

1-1-1996

Algorithms for document image skew estimation

Andrew David Bagdanov
University of Nevada, Las Vegas

Follow this and additional works at: <https://digitalscholarship.unlv.edu/rtds>

Repository Citation

Bagdanov, Andrew David, "Algorithms for document image skew estimation" (1996). *UNLV Retrospective Theses & Dissertations*. 3231.

<http://dx.doi.org/10.25669/ao3r-anf1>

This Thesis is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Thesis has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Algorithms for Document Image Skew Estimation

by

Andrew D. Bagdanov

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science

in

Computer Science

Department of Computer Science
University of Nevada, Las Vegas
December 1996

UMI Number: 1383141

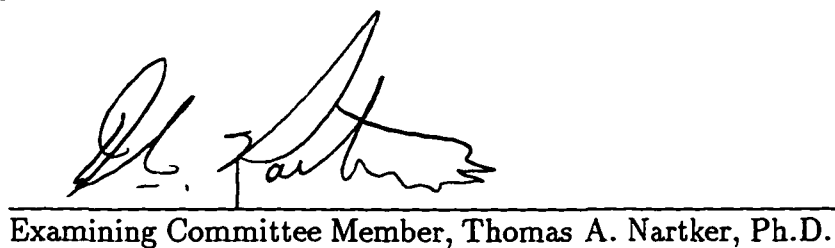
UMI Microform 1383141
Copyright 1997, by UMI Company. All rights reserved.


**This microform edition is protected against unauthorized
copying under Title 17, United States Code.**

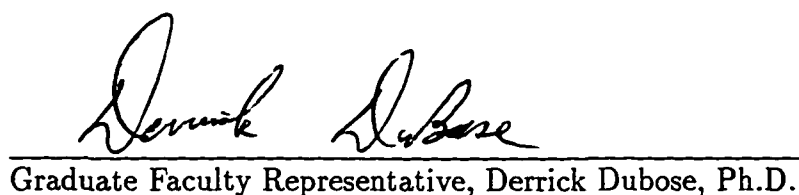
UMI
300 North Zeeb Road
Ann Arbor, MI 48103

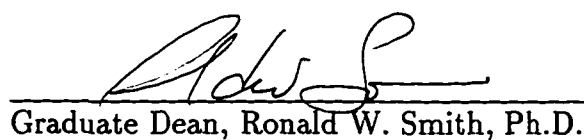
The thesis of Andrew D. Bagdanov for the degree of Master of Science in Computer Science is approved.


Chairperson, Junichi Kanai, Ph.D.


Examining Committee Member, Thomas A. Nartker, Ph.D.


Examining Committee Member, Lawrence L. Larmore, Ph.D.


Graduate Faculty Representative, Derrick Dubose, Ph.D.


Graduate Dean, Ronald W. Smith, Ph.D.

University of Nevada, Las Vegas
December, 1996

ABSTRACT

A new projection profile based skew estimation algorithm was developed. This algorithm extracts fiducial points representing character elements by decoding a JBIG compressed image without reconstructing the original image. These points are projected along parallel lines into an accumulator array to determine the maximum alignment and the corresponding skew angle. Methods for characterizing the performance of skew estimation techniques were also investigated. In addition to the new skew estimator, three projection based algorithms were implemented and tested using 1,246 single column text zones extracted from a sample of 460 page images. Linear regression analyses of the experimental results indicate that our new skew estimation algorithm performs competitively with the other three techniques. These analyses also show that estimators using connected components as a fiducial representation perform worse than the others on the entire set of text zones. It is shown that all of the algorithms are sensitive to typographical features. The number of text lines in a zone significantly affects the accuracy of the connected component based methods. We also developed two aggregate measures of skew for entire pages. Experiments performed on the 460 unconstrained pages indicate the need to filter non-text features from consideration. Graphic and noise elements from page images contribute a significant amount of the error for the JBIG algorithm.

CONTENTS

ABSTRACT	iii
CONTENTS	iv
LIST OF FIGURES	vi
LIST OF TABLES	vii
ACKNOWLEDGMENTS	viii
 CHAPTER 1 INTRODUCTION	 1
1.1 A Model for Document Processing Systems	2
1.2 Document Image Skew	3
1.3 Efficiency of Skew Estimation	4
1.4 Measures of Skew Estimation Accuracy	5
 CHAPTER 2 A SURVEY OF SKEW ESTIMATION ALGORITHMS	 7
2.1 Text Row Accumulation	7
2.2 Recursive Morphological Transformation	9
2.3 Local Region Complexity	10
2.4 Fourier Transformation	11
2.5 Hough Transformation	12
 CHAPTER 3 PROJECTION PROFILE BASED ALGORITHMS	 15
3.1 The Basic Projection Profile Algorithm	16
3.2 A Framework for Projection Profile Based Algorithms	17
3.3 The Method of Baird	18
3.4 The Method of Nakano, et. al.	18
3.5 The Method of Postl	20
3.6 The Method of Bloomberg, et. al.	20
3.7 The Method of Spitz	22
3.7.1 CCITT G4 Compression	23
3.7.2 White Pass Code Detection	24
 CHAPTER 4 ESTIMATION OF SKEW IN JBIG IMAGES	 26
4.1 Arithmetic Coding Basics	27
4.2 The JBIG Algorithm	28
4.3 Detecting White Pass Codes in JBIG Images	30
4.4 A Skew Estimator for JBIG Compressed Images	31

CHAPTER 5	ZONE BASED EVALUATION	33
5.1	Procedures and Methodology	33
5.1.1	Test Data	34
5.1.2	Experiment Methodology	35
5.2	Analysis	35
5.2.1	Error Distribution	35
5.2.2	Regression Analysis	36
5.2.3	Sensitivity to Typographical Features	38
5.2.4	Non-parametric Measures	40
CHAPTER 6	PAGE BASED EVALUATION	43
6.1	Test Data	43
6.2	Aggregate Measures of Skew	44
6.3	Experiment Methodology	45
6.4	Results and Discussion	45
6.4.1	Error Distribution	45
6.4.2	Regression Analysis	45
6.4.3	Sensitivity to Typographical Features	46
CHAPTER 7	CONCLUSIONS AND FUTURE WORK	50
7.1	Skew Estimation Algorithms	50
7.2	Evaluation of Skew Estimation Algorithms	51
7.3	Estimating Skew in Compressed Images	52
7.4	Contributions	54
APPENDIX A	PASS CODE AUTOMATON	55
APPENDIX B	INITIAL SURVEY OF ALGORITHMS	57
BIBLIOGRAPHY		58

LIST OF FIGURES

1.1	A simple model for document processing systems	2
1.2	Sample pages with positive and negative skew	3
2.1	Algorithm for computing initial rows in text row accumulation	8
3.1	Variables used to determine CCITT G4 coding mode	23
3.2	A white pass mode coding situation	24
4.1	Example of Interval Subdivision for Arithmetic Encoding	27
4.2	Two-line Template for Determining Coding Context	29
4.3	Three-line Template for Determining Coding Context	29
4.4	Automaton to Detect White Passes from JBIG Contexts	30
5.1	The distribution of skew angles in test sample	35
5.2	The Distribution of Absolute Error for Postl Algorithm	36
5.3	Raw scatterplots and regression lines for each algorithm over the entire set of text zones	37
5.4	Effect of textlines on accuracy. Each point represents the correlation of the algorithm on zones of greater than x textlines.	39
5.5	Effect of textlines on accuracy. Each point represents the median of absolute error of the algorithm on zones of greater than x textlines.	40
5.6	Effect of textlines and zone width on accuracy. Each point (x, y) represents the correlation on zones of greater than x textlines and width greater than y	41
6.1	Scatterplot of Weighted Average vs. Dominant Skew	44
6.2	The Distribution of Absolute Error for JBIG Algorithm	45
6.3	Scatterplots of Dominant Skew vs. Estimates	47
6.4	Scatterplots of Weighted Average Skew vs. Estimates	48
6.5	Effect of textlines on accuracy. Each point represents the correlation of algorithm on pages of greater than x textlines.	49

LIST OF TABLES

2.1	Accuracy statistics for text row accumulation algorithm	9
2.2	Accuracy values for Hough transformation algorithm	14
5.1	The four algorithms evaluated and parameters for each	33
5.2	Median of Absolute Error for All Sample Zones	36
5.3	Summary of linear regression for all sample zones	38
5.4	Summary of linear regression for zones of type <i>Text</i>	38
6.1	Summary of Linear Regressions for Pages and Dominant Skew	46
6.2	Summary of Linear Regressions for Pages and Weighted Average Skew	46
B.1	Algorithms in Question and Associated Parameters	58
B.2	Typographical Features Possibly Affecting Skew Estimation	59

ACKNOWLEDGMENTS

This research was supported in part by a grant from the United States Department of Energy.

I would like to thank Dr. George Nagy (RPI), Dr. E. A. Yfantis (UNLV), Dr. Stephen Rice (ISRI), and Larry Spitz (Daimler Benz) for the many helpful suggestions and words of encouragement freely given throughout this project.

I would also like to thank my advisor Dr. Junichi Kanai for his constant encouragement and support throughout this effort. I am also indebted to Dr. Thomas Nartker, Dr. Lawrence Larmore, and Dr. Derrick Dubose for agreeing to serve on my committee.

Lastly, but not least, I would like to thank all of the staff at ISRI for their support. I am particularly grateful to Diego Vinas for the preparation of the ground truth skew data essential to my experiments.

CHAPTER 1

INTRODUCTION

We live in an age of rapidly evolving and changing information. This fact and new information technology have prompted the development of automated document conversion systems that bridge the gap between the hardcopy information society of 50 years ago and the electronic information age of today. The goal of such *automated document processing systems* is quite simple:

- Given a hardcopy document, create images of the document pages using some type of scanning device.
- Once scanned, *optical character recognition*¹ (OCR) is performed on the pages, converting them to some character coded (e.g., ASCII, UNICODE, etc.) version of the page.
- The text version of the document is then added to a *text retrieval* (TR) system for recall at a later time.

There are many advantages to using an electronic retrieval system as opposed to the hardcopy original. First, paper document collections are fragile, difficult to copy, and require enormous ammounts of physical space for storing large collections. Also, paper documents are difficult to index, making the task of retrieving relevant documents quite troublesome. Lastly, searching a paper document collection requires the searcher to be in the same physical location as the documents, while an electronic database of documents can be searched and read remotely.

¹Note: all documents considered are *machine printed*.

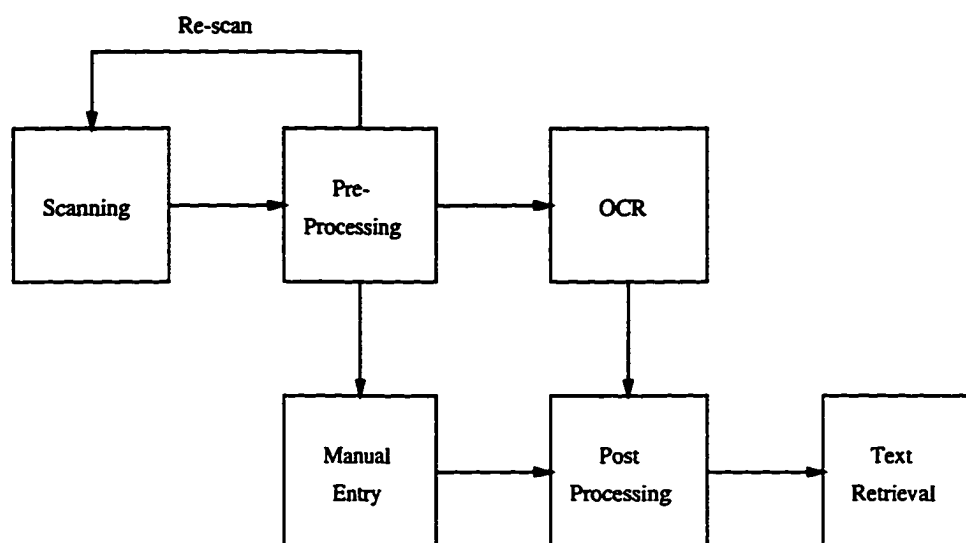


Figure 1.1: A simple model for document processing systems

1.1 A Model for Document Processing Systems

Figure 1.1 gives a block diagram for a simple document conversion system. In this simple model each subsystem consists of the following:

1. Scanning

Each document page is placed in a scanning device, which provides a digital image to the pre-processing subsystem.

2. Pre-processing

In this stage the page is enhanced to correct various image degradations, such as broken characters, touching characters, and skew. In some cases, the pre-processor may decide the page is so badly degraded that it must be re-scanned. In extreme cases, the pre-processor may decide that the page is unsuitable for OCR. Such pages will be manually re-keyed.

3. OCR

Here the image is converted from a digital array of pixels into a character coded text file.

4. Manual Entry

In this stage, the page is entirely re-keyed by a human operator.

5. Post-processing

After the page has been recognized (or re-keyed) the resulting text is corrected for any errors and possibly marked up in preparation for indexing in a text retrieval system.

6. Text Retrieval

Finally, the text for the page is added to a text retrieval system for later recall.

One of the main problems with such systems is that errors occurring early in the process propagate forward in the system, causing errors to accumulate in subsequent stages. Consequently, it is important to detect any such errors at the earliest possible stage in the document conversion process. In addition, it is important to characterize and quantify the performance and reliability of any component designed to detect or correct these errors.

1.2 Document Image Skew

One type of image degradation commonly occurring is *skew*. In the simplest sense, *document image skew* can be defined as rotation of the page image about a fixed point. It arises most frequently from pages that are improperly aligned on the scanner bed when initially scanned, or when the original hardcopy is poorly aligned when photocopying. A skew estimation algorithm should return an angle corresponding to the degree of rotation of the text on that page. Note that we are primarily concerned with the skew angle of *text* on the page, as opposed to other page elements such as drawings.

