part_7_text.pdf, Date Accessed: March 14, 2013.

[15] T. M. Breuel and U. Kaiserslautern, "The hOCR Microformat for OCR Workflow and Results," *Ninth International Conference on Document Analysis and Recognition (ICDAR)*, 2007, pp. 1063 - 1067.

[16] R. Griffin, *Statistics*. London: Macmillon and Co., 1913, pp. 121-122.

[17] F. d’Albe, "On a type-reading optophone," *Proc. Roy. Soc., Lond.*, 1914, p. 90A, 373–375.

[18] G. Tauschek, "*Reading Machine*," United States Patent Office. 2,026,329, May 27, 1929.

[19] M. Martin*, Reading Machine Speaks Out Loud*, Popular Science, vol. , no. 154 (2), p. 125–27, Feb. 1949.

[20] D. Shepard, "*Apparatus for Reading*," United States Patent Office. 2,663,758, Dec. 22, 1953.

[21] J. Leimer, "Design Factors in the Development of an Optical Character Recognition Machine," *IRE Transaction on Information Theory*, 1962, pp. 167 -171.

[22] M. H. Weik. "A Fourth Survey of Domestic Electronic Digital Computing Systems." Internet: http://ed-thelen.org/comp-hist/BRL64-i.html#IBM-1401, Date Accessed: 2013.

[23] IBM Systems Reference Library. 1964.

[24] L. O. Eikvil, "*Optical Character Recognition*," Norwegian Computing Center, 1993.

[25] J. J. Hull and S. L. Taylor, "Document Image Skew Detection: Survey and Annotated Bibliography," *World Scientific*, 1998, pp. 40-64.

[26] R. Smith, "*Apparatus and method for use in image processing*," United States Patent Office. 5,583,949, Dec 10, 1996

[27] S. Li, Q. Shen and J. Sun*, Skew Detection Using Wavelet Decomposition and Projection Profile Analysis*, Pattern Recognition Letters, vol. 28, no. 5, p. 555–562, Jan. 2007.

[28] W. Postl, "Detection of linear oblique structures and skew scan in digitized documents," *Proc. 6th Int. Conf. Pattern Recognition*, 1986, p. 687–689.

[29] S. N. Srihari and V. Govindaraju*, Analysis of Textual Images Using the Hough Transform*, Machine Vision and Applications, vol. 2, no. 1, pp. 141-153, Jan. 1989.

[30] R. Smith*, A Simple and Efficient Skew Detection Algorithm via Text Row Accumulation*, Proc. of the 3rd Int. Conf. on Document Analysis and Recognition, vol. 2, no. 1, pp. 1145-1148, Jan. 1995.

[31] R. Smith, "An overview of the Tesseract OCR engine," *Proc. Int. Conf. Document Anal. Recognit.*, 2007, pp. 629 - 633.

[32] S. S. Bukhari, F. Shafait and T. M. Breuel, "Coupled snakelet model for curled textline segmentation of camera-captured document images," *Proc. 10th Int. Conf. on Document Analysis and Recognition*, 2009, pp. 33-53.

[33] L. Likforman-Sulem, A. Zahour and B. Taconet*, Text line segmentation of historical documents: a survey*, Int. J. Doc. Anal. Recogn. , vol. 9, no. 2, p. 123–138, Jan. 2007.

[34] R. Smith, "*Tesseract OCR Engine: What it is, where it came from, where it is going.*," OSCON 2007

[35] S. Mori, C. Y. Suen and K. Yamamoto*, Historical Review of OCR Research and Development*
, Proceedings of the IEEE, vol. 80, no. 7, pp. 1029-1058, Jul. 1992.

[36] S. Mori, N. Hirobumi and Y. Hiromitsu, *Optical Character Recognition*. New York: Wiley & Sons, Inc., 1999, pp. 193-367.

[37] R. Smith*, History of the Tesseract OCR Engine: What Worked and What Didn't. How to Build a World-Class OCR Engine in Less Than 20 Years*
, SPIE-IS&T, vol. 8658, no. 02-1, p. 12, Feb. 2013.

[38] F. L. Alt*, Digital pattern recognition by moments*, J. Assoc. Computing Machinery, vol. 9, no. 1, pp. 240-258, Jan. 1962.

[39] R. Casey, "*Moment normalization of hand printed characters*," IBM J. Res. Dev. (1970), pp. 548–557

[40] AForge.NET. "Blobs Processing." Internet: http://www.aforgenet.com/framework/features/blobs_processing.html, Date Accessed: 2013.