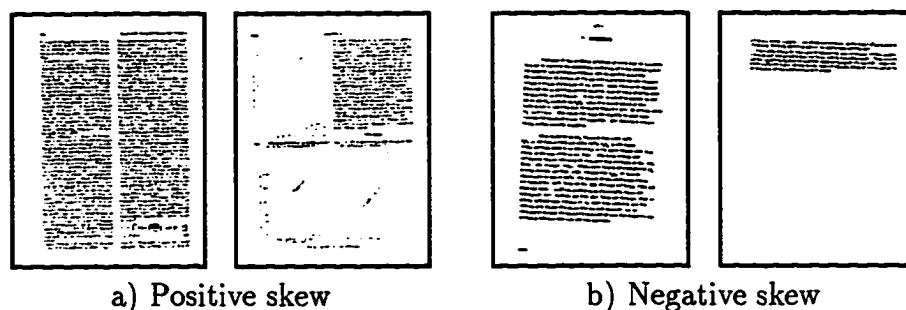


Figure 1.2: Sample pages with positive and negative skew

For the purposes of our experiments, the origin of the coordinate system is at the geometric center of the page image. A positive skew angle indicates counter-clockwise rotation, and a negative skew angle indicates clockwise rotation (see Figure 1.2).

In an ideal world, this definition would suffice. However, some skew phenomena are not captured by this simple formulation. For example, images that contain curved baselines cannot easily be classified. Also, multi-column pages in which the columns possess independent skew angles present problems. In many cases it is not clear how skew should be defined, and as a result it is difficult to specify the desired output for an algorithm designed to estimate the skew angle.

We performed two types of experiments, zone based and page based. For our zone based experiments, we separate all pages into single column text zones and perform the estimation procedures on each zone independently. In this way, we can easily assign a single skew angle to each zone. For the page based experiment, we devised aggregate measures of skew based on the individual manual skew estimates for each zone.

According to Bloomberg [2], correct detection and/or correction of skew in the pre-processing stage can improve text recognition, simplify layout analysis, and improve baseline determination at the OCR stage. Also, O’Gorman and Kasturi [16] indicate that most OCR and page analysis algorithms *depend* on an input page with zero skew, and that skew estimation and correction must be performed before making use of these algorithms. Many algorithms and techniques have been applied to this problem. Chapter 2 provides a thorough survey of skew estimation algorithms appearing in the literature. Chapter 3 presents a limited model of skew estimation that captures the essence of the *projection profile* skew estimation algorithm.

1.3 Efficiency of Skew Estimation

It is difficult to perform quantitative experiments of the speed efficiency of skew estimation algorithms. It should be noted that all algorithms evaluated are the current author’s implementations, and the emphasis is placed primarily on *correctness* rather than efficiency. Thus, a fair comparison of speed efficiency could not be performed without the original author’s implementation.

However, some skew estimation algorithms operate on compressed images, and are thus more efficient than those operating on full size images. In Chapter 4 a method for detecting the skew angle of JBIG compressed images is proposed and evaluated.

1.4 Measures of Skew Estimation Accuracy

As discussed in Section 1.1, it is important to characterize the performance of any skew estimation algorithm over a broad range of image features. Most of the papers on the topic of skew estimation provide some experimental results for proposed methods, but there is no universally accepted measure of overall accuracy for this type of algorithm. Additionally, many of the results given provide no details regarding the sample document images and parameter settings used for experimental trials. Consequently, it is difficult to objectively determine the relative performance of a specific algorithm.

Some measures of estimation accuracy that appear in the literature are:

- *Average error.* This measure computes the sample mean of the error in all observations. It is not a statistically significant measure, since positive and negative errors will offset one another.
- *Root-mean-square (RMS) error* [20]. This measure computes the mean of observed absolute (unsigned) error. While more significant than the raw average, the mean of absolute error is still of limited use and is sensitive to outliers.
- *Cumulative absolute error probability distribution* [7, 2]. This measure uses the RMS error in estimation to compute the cumulative distribution: $\text{Prob}(|\theta_{\text{True}} - \theta_{\text{Est}}| \leq x)$. Such a measure allows us to compute the interval which contains 95% of observed absolute errors.
- *Regression analysis* [20]. This method computes the linear regression of the true skew angle versus skew angle estimate. The correlation can then be used as an accuracy measure.

Few papers in this area deal with the problems involved with defining the skew angle for entire pages. In chapters 5 and 6 some alternate methods for characterizing

the performance of skew estimation are proposed and evaluated, and the results of a series of experiments are discussed and analyzed with the new methods.

CHAPTER 2

A SURVEY OF SKEW ESTIMATION ALGORITHMS

Many techniques have been applied to the problem of estimating skew in document images. This chapter gives a brief description of each algorithm as it occurs in the literature. Algorithms based on the projection profile approach are presented in chapter 3.

2.1 Text Row Accumulation

In Smith [20] an algorithm based on text row accumulation is given. The algorithm is designed to determine the skew angle of any document image, even in the presence of broken/touching characters, speckle, and skew in excess of 20 degrees. The x-coordinate of Smith's coordinate system corresponds to the horizontal axis of the page image, and

The steps of the algorithm are described as follows:

1. **Perform connected component analysis of the image.**

Black connected components are henceforth referred to as *blobs*. The bounding box of a blob is considered representative of the blob itself.

2. **Filter blobs.**

Blobs that are considered too small to represent whole text elements are removed from consideration. This filter is designed to select the subset of blobs that most accurately represent the main body text. The filter described by Smith removes

```

MAKE-INITIAL-ROWS( $B, \alpha$ )
 $\alpha \in (0.5, 0.7)$ ,  $B$  a sorted list of blobs
 $Shift := 0$ 
for each blob  $\in B$  do
    Find row with most vertical overlap with blob (offset by  $Shift$ ).
    if there is no overlapping row then
        Make a new row, put current blob in it.
        Record top and bottom of blob as top and bottom of row.
    else
        Add the blob to the row.
        Expand top and bottom limits of row with top and bottom of blob.
        Update the running average  $y$  shift with bottom of blob:
             $Shift = \alpha Shift + (1 - \alpha) NewShift$ ,
            where  $NewShift$  is the shift from the previous to current blob.
    endif
endfor

```

Figure 2.1: Algorithm for computing initial rows in text row accumulation

any blobs smaller than a fixed height, and retains all blobs with height between the 20th and 95th percentile.

3. Sort blobs.

The blobs are sorted in ascending order using the x-coordinate of the left edge of the blob.

4. Make initial rows.

See figure 2.1. The sorted blobs allow for the maintenance of a running average y shift used to vertically shift blobs before inserting into rows. This helps rows collect blobs that actually belong to them, rather than creating new rows.

5. Fit baselines.

A least median of squares fit is used to eliminate effects of outliers. The median gradient of the baselines provides the estimated skew angle.

Smith also provides experimental results for the new algorithm. The technique was tested on a database of 400 page images scanned at 400dpi and compared against

Algorithm	Mean Angular Error	RMS Angular Error
Including Outliers		
Text Row	0.216	2.83
Simple_Baird	-0.148	7.69
Fast_Baird	-0.284	11.7
Excluding Outliers		
Text Row	-0.0034	0.0718
Simple_Baird	0.0023	0.139
Fast_Baird	0.0061	0.132

Table 2.1: Accuracy statistics for text row accumulation algorithm

Baird’s [1] projection profile algorithm. Two versions of the Baird algorithm were tested. Simple_Baird searches an entire interval of angles, while Fast_Baird applies a hierarchical search of the interval. Both algorithms were run on the original page image and on three rotations of the original image, with the rotations selected randomly from $(-28.6, 28.6)$ degrees. The accuracy statistics from [20] are summarized in Table 2.1. The mean angle error and RMS angle error are reported here for the entire image set and the image set without outliers.

2.2 Recursive Morphological Transformation

Chen [7] gives a skew estimation algorithm that uses recursive morphological transformations. The algorithm first detects all text lines on the page, and then obtains a skew estimate for each line. These independent skew estimates are then combined to create the composite, or aggregate, skew angle for the entire page.

First, the algorithm applies a *recursive closing transform* (RCT) on the image. The idea is to pick the structuring element of the RCT so that the inter-character gaps are filled while the inter-line gaps remain open. I_c denotes the image after applying the RCT.

Next, a *recursive opening transform* (ROT) is applied to I_c . The structuring element of the ROT is selected so that ascenders and descenders are eliminated. I_o denotes the final image after applying the ROT to I_c . For details of the ROT and RCT procedures, see [8] and [5].

After applying the RCT and ROT transforms, the algorithm then does a connected

component analysis of I_o . If the structuring elements are chosen wisely, this will effectively extract the textlines from the image.

Now lines are fitted to the points in each connected component. If

$$C_k = \{(x_0, y_0), \dots, (x_{n_k}, y_{n_k})\}$$

denotes the points of the k th connected component, the direction and variance of the line minimizing the sum of the squared error can be expressed as a function of the second order spatial moments (denoted μ_{xx} , μ_{yy} , and μ_{xy}) of C_k .

Specifically, the direction of the line, ϕ_k is

$$\phi_k = -\frac{1}{2} \arctan \left(\frac{2\mu_{xy}}{\mu_{xx} - \mu_{yy}} \right)$$

and the variance is

$$\sigma_{\phi_k} = \frac{1}{n_k - 2} \times \frac{\mu_{xx} + \mu_{yy} + 2\mu_{xy} \sin 2\phi_k - (\mu_{xx} - \mu_{yy}) \cos 2\phi_k}{\mu_{xx} + \mu_{yy} - 2\mu_{xy} \sin 2\phi_k + (\mu_{xx} - \mu_{yy}) \cos 2\phi_k}.$$

The last step of the procedure is to combine all estimates into a single aggregate skew estimate for the entire page. This is done using a Bayesian estimate. If there are L textlines on the page, the Bayesian estimate of ϕ is computed as:

$$\phi = \frac{\sum_{i=0}^L \frac{1}{\sigma_{\phi_i}} \phi_i}{\sum_{i=0}^L \frac{1}{\sigma_{\phi_i}}}$$

In experimental trials, the RCT and ROT transforms were trained on a set of 12,617 real world and synthetic training images to determine the best possible structuring elements. All tests of skew estimation were then conducted on the same set of images, using these “optimal” structuring elements. The tests indicate that when using a 2x3 structuring element, approximately 95% of the skew angles for synthetic images are estimated to within 0 degrees of absolute error, while 95% of the skew angles for real world images are estimated to within 0.25 degrees of absolute error. Different results were obtained for different sized structuring elements.

2.3 Local Region Complexity

Ishitani [12] presents a skew estimation algorithm based on a measure of local region complexity. The complexity parameter enables the algorithm to correctly isolate

regions of a page image that are likely to contain text, and to detect skew angles on pages containing multiple text blocks that are skewed at different angles.

The measurement of local region complexity is based on the distribution of variance in the horizontal projection of white to black transitions when scanned at various angles within a specified search interval. The general estimation algorithm is described as follows:

1. For an angle θ , scan the image along parallel lines at angle θ .
2. For each line in the scan, compute N_i as the total number of white to black transitions in that line.
3. The complexity variance $V(\theta)$ is then computed as

$$V(\theta) = \frac{1}{n} \sum_{i=1}^n (N_i - M)^2$$

where n is the number of lines scanned, and

$$M = \frac{\sum_{i=1}^n N_i}{n}$$

4. The values of $V(\theta)$ are computed for all angles in the desired search interval, and the global skew estimate is the value θ which maximizes $V(\theta)$.

The algorithm first subdivides the image into local regions by scanning a sequence of circular regions and computing the complexity for the region at zero degrees, $V(0)$. A number of candidates with high complexity are retained, and the algorithm described above is used to determine the skew angle of each local region.

The author presents experimental results from a test involving 40 document images scanned at 300dpi and rotated up to ± 30 degrees. The average accuracy is reported as ± 1.2 degrees. It is unclear whether the author refers to the actual average accuracy or the RMS accuracy.

2.4 Fourier Transformation

Hase and Hoshino [9] give an algorithm based on the discrete two dimensional Fourier transform. The algorithm is simple and efficient. However, it seems to be more appropriate for greyscale images rather than binary.

The algorithm is given as follows:

1. Compute the discrete 2-d Fourier transform as:

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(s, x) e^{-j2\pi(ux+vy)}$$

where N is the image width and height, and $f(x, y)$ is the image function.

2. Find the maximum element in $F(u, v)$ excluding the origin. Let (u_1, v_1) be the coordinates of this maximal element.
3. The skew estimate, θ_{Est} is computed as:

$$\theta_{Est} = \arctan \frac{u_1}{v_1}$$

The authors claim accuracy of ± 2.0 degrees for this technique, but provide no experimental results for the skew estimation algorithm. In the preliminary experiments we performed we were unable to obtain consistent results for bi-level images, and the algorithm was excluded from future experiments.

2.5 Hough Transformation

A method similar to the projection profile algorithm is given by Hinds *et al* [10]. This algorithm uses a sinusoidal Hough transformation as opposed to a linear projection transformation. It uses the set of vertical black run lengths as a set of representative points. To eliminate the effects of long black runs (possibly due to graphic regions), only runs with length in the range $[1, 25]$ for images digitized at 75 dpi are considered. These representatives are then projected into a sequence of accumulator arrays. Essentially, this procedure transforms the image into $\rho\theta$ space (Hough space). Hough space is represented as a two dimensional array $A[\rho, \theta]$, where $\rho \in [0, \sqrt{w^2 + h^2}]$ and $\theta \in [\theta_{min}, \theta_{max}]$. The interval $[\theta_{min}, \theta_{max}]$ represents the range of angles to search for the skew estimate, w and h are the image width and height.

The algorithm for transforming the set of black runs to Hough space is as follows:


```

HOUGH-PROJECTION( $R, w, h$ )
 $R$  = list of vertical black runs,  $w$  = image width,  $h$  = image height
for  $r \in R$ 
  for  $\theta \in [\theta_{min}, \theta_{max}]$ 
     $\rho = r_x \cos \theta + y_y \sin \theta$ 
     $A[\rho, \theta] += r_{length}$ 
  endfor
endfor

```

The skew estimate for the page, θ_{est} , can then be computed by searching the accumulator array, A , for the cell $A[\rho, \theta_{est}]$ with the maximal element.

The authors provide a few experimental results. A sample of thirteen pages deemed to be representative of government forms was selected for the experiment. All images were scanned at 300 dpi, reduced to 75 dpi, and thresholded. The thirteen images were grouped into five categories on the basis of the presence of typographical features. The five categories are:

- Simple text (single font)
- Multiple font text
- Text and line drawings
- Text and greyscale images
- Forms

One nice feature of this grouping is that the performance of the algorithm can be characterized across a range of typographical features rather than on a homogeneous set of document images. This type of analysis is rare in the literature. Unfortunately, the number of images is too small to make any meaningful statistical assertions about the performance of the algorithm.

The accuracy results from the experiment are shown in Table 2.2. The authors do not define what they mean by “correctly determined” skew angle, and do not provide an average measure of accuracy. The fact that the algorithm “correctly determined” the skew angle on all pages begs the question of how this measurement is made. Also, they indicate that none of the sample pages were severely skewed.

Category	Simple Text	Multiple Fonts	Line Drawing	Grey	Forms
# of Pages	2	2	2	3	4
# Correct	2	2	2	3	4

Table 2.2: Accuracy values for Hough transformation algorithm

Preliminary experiments with this algorithm we performed indicated poor performance. It is unclear if this is due to some misinterpretation of implementation issues, or to a problem with the proposed technique itself.

CHAPTER 3

PROJECTION PROFILE BASED ALGORITHMS

The most common approach to skew estimation is the *projection profile* technique. It is similar to the Hough transformation approach, but instead of a sinusoidal projection, a linear projection into a partitioned accumulator array is used. The main idea is to project the set of representative (fiducial) points along parallel lines oriented at some search angle. Ideally, the fiducial points representative of characters belonging to the same text line will be projected into the same bin when the projection is made at the correct skew angle.

Several projection profile based algorithms appear in the literature, but they share several common characteristics. For example, all use the same projection function. The primary differences between projection profile algorithms are in the fiducial representation and the alignment criterion used to identify the skew angle.

Because of these similarities, we can adopt a general framework for discussing projection profile algorithms. This framework generalizes to many other skew estimation algorithms, but we restrict our discussion to projection based techniques. Before discussing specific algorithms, we give an informal description of the projection profile method, and develop the generalized framework. It should be noted that this framework does not encompass all implementation details (e.g., bin size, angular resolution, etc.), but rather captures the unique features of an individual algorithm. The specific details regarding parameter settings and other implementation issues can be found in the original works.

3.1 The Basic Projection Profile Algorithm

The basic procedure for projection profile based skew estimation consists of three stages:

1. Reduction

The source image is reduced to a set of representative (fiducial) points. This process is typically performed in order to isolate points in the image that correspond to the baselines of text on the page. This procedure also reduces the amount of data that must be projected in future stages.

2. Projection

The fiducial points from Step 1 are now projected along parallel lines into an accumulator array. The accumulator array is typically partitioned into fixed height bins. The *BINSIZE* is selected so as to maximize the chance of fiducial points belonging to the same text line being projected into the same bin. The angle of projection is varied within an angular search interval, and a projection is done for each desired angle. This creates a sequence of accumulator arrays corresponding to the search angles.

3. Optimization

Once the projection is performed the accumulator arrays are searched for the array which maximizes some alignment premium. This optimization function is a measure of the alignment of the fiducial points at a given angle of projection. The angle corresponding to the accumulator array that maximizes the alignment premium is then given as the skew estimate for the page.

Most projection profile algorithms do not search an entire interval of angles in the iterative process given above. Rather, a coarse sweep of the search interval is done, with progressive refinements of the search done so as to minimize the total number of projections (the most costly step in the procedure). However, the simple description above suffices for the purposes of determining the accuracy of such algorithms. Any progressive or hierarchical search strategy will have a minimum angular search resolution, and if the entire search interval is searched at that resolution by the above method, the results will be identical.

3.2 A Framework for Projection Profile Based Algorithms

It should be noted from the discussion in Section 3.1 that the only substantive differences between projection profile algorithms are the fiducial representation, and the alignment premium. Thus, we can discuss projection profile based skew estimation in terms of these two parameters.

A *projection profile estimator* (PPE) is a pair, (F, ϕ) , where F is a fiducial reduction of a source image to some subset of points considered representative, and ϕ is an alignment premium which takes an accumulator array and returns a measure of alignment for the projection at that angle. This function is then optimized over the range of search angles specified by the interval $[\theta_{min}, \theta_{max}]$. F will reduce an image to a set of triples, (x, y, w) , which represent the position and weight of a fiducial point.

Given a PPE, (F, ϕ) , a skew angle estimate is obtained by the following algorithm:

```

PPE-ESTIMATE( $I, w, h, (F, \phi)$ )
 $I$  = source image,  $w$  = image width,  $h$  = image height,  $(F, \phi)$  = a PPE
for  $(x, y, w) \in F(I)$ 
  for  $\theta \in [\theta_{min}, \theta_{max}]$ 
     $\rho = y + x * \tan \theta$ 
     $A[\theta, \rho / BINSIZE] + = w$ 
  endfor
endfor
PPE-Est =  $\theta$  such that  $\phi(\theta) = \max_{\theta} \phi(A[\theta])$ 

```

This framework easily generalizes to other skew estimation techniques as well. For example, if the projection method is made a parameter to the above algorithm the Hough transformation based approaches can be discussed in the same way. Accordingly, any measure of performance for this type of skew estimation algorithm is a measure of two basic features. Specifically, when we ask how well a projection profile algorithm works, we are asking the two following questions:

1. How well do the fiducial points represent text lines?
2. How well does the alignment premium measure the alignment of text lines?

This framework gives us a good starting point for evaluating performance of these algorithms. Several projection profile algorithms appearing in the literature are described below. Each is recast according to the above general scheme for ease of comparison.

3.3 The Method of Baird

Baird [1] gives a PPE using connected components for fiducial representation and an “energy alignment measure” for the alignment premium.

Specifically, the PPE given by Baird is (F_B, ϕ_B) where

$F_B(I) = \{(x, y, 1) \mid (x, y) \text{ is the center of lower bounding segment of a blob in } I\}$,

and

$$\phi_B(\theta) = \sum_{\rho=0}^{\text{height}} A[\theta, \rho]^2.$$

Note particularly that each connected component is weighted equally regardless of size. This can be beneficial if there are large non-text structures present on the page. However, if the print on the page is especially heavy and there are many touching characters, there will be fewer samples from which to estimate the text line alignment. On the other hand, if the print is light and there are many broken characters, the correlation between the fiducial points and the actual text lines will suffer.

Baird mentions the need to filter the connected components before projection, but no selection criteria is given. This filtering is critical, since small blobs will most likely represent speckle or other noise. Similarly, large blobs typically represent line or half-tone graphic regions.

3.4 The Method of Nakano, et. al.

Nakano [14] gives a skew estimator similar to that of Baird. In this paper four PPEs are described that differ only in the alignment function ϕ .

The PPE, (F_N, ϕ_N) , is described as follows:

$$F_N(I) = \{(x, y, w) \mid (x, y) \text{ is lower left corner of blob of width } w\}$$

and ϕ_N is one of

1. Sum of Absolute Differences:

$$\phi_{N_1}(\theta) = \sum_{\rho=0}^{\text{height}} |A[\theta, \rho] - A[\theta, \rho - 1]|$$

2. Maximum Voting:

$$\phi_{N_2}(\theta) = \max_{\rho} (A[\theta, \rho + 1] - A[\theta, \rho])$$

3. Sum of Averaged Inclination:

$$\phi_{N_3}(\theta) = \sum_j \sum_{\rho=y_1(j)}^{y_2(j)} \frac{|A[\theta, \rho + 1] - A[\theta, \rho]|}{y_1(j) - y_2(j)},$$

where j ranges over the non-zero runs in the accumulator array $A[\theta]$, and $y_1(j)$ and $y_2(j)$ represent the upper and lower bounds of non-zero run j .

4. Number of Zeros:

$$\phi_{N_4}(\theta) = \sum_{\rho=0}^{\text{Height}} [1 - u(A[\theta, \rho])],$$

where

$$u(x) = \begin{cases} 1 & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases}$$

The only difference in the fiducial reduction between this method and Baird's is that Nakano chooses to weight each blob by its width, and to position the fiducial point at the lower left corner of the blob rather than in the center of the lower bounding segment. There seems to be no reason to prefer one positioning strategy over the other, but there is strong evidence against weighting blobs by width rather than equally. While it is true that smaller blobs (perhaps representing noise) will have a smaller effect, larger zones which almost certainly do not represent text will create sharp peaks in the projection profile. On a page containing graphic structures this effect could be particularly damaging.

Nakano claims that ϕ_3 and ϕ_4 perform best, and this claim is indeed supported by the experimental results given in Chapters 5 and 6. ϕ_4 , in fact, performed surprisingly well considering its simplicity.

3.5 The Method of Postl

Postl [17] gives a simple and elegant presentation of a PPE based on fixed subsampling. The PPE, (F_P, ϕ_P) , is given by:

$$F_P(I) = \{(a \times \Delta\xi, b \times \Delta\eta, 1) \mid 0 < a < w/\Delta\xi, 0 < b < h/\Delta\eta, I[a \times \Delta\xi, b \times \Delta\eta] = 1\}$$

and

$$\phi_P(\theta) = \sum_{\rho=0}^{Height-1} (A[\theta, \rho+1] - A[\theta, \rho])^2$$

where $\Delta\xi$ and $\Delta\eta$ are the horizontal and vertical sampling frequencies respectively.

The basic idea here is to subsample the image according to fixed horizontal and vertical sampling frequencies. This simple strategy should reduce the effect of noise, since textual and other periodic structures will be sampled more regularly and therefore be amplified in the final projection array. The alignment function ϕ_P will then detect the angle producing the most low to high transitions in the accumulator array. Since text blocks tend to produce this characteristic pattern in the accumulator, the angle detected should be close to the desired skew angle.

The choice of $\Delta\xi$ and $\Delta\eta$ is highly dependent on the assumed text size on the page. $\Delta\eta$, for example, must be selected to maximize the number of raster lines sampling the actual text lines. Of course $\Delta\xi = \Delta\eta = 1$ may be selected as sampling frequencies, but this is computationally expensive, and any filtration of noise elements is lost.

The author claims error of <0.01 radians for this technique. However, no explanation of how this result was obtained is provided. In order to judge the merit of such a claim, one would have to know the specific details of the experiments conducted.

3.6 The Method of Bloomberg, et. al.

Bloomberg [2] describes a PPE similar to those of Baird and Postl. A subsampling strategy is used, but instead of the naive fixed sampling strategy of Postl, Bloomberg reduces square clusters of pixels to one composite pixel. All pixels in the reduced image are then used as fiducial points in the projection analysis. This reduction

creates a large computational savings. A 2x reduction in image size constitutes a 4x reduction in processing during the projection stage.

The fiducial reduction given by Bloomberg allows for great flexibility. The author provides details for two such resolution reduction techniques. Both reduce a square array of pixels from the source image to a single pixel in the final reduced image. The first uses a 2x2 array of pixels, and the second a 2^n square array, where n is arbitrarily chosen. For convenience, let the square array of pixels corresponding to pixel (x, y) in the reduced image be $S_{(x,y)}$.

The two resolution reduction schemes detailed in the paper are:

$$F_{Bl_1}(I) = \{(x, y, 1) \mid \text{any pixel in } S_{(x,y)} \text{ are on}\}$$

and

$$F_{Bl_2}(I) = \{(x, y, 1) \mid \text{a pixel in some arbitrary row of } S_{(x,y)} \text{ is on}\}.$$

This fiducial reduction can be generalized to support arbitrary conditions on the source array $S_{(x,y)}$. This allows for great flexibility in retaining important typographical features, while improving the computational efficiency of the projection.

Two alignment functions are given as

$$\phi_{Bl_1}(\theta) = \sum_{\rho=0}^{height} A[\theta, \rho]^2.$$

and

$$\phi_{Bl_2}(\theta) = \sum_{\rho=0}^{Height-1} (A[\theta, \rho+1] - A[\theta, \rho])^2.$$

Note that ϕ_{Bl_1} is equivalent to ϕ_B and ϕ_{Bl_2} is equivalent to ϕ_P given above. So that the real contribution of this technique is the fiducial reduction strategy.

Bloomberg also gives several new techniques for optimizing the alignment function, as well as a method for associating a confidence interval for each estimate. This is an important step in the evaluation of skew estimators, as no previous author has made an attempt to qualify the output for each estimate.

Bloomberg lists the following as features in the projection profile of ϕ_{Bl_2} which may indicate faulty results:

- Large signal to noise ratio. *Noise* is defined as the average background from the profile, but not including the peak and its two neighbors. The *signal* is then the difference between the peak and the average background.
- Large fluctuations in background about the average. Normalize the variance and eliminate the peak and two neighbors.
- Large values of ϕ_{Bl_2} far from the peak. Calculate a power of ϕ_{Bl_2} , weighted by the absolute value of the angle from the peak.

Bloomberg suggests that these measures be normalized to ensure invariance to type size and black pixel density. Then the measures can be combined using weighting factors and subtracted from a “perfect” confidence level.

Bloomberg provides a wide assortment of accuracy measures and analysis, including RMS error and absolute angular error, for several levels of resolution reduction. The algorithm was tested on synthesized images, as well as scanned pages from the University of Washington English Document Image Database [6]. The main result of their experiments was that there was no significant decrease in accuracy between 2x, 4x, and 8x reduction of image size. At all levels of reduction approximately 95% of all pages yielded a skew estimate within 0.5 degrees of the true skew angle.

3.7 The Method of Spitz

Spitz [21] describes a PPE that estimates the skew using a CCITT Group 4 compressed source image. This approach yields significant computational savings by operating on a compressed image rather than the entire uncompressed page. The importance of detecting skew in CCITT G4 compressed images should not be appreciated for the reduction in processing time alone. CCITT G4 compression is the current standard for Facsimile transmission, and a primary application of document conversion systems is for incoming fax transmission.

CCITT G4 uses several coding modes. One of these modes, “white pass mode”, can be used as the basis for fiducial reduction. We briefly describe the CCITT G4 compression algorithm, and then present the PPE given by Spitz.