[41] M. D. McIlroy*, Development of a spelling list*, IEEE Trans on Communications, vol. 30, no. 1, p. 91–99, Jan. 1982.

[42] G. Nagy*, At the frontiers of OCR*, Proc. IEEE, vol. 80, no. 2, p. 1093–1100, Jul. 1992.

[43] Y. Y. Tang, C. D. Yan and C. Y. Suen*, Document Processing for Automatic Knowledge Acquisition*, IEEE Transactions on Knowledge and Data Engineering, vol. 6, no. 1, pp. 3-21, Feb. 1994.

[44] A. M. Namboodiri and A. Jain, "*Document Structure and Layout Analysis*," Advances in Pattern Recognition. Springer-Verlag, London. 2007

[45] M. Nadler*, A survey of document segmentation and coding techniques*, Comput. Vision Image Graphics Process., vol. 28, no. 2, p. 240–262, Nov. 1984.

[46] R. Haralick, "Document Image Understanding: Geometric and Logical Layout," *Proc. IEEE Conf. Computer Vision and Pattern Recognition. Seattle*, 1994, pp. 385-390.

[47] Y. Y. Tang, S. W. Lee and C. Y. Suen*, Automatic document processing: A survey*, Pattern Recognition, vol. 29, no. 12, pp. 1931-1952, Dec. 1996.

[48] S. Mao, A. Rosenfeld and T. Kanungo*, Document structure analysis algorithms: A literature survey*, Document Recognition and Retrieval, vol. 5010, no. 10, pp. 197-207, Jan. 2003.

[49] N. Chen and D. Blostein*, A survey of document image classification: problem statement, classifier architecture and performance evaluation*, Int. J. Doc. Anal. Recognit, vol. 10, no. 1, pp. 1-16, May. 2007.

[50] S. Marinai, "Introduction to document analysis and recognition," *Machine learning in document analysis and recognition, Berlin, Germany: Springer*, 2008, p. 1–20.

[51] G. Nagy, S. Seth and M. Viswanathan*, A prototype document image analysis system for technical journals*, Computer, vol. 25, no. 1, p. 10–22, Jan. 1992.

[52] Y. N. Elglaly, F. Quek, T. Smith-Jackson and D. Dhillon, "Touch-Screens are Not Tangible: Fusing Tangible Interaction with Touch Glass in Readers for the Blind," *ACM International Conference on Tangible, Embedded and Embodied Interaction (TEI), Barcelona, Spain*, 2013, pp. 245-252.

[53] F. Goudail, "*Statistical Image Processing Techniques for Noisy Images*," New York: Klewer Academic Plenum Publishers, 2004.

[54] R. Miller. "Ink-Jet Basics." Internet: http://www.thetonesystem.com/inkjet_basics.html, Date Accessed: 2013.

[55] O. G. Guleryuz*, A multiresolutional algorithm for halftone detection*, Proc. SPIE Image and Video Communications and Processing, vol. 5685, no. 1, pp. 1098 -1105, Jan. 2005.

[56] N. Otsu*, A threshold selection method from gray-level histograms*, IEEE Trans. Sys., Man., Cyber, vol. 9, no. 1, pp. 62-66, Jan. 1979.

[57] W. R. Smith, "Hybrid Page Layout Analysis via Tab-Stop Detection," *Proceedings of the 2009 10th International Conference on Document Analysis and Recognition*, 2009, pp. 241-245.

[58] B. Xie and G. Agam*, Boosting based text and non-text region classification*, Document Recognition and Retrieval Proc. SPIE, vol. XVIII, no. 1, pp. 1-9, Jan. 2011.

[59] A. Gourdol. "CSS3 regions: Rich page layout with HTML and CSS3." Internet: http://www.adobe.com/devnet/html5/articles/css3-regions.html, Date Accessed: 2013.

[60] T. Pavlidis and J. Zhou*, Page segmentation and classification*, Graphical Models and Image Processing, vol. 54, no. 1, pp. 484-496, Jan. 1992.

[61] H. S. Baird, H. Bunke and P. S. P. Wang, "Background structure in document images," *Document Image Analysis, World Scientific, Singapore*, 1994, p. 17–34.

[62] G. Nagy and S. Seth, "Hierarchical representation of optically scanned documents," *Proc. of the 17th Conf. on Pattern Recognition*, 1984, p. 347–349.