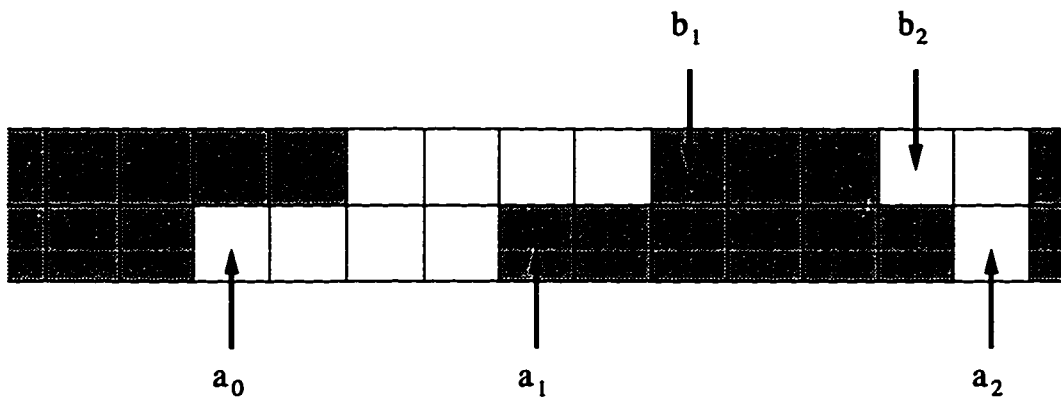


Figure 3.1: Variables used to determine CCITT G4 coding mode

3.7.1 CCITT G4 Compression

The CCITT Group 4 compression algorithm is a two dimensional coding scheme that encodes the current *coding* line with respect to the previous, or *reference* line. There are three coding modes used by the algorithm. The algorithm determines which mode to use to encode a block of pixels by scanning for transitions in the reference and coding lines.

Figure 3.1 details the variables used in determining the correct coding mode to use. The reference and coding lines are scanned for color transitions as follows. The following variables are used for coding the current block:

- a_0 : The starting element on the coding line. If this is the first code for a line, a_0 is defined to be an imaginary white element.
- a_1 : The next transition element to the right of a_0 on the coding line.
- a_2 : The next transition element to the right of a_1 on the coding line.
- b_1 : The first transition element on the reference line to the right of a_0 , and of opposite color to a_0 .
- b_2 : The next transition element to the right of b_1 on the reference line.

There are three coding modes used for the encoding of the current line. The determination of mode is made as follows:

- If a_1 lies *strictly* to the right of b_2 , then *pass mode* is used.

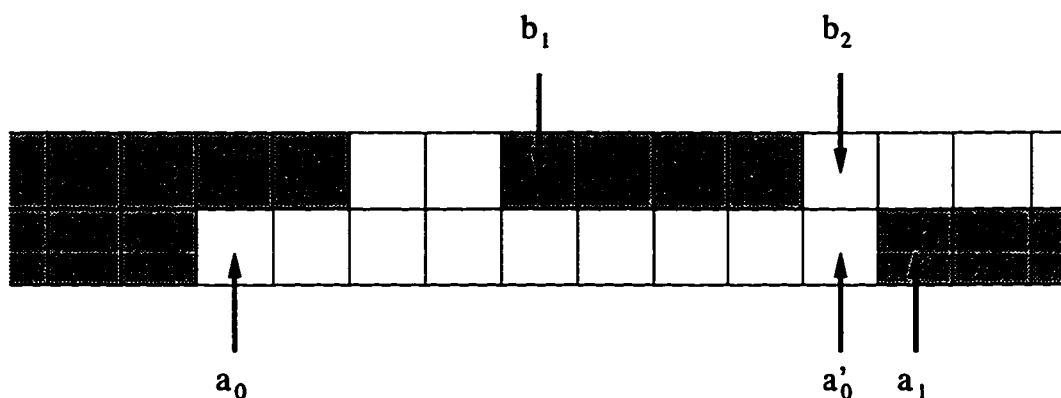


Figure 3.2: A white pass mode coding situation

- If $|a_1 - b_1| \leq 3$, then *vertical mode* is used.
- If $|a_1 - b_1| > 3$, then *horizontal mode* is used.

Once the mode determination has been made, it is encoded using a modified Huffman (MH) encoding along with any additional information needed to reconstruct the coding line block.

For our purposes we are only interested in pass mode coding. Specifically, however, we are interested in *white pass codes*. The details for the CCITT G4 encoding scheme can be found in [4]. An example of a white pass code situation is shown in Figure 3.2.

3.7.2 White Pass Code Detection

The estimation procedure described by Spitz [21] uses white pass codes occurring in CCITT G4 compressed page images as the fiducial points in the projection. He uses the same alignment function as Baird, and concentrates on analyzing the performance of the pass codes fiducial reduction. The MH encoding makes no distinction between white and black pass codes, and some state information must be saved to detect white pass codes specifically. By remembering if white or black pixels are being output, a white pass code can be distinguished immediately from a black pass.

The PPE, (F_S, ϕ_S) , is described as:

$$F_P(I) = \{(x, y, 1) \mid (x, y) \text{ is a white pass code in CCITT G4 source image}\}$$

and

$$\phi_S(A[\theta]) = \sum_{\rho=0}^{\text{height}} A[\theta, \rho]^2.$$

Note that a fiducial point $(x, y, 1) \in F_S(I)$ is actually $(a_0, \text{row}, 1)$, where a_0 is the variable computed as in Figure 3.2 for a white pass code, and row is the current coding line at the time the white pass code is detected.

Spitz additionally provides a comparison of speed and accuracy between his method and Baird's algorithm. The algorithms were tested on a sample of 11 images, and the estimates of both algorithms were nearly identical. The computational savings of performing the fiducial reduction on a compressed image seems to be the critical issue in evaluating the performance of this method. Spitz even suggests that the cost of compressing the source image into CCITT G4 format is outweighed by the efficiency of detecting the white pass codes as opposed to performing the connected component analysis required by Baird's algorithm. One drawback of using white pass codes as the fiducial reduction is that non-character objects cannot be easily filtered. Connected components can be eliminated on the basis of size, while white pass codes have no similar notion.

CHAPTER 4

ESTIMATION OF SKEW IN JBIG IMAGES

Since a major application of document conversion technology is the conversion of incoming fax images, the algorithm given by Spitz is of great practical importance. However, newer compression techniques are slowly being integrated by facsimile manufacturers. The JBIG compression standard will most likely become the next facsimile compression standard. Additionally, online digital libraries need to support rapid transmission of digitized images. Since JBIG is a progressive encoding scheme, multiple levels of resolution can be encoded in a single image. Thus only the bandwidth needed for the local device display resolution need be consumed.

In this Chapter we present a new method for detecting skew in JBIG compressed images. In the next section, the JBIG compression scheme is described. After that, a method for detecting “white pass codes” in JBIG images is presented. It should be noted that these white pass codes are not an artifact of the JBIG compression algorithm as they are for CCITT G4 compression, but rather are artificially detected as a side effect of the JBIG decoding process. It is somewhat of a misnomer to refer to these as white pass codes, but we will do so anyway. White pass codes were chosen because they have been shown by Spitz to be a good fiducial representation of a source image, and are relatively easy to detect from local structures. Lastly, a PPE for detecting the skew angle of JBIG compressed images is described.

4.1 Arithmetic Coding Basics

Since arithmetic coding forms the basis of the JBIG compression algorithm, it is useful to understand some of the rudimentary aspects of this coding technique.

The arithmetic coding process works through a recursive interval subdivision procedure. An binary input string is mapped to a real x in the interval $[0, 1)$. This value x is transmitted instead of the original binary string. Figure 4.1 gives an example of such a procedure.

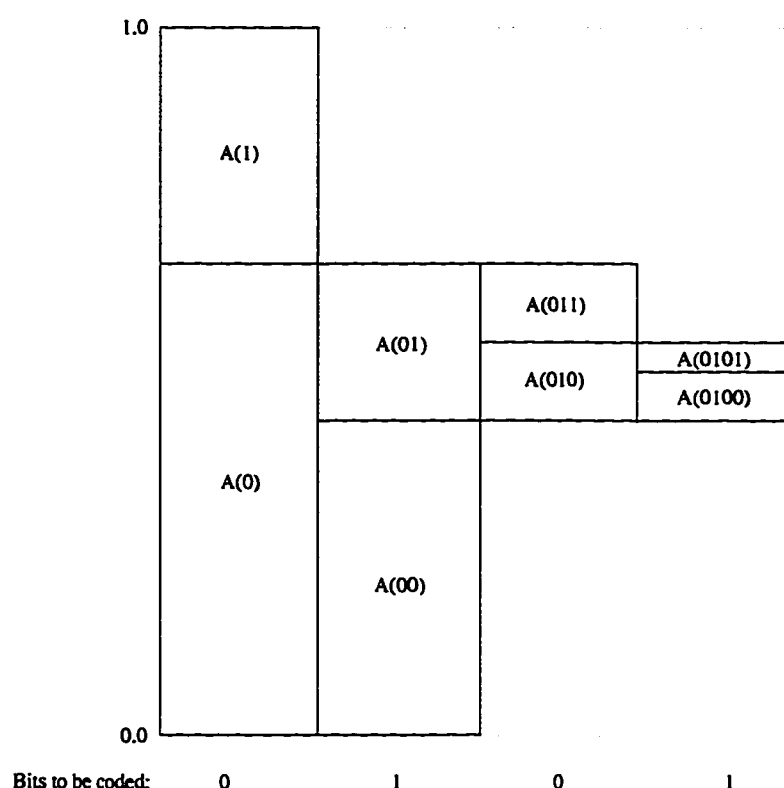


Figure 4.1: Example of Interval Subdivision for Arithmetic Encoding

The interval in which x is known to lie after encountering a string of symbols to be coded is known as the *current coding interval*. This interval is subdivided into two sub-intervals, each one representing the two possible symbols to be coded next. The size of each sub-interval is chosen to reflect the relative probabilities of encountering the corresponding symbol. For example, in Figure 4.1 the interval $A(01)$ is smaller than the interval $A(00)$, indicating that in this interval a 0 is the more probable symbol (MPS), and a 1 is the less probable symbol (LPS).

When an arithmetic decoder receives a code string, x , it must then recursively reconstruct the interval subdivision procedure performed by the encoder. Once this is done, the original binary sequence can be reconstructed.

Since the arithmetic coding process must use integer approximations to the real code symbol x , there are limitations to the amount of information that may be coded by a given code symbol.

4.2 The JBIG Algorithm

The JBIG compression algorithm is a progressive encoding scheme. By this we mean that multiple resolution layers are encoded into a single compressed image. This has the advantage of improving the overall compression ratio, as well as allowing for display on devices of varying resolution capabilities.

The JBIG algorithm employs different techniques for encoding the lowest level resolution layer and the subsequent higher, or *differential*, resolution layers. The encoding of the lowest resolution layer is much simpler than the differential encoding of higher resolution layers, and is an ideal starting point for the development of a fiducial point extraction technique. This discussion is therefore restricted to the details of the lowest resolution layer encoding scheme. It is important to note that for our experiments *no* resolution reduction was performed. The images are encoded using only one resolution layer (i.e., the original image resolution). Those interested in the details of differential coding in the JBIG compression scheme should consult the draft JBIG standard [11].

The first step in encoding an image is to subdivide it into fixed height horizontal stripes. Each stripe will then be coded independently. The raster lines in a stripe can be coded in two distinct ways. A “psuedo-pixel” is coded at the beginning of the raster line to indicate which coding mode to use.

If the current coding line is identical to the previous line, the psuedo-pixel will indicate this. This coding mode, lowest-resolution-layer typical prediction (TP), requires no additional information about the pixels of the current coding line. The previous line can simply be copied into the current.

If the psuedo pixel does not indicate that the current line is “typical”, the line

must be arithmetically coded. The JBIG algorithm uses 1024 parallel arithmetic encoders/decoders to maximize the compression efficiency. The algorithm determines which ecoder/decoder to use on the basis of previously coded pixels. For the lowest resolution layer, two different spatial coding *contexts* are used. These are shown in Figures 4.2 and 4.3.

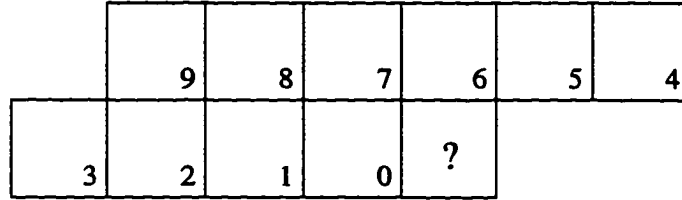


Figure 4.2: Two-line Template for Determining Coding Context

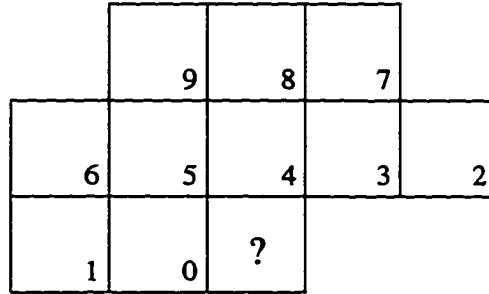


Figure 4.3: Three-line Template for Determining Coding Context

In Figures 4.2 and 4.3, the pixel labeled “?” indicates the pixel currently being coded. The other pixels that are spatially located as in the above figures are combined to create the coding context. If p_i , where $0 \leq i \leq 9$, denotes the pixel value of a pixel in a coding context as above, we can compute an integer representing the coding context of the current pixel as:

$$Cx = \sum_{i=0}^9 2^i p_i.$$

Thus any coding context can be represented as a 10 bit binary integer. This context Cx is used to index the array of arithmetic encoders/decoders. This scheme is used so the relative probabilities for the MPS and LPS can be independently tuned for each context.

4.3 Detecting White Pass Codes in JBIG Images

The arithmetic coding used for JBIG compression obfuscates the underlying spatial features of the image. Consequently, it is difficult to detect such features from the raw stream of arithmetic codes. However, it is possible to use the contexts (which *must* be constructed by the decoder) as the basis for such feature detection.

Recall that a white pass code is generated by the CCITT G4 compression algorithm anytime a white run is encountered on the current coding line that “passes” a black run on the reference line (Figure 3.2). This situation can be detected in the process of decoding a JBIG image by analyzing the sequence of coding contexts used to decode incoming pixels. We define an automaton of two states (plus an output state) that detects these pass code situations in images using the two-line context (Figure 4.2).

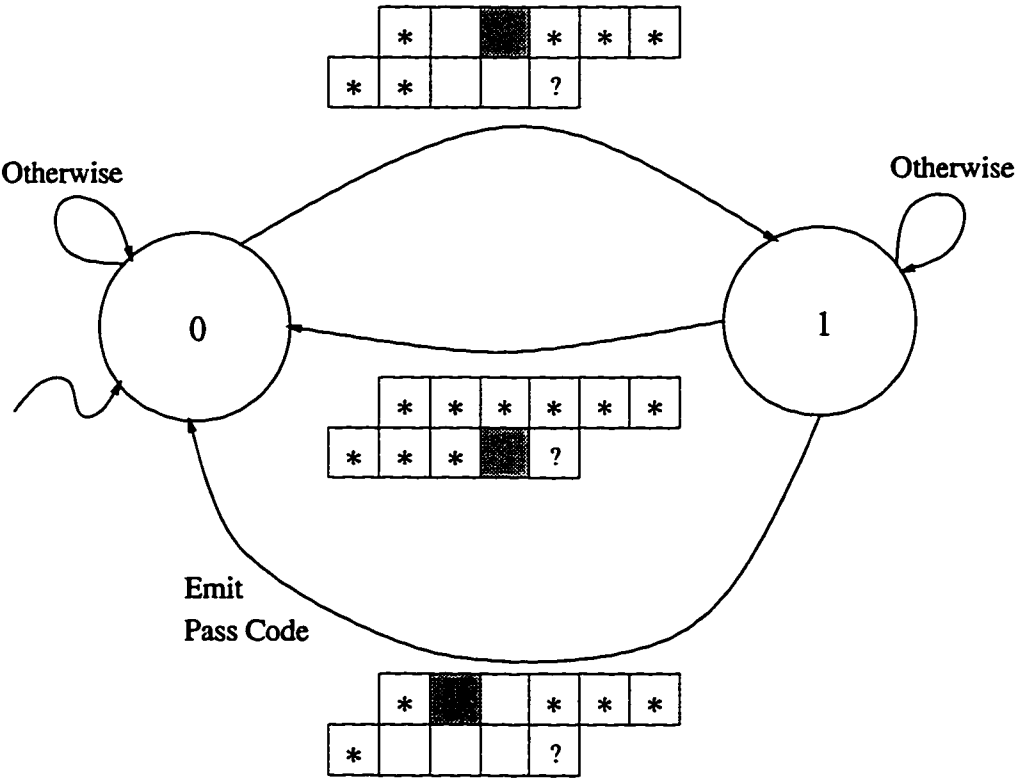


Figure 4.4: Automaton to Detect White Passes from JBIG Contexts

This automaton is pictured in Figure 4.4. In the diagram, a shaded box indicates that the corresponding pixel of the context must be on for that transition. An empty

box indicates that the corresponding pixel must be off. A “*” in a context position indicates that either pixel value is allowed. The “?” again indicates the pixel currently being coded.

The operation of the above machine can be described as follows:

- In State 0, no potential pass code has been encountered. The automaton remains in this state until a sequence of two white pixels with a black pixel above is seen. In this case there is a potential white pass, and the machine enters state 1.
- In State 1, if a black pixel is encountered on the coding line, there is no pass code and the automaton returns to state 0.
- If in State 1 the machine encounters a terminal black pixel (i.e. followed by a white pixel) on the reference line, and there is a run of *at least* three pixels on the coding line, a pass code is detected. The machine emits a pass and returns to state 0. When the machine emits a pass code, it also emits the x and y position at the point of detection.

The fact that a context can be represented by a single 10 bit integer makes the implementation of the above automaton particularly simple. Since there are only two states (plus an output indicator), only 3 bits of state information has to be retained to execute the machine. Thus the entire transition table for the automaton can be easily coded with a 512 byte table that is then indexed by the context Cx . Thus the problem of detecting white passes in JBIG compressed images is solved as a side effect of the decoding procedure itself. A similar automaton can be devised to detect white pass codes in images using the three-line context to encode the lowest resolution layer. Appendix A includes the transition table for our automaton, as well as the source code implementing it.

4.4 A Skew Estimator for JBIG Compressed Images

We implemented the above algorithm for detecting white passes in JBIG images. The algorithm detects white passes in images with no resolution reduction (i.e. no differ-

ential layers). This forms the basis for the fiducial reduction for our skew estimator. A free implementation of the JBIG compression/decompression algorithms was used to facilitate our implementation of the white pass code detector [13].

The PPE, (F_J, ϕ_J) , is described as follows:

$$F_J(I) = \{(x, y, 1) | \text{a white pass code is detected at } (x, y)\}$$

and

$$\phi_J(A[\theta]) = \sum_{\rho=0}^{h-1} (A[\theta, \rho+1] - A[\theta, \rho])^2$$

Note that we choose $\phi_J = \phi_P$. Preliminary results indicated that this alignment function performed more accurately than the alignment function ϕ_S used by Spitz.

CHAPTER 5

ZONE BASED EVALUATION

5.1 Procedures and Methodology

A number of experiments were performed on real world scanned page images. The performance of four PPEs was characterized on a large set of sample images. Table 5.1 shows the algorithms tested along with the parameter settings specific to each algorithm.

Algorithm	Parameters
Baird	BINSIZE=8 Bounding boxes located at midpoint of bottom No pre-processing of bounding boxes
Postl	$\Delta\eta = 8$ $\Delta\xi = 16$
Nakano Measure #4	BINSIZE=8 Bounding boxes located at upper left Measure: Number of Zeros No pre-processing of bounding boxes
JBIG	BINSIZE=8 No pre-processing of white pass codes

Table 5.1: The four algorithms evaluated and parameters for each

We chose to evaluate the performance of these algorithms on real world images rather than synthesized images. After considering which typographical features might affect skew estimation accuracy, we realized that an experiment that completely explored the entire space of possible features would be infeasible. Rather, we decided to use a random sample of real world images that spanned a range of typographical

features. It was hoped that such experiments would indicate which features significantly effect the accuracy of skew estimation. Appendix B gives the initial survey of algorithms and our hypotheses about which typographical features will affect each technique. This analysis can be used for designing experiments using synthesized data.

5.1.1 Test Data

The set of test images used is the “DOE sample” as described in the ISRI third annual test [19]. It consists of 460 pages selected at random from a collection of approximately 100,000 pages. This collection contains about 2,500 technical documents [15].

Each page is scanned at 300 dots per inch using a Fujitsu M3096E+ scanner, using the scanner’s default threshold for conversion to a binary TIFF image. The text portions of all pages are then manually zoned and classified into one of the following categories: *Text*, *Caption*, *Footnote*, *Other_Text*, *Table* [18]. The DOE sample consists of 1,313 text zones, of which 1,246 have determinable skew angles and were used for this study. Some zones were excluded because no skew angle could be measured (for example, zones containing only one printed character such as a page number).

This sample contains a wide variety of typographical features which are of interest in the analysis of skew estimation algorithms. Some of the interesting features include broken characters, touching characters, small zones, tables, line drawings, and half-toned pictures. Since we are primarily concerned with estimating the skew angle of text, experiments are restricted to text zones in the sample. All zones considered contain a single column of text. In this way the effects of non-text features are reduced.

After scanning, the skew angle of each zone (θ_{True}) was manually measured by calculating the relative vertical offset of the right side of the zone from the left (Δy). The width of the zone (Δx) is then used to compute the skew angle as $\arctan(\Delta y/\Delta x)$. Each skew angle is measured at three locations in the zone to ensure accuracy.

An interesting characteristic of the skew angles observed in this sample is shown in Figure 5.1. The frequency histogram shows a symmetric distribution of observations peaked close to zero (Figure 5.1a). Figure 5.1b shows that the observed frequencies

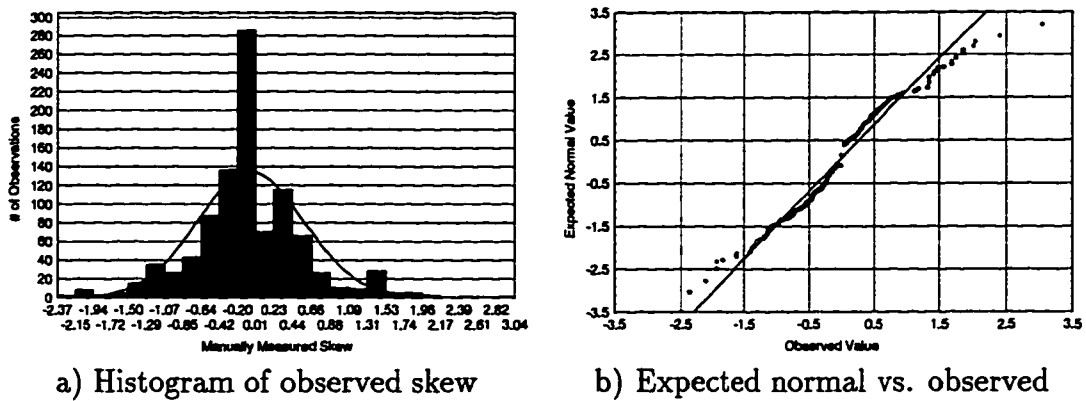


Figure 5.1: The distribution of skew angles in test sample

vary little from the expected value of the normal distribution with $\mu = -0.05$ and $\sigma = 0.6$.

5.1.2 Experiment Methodology

Since most typographical features are easier to classify by zone, we chose to first evaluate the accuracy of the algorithms on a zone-by-zone basis. This allows us to isolate the typographical features that are problematic for skew estimation.

A skew estimate is obtained from each algorithm for each zone in the sample for which a manual skew angle was obtained. The absolute difference of the estimated skew and the ground truth skew angle is used as the accuracy measure for each estimate.

5.2 Analysis

Four methods of analysis were used to explore the limitations of these algorithms. The experiments are intended to examine the limiting factors of each technique, rather than average case performance.

5.2.1 Error Distribution

First, the distribution of absolute error for each algorithm was examined. Our goal was to establish meaningful statistics that are representative of the overall accuracy of the skew estimate. Additionally, all such assertions of overall accuracy could be

Algorithm	Median of Absolute Error
Baird	0.112672
Postl	0.109070
Nakano4	0.148556
JBIG	0.092236

Table 5.2: Median of Absolute Error for All Sample Zones

qualified with appropriate confidence intervals.

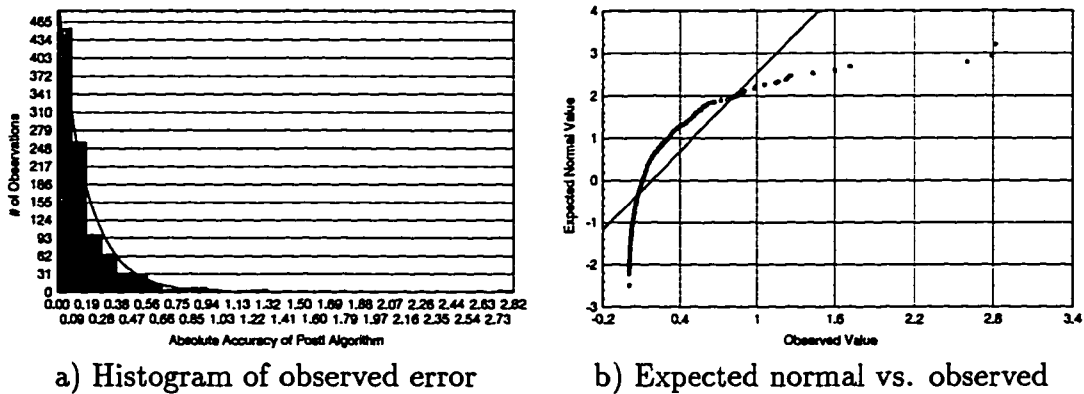


Figure 5.2: The Distribution of Absolute Error for Postl Algorithm

Figure 5.2a shows a typical distribution of observed accuracy (in this case for Postl’s algorithm). The distribution is sharply peaked near zero, and is decidedly non-normal (Figure 5.2b). The observed frequencies are instead characteristic of an exponential distribution. Consequently, classical interval estimation becomes difficult, and it is unclear if the mean accuracy is even a meaningful statistic. On the basis of this analysis, it seems that non-parametric methods for performance analysis will be necessary to establish a measure of overall performance. One statistic based on the absolute error distribution can be useful. Table 5.2 shows the *median* of absolute error. Unlike the mean, the median statistic is less sensitive to outliers in the sample.