[63] H. S. Baird, S. E. Jones and S. J. Fortune, "Image Segmentation by Shape-Directed Covers," *Proceedings of International Conference on Pattern Recognition, Atlantic City, NJ*, 1990, p. 820–825.

[64] T. M. Breuel, "Two Geometric Algorithms for Layout Analysis," *Proceedings of the Fifth International Workshop on Document Analysis Systems, Princeton, NY*, 2002, p. 188–199.

[65] Intelligent Systems Laboratory. "UW-III English/Technical Document Image Database ." Internet: http://www.science.uva.nl/research/dlia/datasets/uwash3.html, Date Accessed: 2013.

[66] F. Wahl, K. Wong and R. Casey*, Block Segmentation and Text Extraction in Mixed Text/Image Documents*, Graphical Models and Image Processing, vol. 20, no. 1, p. 375–390, Jan. 1982.

[67] J. Higashino, H. Fujisawa, Y. Nakano and M. Ejiri, "A knowledge-based segmentation method for document understanding," *Proc. 8th Int. Conf. on Pattern Recognition*, 1986, pp. 745-748.

[68] A. Dengel and F. Dubiel*, Computer understanding of document structure*, International Journal of Imaging Systems and Technology, vol. 7, no. 1, p. 271–278, Jan. 1996.

[69] J. P. Bixler, "Tracking text in mixed-mode document," *Proc. ACM Conference on Document Processing System*, 1998, pp. 177-185.

[70] T. Pavlidis*, Algorithms for Graphics and Image Processing*, ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik, vol. 63, no. 8, p. 395, Jan. 1983.

[71] L. O'Gorman*, The Document Spectrum for Page Layout Analysis*, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 15, no. 11, pp. 162-173, Nov. 1993.

[72] S. Arya, T. Malamatos and D. M. Mount, "Space-efficient approximate Voronoi diagrams," *Proc. 34th ACM Sympos. Theory Comput.*, 2002, p. 721–730.

[73] Wikipedia. "Voronoi diagram." Internet: http://en.wikipedia.org/wiki/Voronoi_diagram, Date Accessed: 2013.

[74] K. Kise, A. Sato and M. Iwata*, Segmentation of Page Images Using the Area Voronoi Diagram*, Computer Vision and Image Understanding, vol. 70, no. 3, pp. 370 -382, Jun. 1998.

[75] E. G. Johnston*, Printed Text Discrimination*, Computer Graphics and Image Processing, vol. 3, no. 1, p. 83–89, Mar. 1974.

[76] D. S. Bloomberg, "Multiresolution Morphological Approach to Document Image Analysis," *1st ICDAR*, 1991, pp. 963-971.

[76] M. Benjelil, S. Kanoun, R. Mullot and A. M. Alimi, "Steerable pyramid based complex documents images segmentation," *10th International Conference on Document Analysis and Recognition*, 2009, pp. 833-837.

[77] A. K. Jain and S. K. Bhattacharjee*, Text segmentation using Gabor filters for automatic document processing*, Mach. Vis. Appl., vol. 5, no. 3, pp. 169-184, Jul. 1992.

[78] D. Niyogi and S. N. Sihari, "*Using domain knowledge to derive the logical structure of documents*," Proceedings of SPIE Document Recognition and Retrieval III, San Jose. January, 1996.

[79] Y. Ishitani, "Logical structure analysis of document images based on emergent computation," *Proceedings of International Conference on Document Analysis and Recognition*, 1999, p. 189–192.

Colophon

<A colophon (literally, end stroke) is an inscription at the end of a written work, containing facts about its production. It may name artists, printers etc. and discuss typographic and technical details such as typefaces and papers.>

Attached electronic data

Description of attached files and folders[5]

 <file name> <description>
 <folder name> <description>
 <file name> <description>
 <folder name> <description>
 <folder name> <description>
 <file name> <description>
 <folder name> <description>
 <file name> <description>
 <file name> <description>
 <file name> <description>
 <folder name> <description>

[5]You might find additional auxiliary files for download at this thesis' web location.

Accessing the attached electronic data

The paper version of this thesis should contain an envelope with an optical data medium (CD or DVD) here.

The digital version of this thesis contains the content of this medium as attachments to the PDF file. If your PDF viewer cannot handle attachments you may access these files at this thesis' web location.