5.2.2 Regression Analysis

Intuitively what we desire of a “good” skew estimation algorithm is a linear correlation between the estimated angle and the ground truth skew angle. A concise and informative measure of this can be obtained from a simple scatterplot and linear

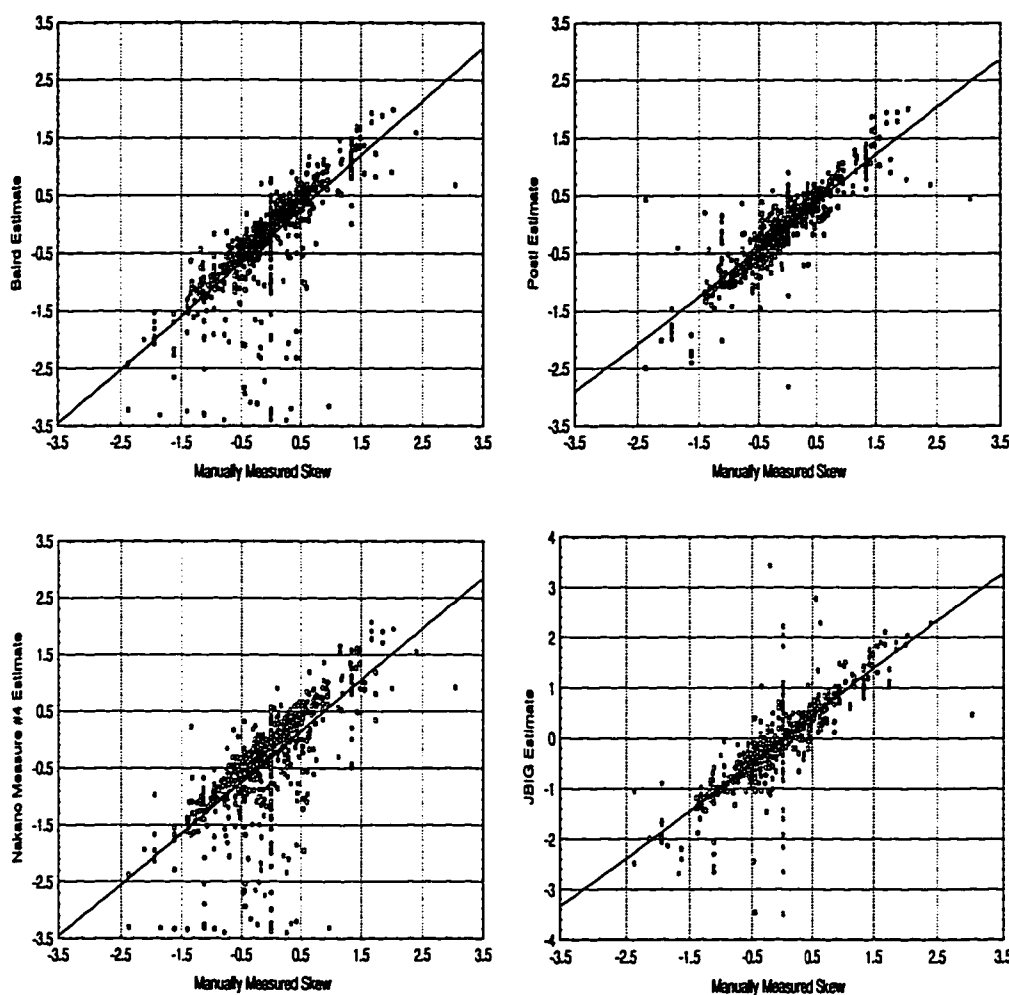


Figure 5.3: Raw scatterplots and regression lines for each algorithm over the entire set of text zones

regression.

Figure 5.3 shows the scatterplots with regression lines for each algorithm on the entire set of sample zones. Table 5.3 summarizes the linear correlation values obtained from the linear regressions.

Additionally, a visual inspection of the scatterplots in Figure 5.3 provides valuable information about outliers in the test samples. By analyzing the residual information from the linear regressions, the pathological zones that were causing problems for every algorithm were isolated.

Algorithm	Correlation ρ	Y-intercept α	Slope β
Baird	0.79553	-0.15906	0.93689
Postl	0.89724	-0.01824	0.82893
Nakano4	0.73662	-0.25989	0.90753
JBIG	0.88918	-0.02940	0.94513

Table 5.3: Summary of linear regression for all sample zones

Algorithm	Correlation ρ	Y-intercept α	Slope β
Baird	0.88189	-0.10083	0.92612
Postl	0.92588	-0.02547	0.86555
Nakano4	0.84354	-0.15201	0.92119
JBIG	0.92588	-0.02547	0.86555

Table 5.4: Summary of linear regression for zones of type *Text*

5.2.3 Sensitivity to Typographical Features

The majority of zones in the set of common outliers contained few textlines. This led us to investigate the number of textlines as a feature affecting skew estimation accuracy.

Intuitively, the number of textlines represents the amount of information available for an algorithm to infer the skew angle. When the linear regressions are computed using only zones of type *Text*, which represent the main body text consisting of many text lines from the sample, the correlation improves dramatically. The regression information for zones of type *Text* are summarized in Table 5.4.

To determine the effect that the number of textlines has on the accuracy of the algorithms, we plot the correlation of each algorithm as a function of the number of textlines in each zone. The observations corresponding to zones with fewer than a certain number of text lines were iteratively eliminated from the linear regression. In this way it can be determined how well an algorithm performs on zones with x or more textlines. Figure 5.2.3 shows the results of this analysis. Note that the methods that use bounding rectangles show a more rapid increase in accuracy than the Postl algorithm which is based on the original image. This is due to the fact that most of the small text zones contain very few connected components which further reduces

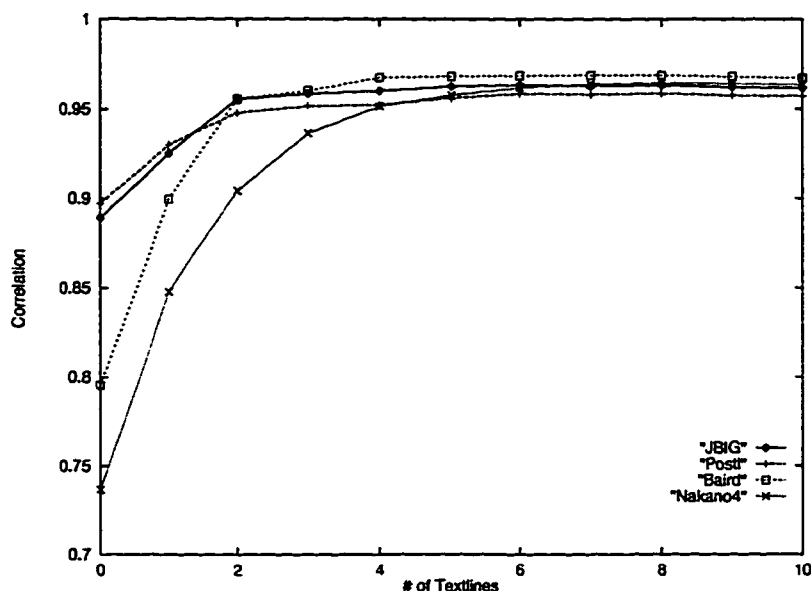


Figure 5.4: Effect of textlines on accuracy. Each point represents the correlation of the algorithm on zones of greater than x textlines.

the amount of information the algorithm has. Most of the algorithms achieve a stable correlation value before ten textlines.

The use of the correlation coefficient in this analysis can be misleading. The correlation indicates how well the data fits the best estimated linear line for that data set. Alternately, we can use the median of absolute error as our measure of accuracy. Figure 5.2.3 shows this same analysis with the median statistic. As with the correlation measure, the median statistic indicates that the connected component based algorithms (Baird and Nakano) are more sensitive to the number of textlines than the others.

We can extend this method of analysis to determine whether or not the zone width has any effect on the accuracy above and beyond the number of textlines. Figure 5.6 shows the result of this analysis. It can be seen that the width of a zone has some effect on the accuracy of the algorithms. Specifically, if a zone has few textlines, the width of a zone can affect the performance of skew estimation. However, the number of textlines is certainly the dominant feature.

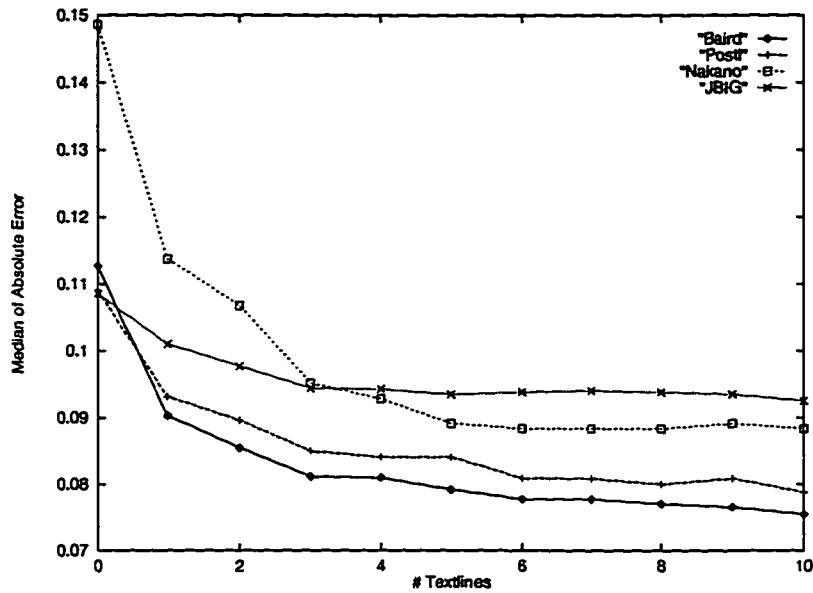


Figure 5.5: Effect of textlines on accuracy. Each point represents the median of absolute error of the algorithm on zones of greater than x textlines.

5.2.4 Non-parametric Measures

Due to the non-normal distribution of the observed error, it is difficult to perform classical interval estimation on the statistics derived from this distribution. Non-parametric methods must be used to infer more about the behavior of these algorithms. A sign test [3] was performed on the absolute error of all the methods. This allows us to determine which algorithms perform significantly better than the others on a pairwise basis. While this method does not yield an independent measure of overall accuracy as desired, it does allow us to make some assertions without making unfounded assumptions about the distribution of observed error. However, the sign test analysis did not indicate a statistically significant difference between any pair. This is unsurprising given that all four algorithms converge to similar values when the number of textlines reaches some minimal number. The outliers affecting the overall average case performance do not influence the sign test results.

It is our hypothesis, however, that the number of textlines will affect the performance of these algorithms, and that this will be reflected in the results of a sign test analysis on such a sample. These results and hypotheses indicate the need for a *stratified sampling* paradigm [3] that would allow for the characterization of performance

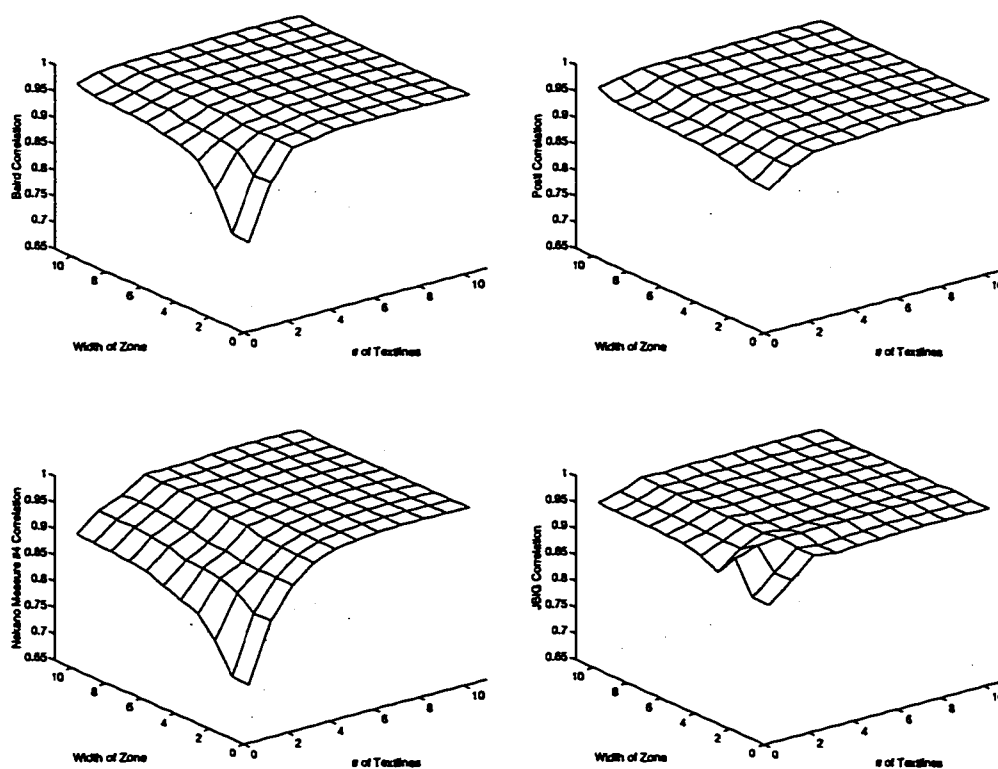


Figure 5.6: Effect of textlines and zone width on accuracy. Each point (x, y) represents the correlation on zones of greater than x textlines and width greater than y .

for these extreme cases rather than an average case measure.

CHAPTER 6

PAGE BASED EVALUATION

Since both global and local skew angles are required when analyzing a document, it is also important to evaluate the performance of skew estimation algorithms on pages. The problem of estimating page based skew is complicated by graphical objects and other non-text regions. In Chapter 5 we evaluated performance on single column text zones. In this chapter we present the results of experiments on entire pages. The algorithms tested are the same as for the zone based experiments of the previous chapter.

6.1 Test Data

The sample used is the same as for the zone based evaluation. It consists of 460 pages randomly sampled from a collection of about 2,500 technical documents [15]. The skew angles for this sample were manually measured on a zone by zone basis. Several problems were encountered when attempting to derive a single skew angle for entire pages. For example:

- Some pages had zones skewed at different angles. This can arise, for example, when two columns are pasted onto the original and then photocopied, or when each column is typed separately in a mechanical typewriter.
- Some zones possess a non-linear skew. When a book is photocopied this effect occurs at the inside margin of the page(s) being copied.

These problems make it difficult to define the skew angle of the entire page. Consequently, it is difficult to define the expected output for a skew estimator operating

on a page image. Some aggregate measure of skew seems necessary.

6.2 Aggregate Measures of Skew

Two aggregate measures of skew were evaluated separately. Intuitively, we want a skew estimator to return the skew angle corresponding to the angle of the majority of text on the page. The first aggregate measure is defined to be the manually measured skew angle for the largest text zone on the page. We call this the *dominant skew angle*. The second measure is a weighted average of all zone skew angles for the page. If a zone on a given page is denoted as (a, θ) , where a is the area of the zone, and θ is the manual skew estimate for the zone, we define the *weighted average skew angle* for a page I as:

$$\frac{\sum_{(a,\theta) \in I} a \times \theta}{\sum_{(a,\theta) \in I} a}$$

The weighted average is just an average of all manually measured skew angles weighted by the area of the corresponding zone.

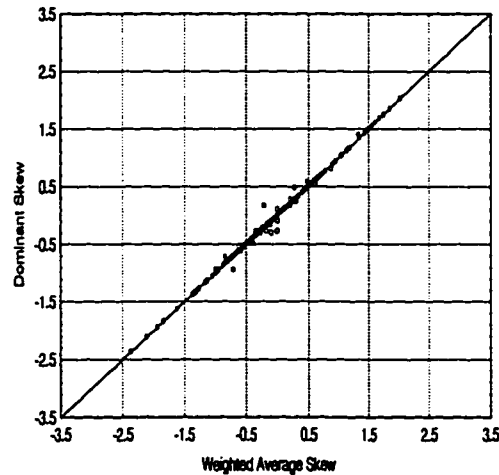


Figure 6.1: Scatterplot of Weighted Average vs. Dominant Skew

For this page sample, which consists primarily of technical documents, there is very little difference between the two aggregate measures. Figure 6.2 shows a scatterplot of the two measures as computed for the 460 page sample. There is a near perfect correlation between the two measures.

6.3 Experiment Methodology

Having computed the aggregate measures for all sample pages, the algorithms were run on each page. The absolute error between a skew estimate and either of the aggregate measures is then used as an accuracy measure for each observation.

6.4 Results and Discussion

Each of the methods of analysis discussed in Chapter 5 was performed for the page based test.

6.4.1 Error Distribution

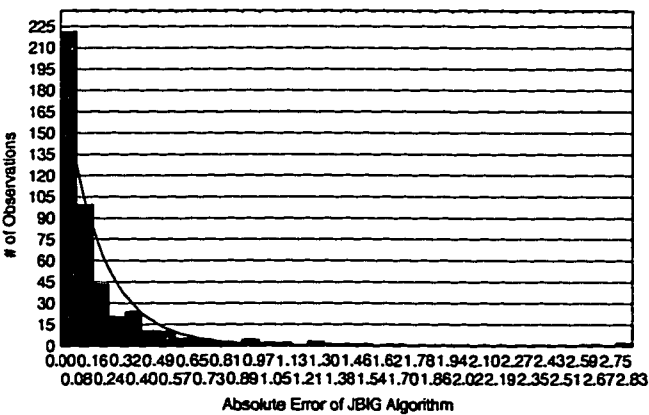


Figure 6.2: The Distribution of Absolute Error for JBIG Algorithm

The distribution of absolute error for the JBIG algorithm in the page sample is shown in Figure 6.2. Once again, the error distribution is non-normal in nature. This is unsurprising, since we would not want the error of such an algorithm to be normally distributed. This complicates the process of analyzing the performance of skew estimators, and other methods must be employed.

6.4.2 Regression Analysis

We performed a linear regression analysis on the page sample, plotting each skew estimate against the aggregate skew measures. The regression information for the

dominant skew angle measure is shown in table 6.1. Table 6.2 gives the regression results for the weighted average skew.

Algorithm	Correlation ρ	Y-intercept α	Slope β
Baird	0.78153	-0.01906	0.77334
Postl	0.85521	-0.01529	0.78050
Nakano4	0.86287	-0.03856	0.85091
JBIG	0.78025	-0.00008	0.74467

Table 6.1: Summary of Linear Regressions for Pages and Dominant Skew

Algorithm	Correlation ρ	Y-intercept α	Slope β
Baird	0.78400	-0.01870	0.77795
Postl	0.85909	-0.01580	0.78614
Nakano4	0.86639	-0.03812	0.85684
JBIG	0.78305	0.00058	0.74934

Table 6.2: Summary of Linear Regressions for Pages and Weighted Average Skew

The surprising result here is that Nakano’s method outperforms all of the others on the basis of the results of the linear regression. This is probably due to the fact that connected components are used for the fiducial reduction, and thus each noise or non-text element will only be sampled once, while the JBIG algorithm could potentially detect many pass codes for a single non-text element. It is not clear why Baird’s algorithm performed so poorly on these pages. Possibly it is due to the fact that each connected component is weighted equally, and thus any speckle will be weighed equally with characters.

Figure 6.3 shows the scatterplots of the four algorithms for the dominant skew angle measure. Figure 6.4 shows the scatterplots for the weighted average measure.

6.4.3 Sensitivity to Typographical Features

In Chapter 5 we established a clear dependence of skew estimation accuracy on the number of textlines present in the zone. To determine if this dependence scales up to the page level, we categorized each page according to the number of textlines contained in all text zones for that page. We then performed a series of linear regressions

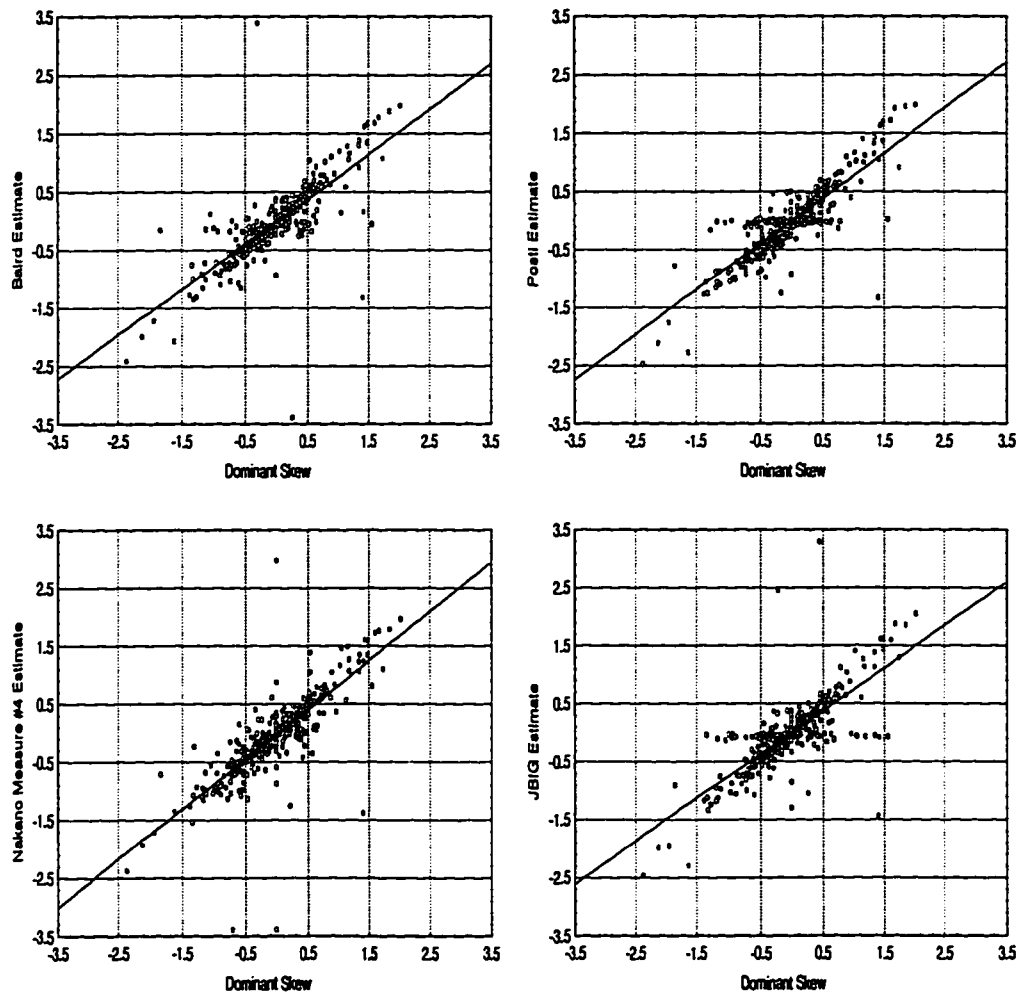


Figure 6.3: Scatterplots of Dominant Skew vs. Estimates

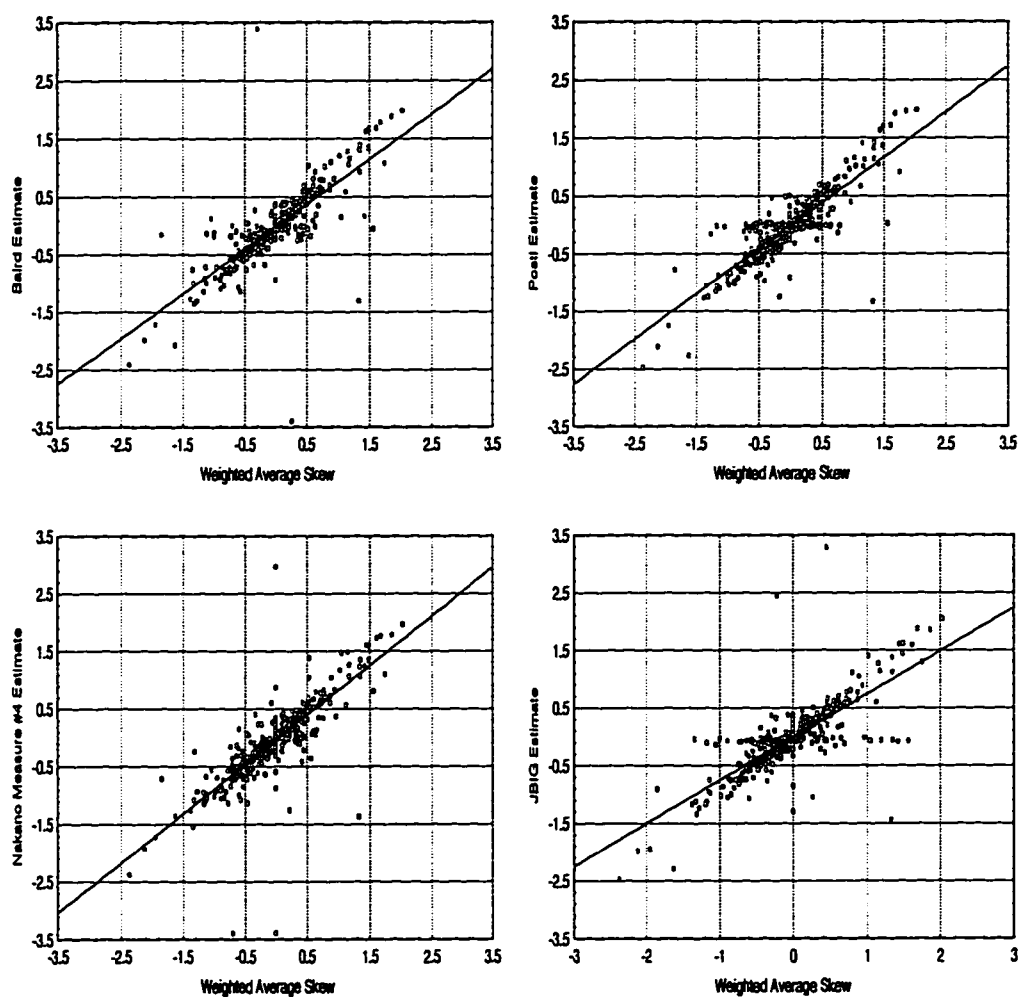


Figure 6.4: Scatterplots of Weighted Average Skew vs. Estimates

by eliminating pages which contained fewer than a fixed number of textlines. The results are shown in Figure 6.4.3.

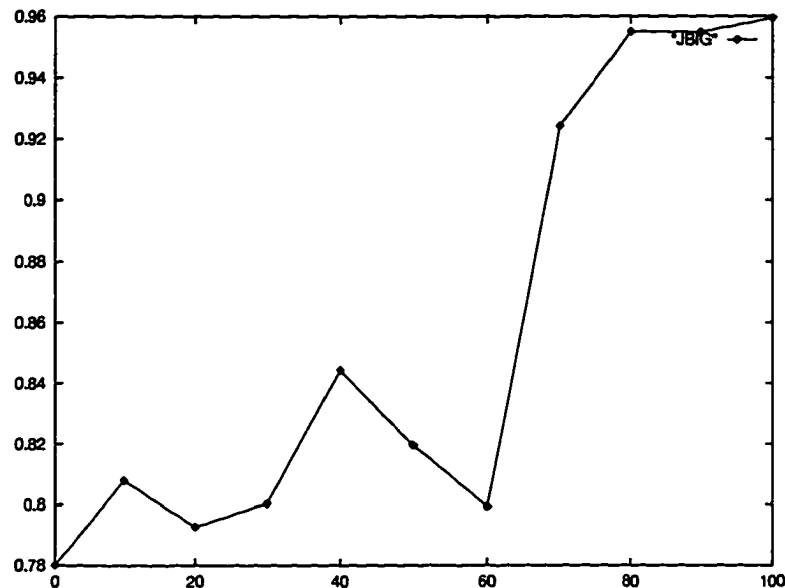


Figure 6.5: Effect of textlines on accuracy. Each point represents the correlation of algorithm on pages of greater than x textlines.

There is no clear relationship between the number of textlines on a page and the skew estimation accuracy. When the JBIG algorithm is tested on the page sample by excluding all pass codes falling outside of text zones, the correlation jumps to 0.89. This indicates that the non-text features of the pages are contributing a significant amount of the error for the JBIG algorithm. Further experiments are required to determine the relationship between typographical features and skew estimation accuracy for pages.

At this time it seems that the aggregate measures of skew for page images are of dubious value. There seems to be no clear reason for preferring them to other measures. Newer skew estimation methods that decompose the image into similarly skewed blocks, and then return a skew estimate for each block, hold the most promise.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Skew Estimation Algorithms

We have investigated the performance of several skew estimation algorithms. Each algorithm has its own unique strengths and weaknesses. Most of the work in this thesis has concentrated on projection profile based skew estimators. These algorithms are particularly easy to analyze since they reduce to two parameters: a fiducial reduction F , and an alignment function ϕ . Other skew estimation techniques (see Chapter 2) are not so easily parametrized. While the projection profile algorithms are very robust over a wide range of typographical features, it is also important to characterize the strengths and weaknesses of other types of estimators.

There are also problems regarding the precise definition of skew. For example, if there are multiple skew angles on a page, it is unclear how to define a single skew angle. It is important to clearly define this concept so that we know what to expect from a skew estimation algorithm. Several new approaches decompose the page into similarly skewed blocks and then perform skew estimation on the individual zones. This approach holds the most promise, since an algorithm providing independent skew angles for all skewed blocks of text on a page is the best solution. An extension of the experiments in this thesis that evaluated the performance of these algorithms would be most useful.

7.2 Evaluation of Skew Estimation Algorithms

Several new approaches to the evaluation of skew estimation algorithms have been investigated here. In summary, we have determined:

- Skew angles in document collections of this type are approximately normally distributed. It is important to know this so that the value of skew estimation can be balanced against the cost of unnecessarily performing it. Of course, it is also important to know the actual effects skew has with respect to the rest of the document conversion process.
- The distribution of absolute error for skew estimation algorithms (at least *good* skew estimators) is decidedly *non-normal*. It is thus difficult to say anything about the error in general. Any average measure of error is particularly sensitive to outliers. Any aggregate measure of average error should weigh each error observation by the number of characters whose skew angle is incorrectly identified.
- Regression analysis provides a clear qualitative measure of skew estimation accuracy for large page samples. A simple visual inspection of the scatter plot gives some indication of performance, the correlation coefficient of the linear regression gives a quantitative measure of accuracy, and the residual information from the regression clearly indicates outliers.
- It is probably impossible to develop a single composite measure of average case performance for skew estimation algorithms. Consequently, it is important to characterize the accuracy of skew estimators over a wide range of typographical features. We have shown how several skew estimators perform poorly on zones with few textlines, and how the width of these sparse zones can further affect the accuracy of skew estimation. Average case performance measures can be misleading due to this variable behavior.
- Non-parametric measures of performance probably hold the most promise for the future. The distribution of error makes statistical inference very difficult. If

confidence intervals for accuracy are to be developed for skew estimators, non parametric methods such as *jackknifing* are probably needed.

Four projection profile skew estimation algorithms were evaluated in the course of this project. Postl's algorithm and the new JBIG technique perform best on the entire set of sample images. However, this can only be inferred from the regression analysis. When the algorithms are limited to zones of four or more text lines they are all competitive. In fact, there is no significant difference in performance beyond a certain point. The poor performance of the Nakano and Baird algorithms on these small zones is due to the fact that they use connected components as fiducial points in their projections. This reduction strategy samples clean characters (i.e. non-touching characters) once only, while other techniques do not suffer from this limitation. White pass codes, for example, outnumber connected components 3 to one. This accounts for the disparity in performance between connected component based techniques and the others.

One particular typographical feature that is of interest is language. The type of fiducial reduction will certainly affect the performance of projection profile algorithms when presented with pages of varying language. Chinese characters, for example, have different moments than Latin characters, and it may be better to position the fiducial points in the center of the bounding boxes rather than on a bounding segment. Arabic (or any cursive language) will prove even more difficult for algorithms using connected components as the basis for fiducial reduction.

7.3 Estimating Skew in Compressed Images

Two methods for detecting skew in compressed images were discussed in this thesis. These techniques have the advantage of being significantly more efficient than algorithms using the entire uncompressed source image. Spitz [21] was the first to propose such a skew estimator for CCITT Group 4 compressed images.

We have developed a new skew estimation procedure for JBIG compressed images. Our technique uses the same fiducial reduction (white pass codes) as Spitz. The new method has been shown to be competitive with the other methods (in terms of accuracy). In fact, the performance of our technique is almost identical to that of

Postl's algorithm.

Since white pass codes are a direct byproduct of CCITT G4 compression, it is unlikely that our method will be competitive with Spitz's in terms of speed. However, JBIG is likely to replace CCITT G4 as the de facto standard for fax transmission, and thus a skew estimator for JBIG images will become more important.

There are several issues regarding skew estimation in JBIG images that still must be investigated. For example:

- There is no reason to use white pass codes for detecting skew in JBIG images. This fiducial reduction was used because it worked for Spitz. It is possible that there are better fiducial representations easily detected from the JBIG contexts. Ideally these fiducial points would be identifiable from the local structures required by the JBIG decoder to decode a pixel. Again, an automaton similar to the one developed to detect white passes could be used.
- The current JBIG skew estimator works only on the lowest level resolution layer (with no resolution reduction). Bloomberg [2] indicates that resolution reduction does not significantly affect skew estimation. An experiment should be performed to determine if the JBIG skew estimator works well at 2x, 4x, and 8x reduction.
- When multiple resolution layers are coded, it may be important to detect skew in some layer other than the lowest. The algorithm should be extended to work on an arbitrary differential layer. It may be possible to modify the DFA used in the lowest layer to work with the modified context of the differential layers. However, the presence of typical and deterministic prediction could complicate the process.
- Lastly, it would be nice to combine lowest level and differential layer skew estimation to produce a progressively refined skew estimate. It may be possible to make a coarse estimate at the lowest level, and then iteratively constrain the angular search interval as new resolution layers are added.

One final note; methods suggested by Bloomberg [2] for associating a confidence interval with each skew estimate should be investigated. In our experiments, some

skew estimates were in error by more than 3 degrees. In cases like this, no skew estimate is certainly better than a bad one.

7.4 Contributions

To briefly summarize our contributions to the subject of document image skew estimation, we have:

1. developed a new skew estimator for JBIG compressed images that is competitive with the other algorithms in the literature,
2. proposed a generalized framework for projection profile skew estimators that reduces the algorithms to two essential parameters,
3. conducted a large scale experiment designed to characterize the performance of several skew estimation algorithms,
4. evaluated the performance of these skew estimators on single column text zones,
5. proposed two aggregate measures of skew for pages and evaluated the accuracy of skew estimation with respect to these measures,
6. determined that a linear regression analysis is useful for quantifying the performance of skew estimators and for identifying outliers, and
7. indicated the dependence of skew estimation accuracy on typographical features such as the number of textlines.

APPENDIX A – PASS CODE AUTOMATON

The transition table for the white pass code detection automaton is given here, along with the routine that implements it. For ease of understanding and implementation, the transition table is not packed into a 512 byte format. Instead, each byte in the table codes a single transition for states one and two. The unused bits in each entry can be used for future expansion.

```
/*
**
** trans_table.c - Transition table for JBIG pass code detector
**
** Low nybble indicates transition for state 0, high nybble for state 1.
** If high bit set, pass code is detected.
**
** 10-13-96 (ADB)
**
*/
unsigned char TransTable[1024] = {
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0, 17,  0, 16,  0,
  128, 0, 16,  0, 16,  0, 16,  0, 128, 0, 16,  0, 16,  0, 16,  0,
  128, 0, 16,  0, 16,  0, 16,  0, 128, 0, 16,  0, 16,  0, 16,  0,
  128, 0, 16,  0, 16,  0, 16,  0, 128, 0, 16,  0, 16,  0, 16,  0,
  128, 0, 16,  0, 16,  0, 16,  0, 128, 0, 16,  0, 16,  0, 16,  0,
```



```
**
** 10-13-96 (ADB)
*/
unsigned int Pass_Code_Automaton(unsigned int cx)
{
    static int state = 0;

    state = (TransTable[cx] >> (state << 2)) & 0x0f;
    if (state & 0x08) {
        state &= 0x01;
        return(TRUE);
    }
    return(FALSE);
}
```

APPENDIX B – INITIAL SURVEY OF ALGORITHMS

When this project was initiated a table was constructed listing all algorithms under consideration. All algorithms parameters were included, as well as many conjectures regarding the effects of many typographical features on the algorithms. The final conclusion drawn from this process was that an experiment testing an algorithm accross the entire range of features expected to impair performance would be infeasible. We include the contents of that initial table here for posterity.

	Baird	Hinds	Nakano	Postl
Parameters	BINSIZE Location of blob Blob filter Weight of blob	BINSIZE Black run filter	BINSIZE Location of blob Blob filter Blob weight Alignment Measure	$\Delta\xi$ $\Delta\eta$
Accuracy Claim	2' of arc	100% correct on sample	0.1 degrees print 0.2 degrees + graphic	0.01 radians
Notes	Fast, performs well	Performs poorly on sample 2		Slow, params, performs well

Table B.1: Algorithms in Question and Associated Parameters

	Baird	Hinds	Nakano	Postl
Italics	No effect	No effect	No effect	No effect
Proportional Pitch	No effect	No effect	No effect	$\Delta\xi$ selection
Fixed Pitch	No effect	No effect	No effect	$\Delta\xi$ selection
Char Spacing	None, unless touching	None, unless touching	None, unless touching	$\Delta\xi$ selection
Typesize	BINSIZE, filter	BINSIZE, filter	BINSIZE, filter	$\Delta\xi$ and $\Delta\eta$
# Textlines	None, unless graphics present	None, unless graphics present	None, unless graphics present	$\Delta\eta$
Textline Width	BINSIZE	BINSIZE	no effect	$\Delta\xi$
Textline Space	\downarrow var in PP	\downarrow var in PP	measure selection	$\Delta\eta$
Column Skew	Weighted avg	Dominant	Measure dependent	Dominant
Baseline Skip	\downarrow var in PP	\downarrow var in PP	Measure dependent	\downarrow var in PP
Line Drawings	Filtered prob with dashed	Filtered, unless wide strokes	Filtered	No effect
Halftone	No effect unless frags	Filtered	Filtered	No effect unless dense
Broken Chars	Big effect weighted equally	No effect	Small effect	No effect
Touching Chars	Some effect, must filter	No effect	Some effect	No effect
Speckle	Must filter	No effect	No effect	No effect
Black Margins	No effect, must filter	No effect	No effect, must filter	No effect
Scanning Resolution	Affect filter, BINSIZE	Filter, BINSIZE	Filter	$\Delta\xi$ and $\Delta\eta$
Baseline Shift	Dominant	Dominant	Dominant	Dominant

Table B.2: Typographical Features Possibly Affecting Skew Estimation

BIBLIOGRAPHY

- [1] Henry S. Baird. The skew angle of printed documents. In *Proc. of SPSE 40th Annual Conference and Symposium on Hybrid Imaging Systems*, Rochester, NY, May 1987.
- [2] Dan S. Bloomberg, Gary E. Kopec, and Lakshmi Dasari. Measuring document image skew and orientation. In *SPIE Proceedings Series: Document Recognition II*, pages 302–316, San Jose, CA, 1995.
- [3] H. D. Brunk. *An Introduction to Mathematical Statistics*. Blaisdell Publishing Company, 1965.
- [4] CCITT Recommendation T.6, “Facsimile Coding Schemes and Control Functions for Group 4 Facsimile Apparatus.” *Terminal Equipment and Protocols for the Telematic Services*, VII, Fascicle VII.3, 1989.
- [5] S. Chen and Robert M. Haralick. Recursive erosion, dilation, opening, and closing transforms. *IEEE Transactions on Image Processing*, 4(3), March 1995.
- [6] S. Chen, M. Y. Jaismha, J. Ha, I. T. Phillips, and R. M. Haralick. Uw english document image database - (i) manual. *Reference Manual*, 1993.
- [7] Su Chen, Robert M. Haralick, and Ihsin T. Phillips. Automatic text skew estimation in document images. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 1153–1156, Montreal, Canada, 1995. IEEE Computer Society Press.
- [8] Robert M. Haralick, S. Chen, and T. Kanungo. Recursive opening transform. In *CVPR*, Champaign, June 1992.
- [9] Masahiko Hase and Yasushi Hoshino. Segmentation method of document images by two-dimensionl fourier trasformation. *Systems and Computers in Japan*, 16(3), 1985.
- [10] Stuart C. Hinds, James L. Fisher, and Donald P. D’Amato. A document skew detection method using run-length encoding and the hough transform. In *Proceedings of the 10th Annual Pattern Recognition Conference*, pages 464–468, Atlantic City, NJ, 1990. IEEE Computer Society Press.

- [11] International Standard ISO/IEC 11544:1993 and ITU-T Recommendation T.82(1993), "Information technology - Coded representation of picture and audio information - progressive bi-level image compression," 1993
- [12] Yasuto Ishitani. Document skew detection based on local region complexity. In *Proceedings of the Second International Conference on Document Analysis and Recognition*, pages 49–52, Los Alamitos, CA, 1993. IEEE Computer Society Press.
- [13] Markus Kuhn, *JBIG-KIT V0.8*, available from <ftp://ftp.informatik.uni-erlangen.de/pub/doc/ISO/JBIG/jbigkit-0.8.tar.gz>.
- [14] Yasuaki Nakano, Yoshihiro Shima, Hiromichi Fujisawa, Jun'ichi Higashino, and Masaaki Fujinawa. An algorithm for the skew normalization of document images. In *Proceedings of the 10th Annual Pattern Recognition Conference*, pages 8–11, Atlantic City, NJ, 1990. IEEE Computer Society Press.
- [15] T. A. Nartker, R. B. Bradford, and B. A. Cerny. A preliminary report on UNLV/GT1: A database for ground-truth testing in document analysis and character recognition. In *Proc. 1st Symp. on Document Analysis and Information Retrieval*, pages 300–315, Las Vegas, NV, March 1992.
- [16] L. O'Gorman and Rangachar Kasturi. Text analysis and recognition. *Document Image Analysis*, 1995.
- [17] W. Postl. Detection of linear oblique structures and skew scan in digitized documents. In *Proceedings of 8th International Conference on Pattern Recognition*, pages 687–689, Paris, France, 1986.
- [18] Stephen V. Rice, Junichi Kanai, and Thomas A. Nartker. Preparing OCR test data. Technical Report 93-08, Information Science Research Institute, University of Nevada, Las Vegas, June 1993.
- [19] Stephen V. Rice, Junichi Kanai, and Thomas A. Nartker. The third annual test of OCR accuracy. Technical Report 94-03, Information Science Research Institute, University of Nevada, Las Vegas, April 1994.
- [20] Ray Smith. A simple and efficient skew detection algorithm via text row accumulation. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 1145–1148, Montreal, Canada, 1995. IEEE Computer Society Press.
- [21] A. Lawrence Spitz. Skew determination in ccitt group 4 compressed document images. In *Proc. of Symposium on Document Analysis and Information Retrieval*, Las Vegas, NV, March 1992.