Chapter 2
Database

Database plays an important role in pattern recognition specially in handwriting recognition tasks. To compare different algorithms and select the best ones, researchers need to use common databases. In this thesis to compare our proposed word recognition technique with some existing techniques, we have used two standard handwritten word databases: (i) CENPARMI database and (ii) IAM database. Brief description of the two benchmark databases are given below. Here we have also proposed a new database, namely Indian Statistical Institute Hand Writing Database (ISIHWD). The detail description of proposed database will be explained later.

2.1 Benchmark databases

CENPARMI database: It is an English legal amount off-line handwritten word database collected from 2500 handwritten Canadian Bank cheques written by about 800 person including the staff of Concordia University. We have chosen this database because many researchers have used it as standard and have shown the performance of their systems on this database. The bank cheques are scanned at 300 dpi, transformed into binary images, and then segmented into tagged words. Some words starts with uppercase letters and others with lowercase letters. This database contains isolated preprocessed words of 32 different word classes including ‘hundred’, ‘thousand’, ‘and’, ‘only’ and ‘dollar’. The sample size
of word classes have large variation. For example word class ‘twelve/Twelve’ has
53 samples where word class ‘hundred/Hundred’ has 818 samples. The developer
of the database has partitioned it into training (5230 words) and test (2514
words) data-sets. Through out our experiment we have followed this partition
for performance analysis of our system for word recognition. This database can
be available in https://www.concordia.ca/research/cenparmi/resources.html.

**IAM Handwriting Database:** IAM off-line handwriting database [77] devel-
oped by Marti et al. in 2002 is a large vocabulary database containing forms
of unconstrained handwritten English text scanned at a resolution of 300 dpi
in gray-level. We have tested our algorithm on IAM500 and IAM200 that are
formed by Pinales et al. in [97] from original IAM database. IAM500 contains
23 word classes having at least 500 samples per class whereas IAM200 contains
55 word classes having at least 200 samples. It is to be noted that IAM500
and IAM200 contain several types of variations (incomplete words, missing or
additional strokes due to segmentation errors and background noise and other
degradation). IAM200 and IAM500 data-set contains 42560 and 32753 words re-
spectively. As in [97], Pinales et al. have not provided fixed partition of training
and test set, we have randomly partitioned the data set into training set (con-
taining 80% words) and test set (containing rest 20% words) for ten iterations.
All experimental results for IAM database reported in this thesis are the aver-
age recognition rate of all ten iterations. This database can be downloaded from

### 2.2 ISIHWD database generation:

A new off-line handwritten word database is proposed here. As handwritten word
recognition is very complex and requires addition supplementary knowledge such
as dictionary, domain knowledge about the document, we have restricted our
domain to legal amount written in words. This has immediate application in
automatic bank check processing. Here we present a new database (ISIHWD) of
off-line handwritten English legal words commonly used in Indian bank cheques.
Thus the database contains the legal amount words generally found in the bank
cheques written in English script. To build this database, we have selected 62
words whose combination may represent any legal amount of Indian bank cheque and also in other relevant countries. The words are: One, Two, Three, Four, Five, Six, Seven, Eight, Nine, Ten, Eleven, Twelve, Thirteen, Fourteen, Fifteen, Sixteen, Seventeen, Eighteen, Nineteen, Twenty, Thirty, Forty, Fifty, Sixty, Seventy, Eighty, Ninety, Rupees, Only, hundred, thousand, lakh, crore, one, two, three, four, five, six, seven, eight, nine, ten, eleven, twelve, thirteen, fourteen, fifteen, sixteen, seventeen, eighteen, nineteen, twenty, thirty, forty, fifty, sixty, seventy, eighty, ninety, rupees and only. Note that if we remove six specific words (e.g. Rupees, Only, lakh, crore, rupees and only), then database can be used for more general purpose. The words are written by common educated people of West Bengal, India. This database contains 31124 handwritten words written by 105 different writers. It is to be noted that all writers have written all 62 words for several times. Our database has writer tag with all words as a part of ground truth.

All writers are requested to write the aforementioned 62 words five times in a plain white A4 sheet using black ink. Example of handwritten 62 words written by three writers are shown in Fig. 2.1. Writers are allowed to write freely in their own style without any constraint such as maintaining strictly horizontal alignment or maintaining word or character gaps. Then the A4 sheets are scanned in 300 ppi (pixel per inch) resolution in gray scale. Then all the scanned image are converted to binary image using Otsu’s binarizing method [86]. Though most of the writers have written five samples of each word, but few have written four or three samples. To add more writing variation, we have included all of the samples in our database. We have collected 105 different writer’s handwriting. Among them, 90 writers have written the above mentioned words five times, 7 writers have written the same four times, and the rest 8 writers three times. All together we have collected (90x62x5 + 7x62x4 + 8x62x3) 31124 words in our database. Though our database contains 62 different legal amount words but they may be grouped into 33 word classes. They are {One, one}; {Two, two}; {Three, three}; {Four, four}; {Five, five}; {Six, six}; {Seven, seven}; {Eight, eight}; {Nine, nine}; {Ten, ten}; {Eleven, eleven}; {Twelve, twelve}; {Thirteen, thirteen}; {Fourteen, fourteen}; {Fifteen, fifteen}; {Sixteen, sixteen}; {Seventeen, seventeen}; {Eighteen, eighteen}; {Nineteen, nineteen};
Figure 2.1: Example of 62 legal words written by three writers of our database.
{Twenty, twenty}; {Thirty, thirty}; {Forty, forty}; {Fifty, fifty}; {Sixty, sixty};
{Seventy, seventy}; {Eighty, eighty}; {Ninety, ninety}; {Rupees, rupees}; {Only, only};
{hundred}; {thousand}; {lakh} and {crore}.

Since handwritten words for developing our database are written as a running
text, extraction of handwritten words from text needs the following preprocessing
steps: (i) Text line segmentation and word separation, (ii) Slope correction (iii)
Slant correction, and (iv) Space compaction. Each steps are explained below. For
some recognition system system thinning is also a required step.

2.3 Preprocessing

2.3.1 Text line segmentation

Problem formulation:

Here we propose a new method of line segmentation considering a very sim-
ple pen-tip movement model. The handwritten texts contain words with several
slants and slopes, which contributes complicacy to the recognition process. Thus
for correct recognition, we have to take the features which are slant and slope
independent. This is very difficult. To avoid this difficulty, each word is made
horizontal with ascenders and descenders aligned along the vertical direction. An
explanation for the selection of vertical direction is also given here. Let us sup-
pose that we want to write something with a pen on a board hanging on a wall.
Now the motion of the pen-tip is of our interest. Generally, this motion may be
determined by the resultant of four different forces. They are: (i) Gravitational
force, which is downward (ii) transational force which is dependent on language
(for English, Bengali, Hindi etc. it is horizontal and left to right), (iii) pattern
generating force, which depends on shape of the character, and (iv) force due to
pen pressure. In case of writing on a vertical surface gravitational force combines
with translational and pattern generating forces, while in case of horizontal sur-
face gravitational force combines with the force due to pen pressure. In either of
these cases the writer applies a force on the writing surface along a slant direc-
tion. Thus an equal and opposite force is applied to the writer's hand. This force
can be broken into vertical and horizontal components, of which vertical force
component is nullified by gravitational force and the horizontal force component combines with the translational force. There is also a frictional force, which opposes the translational and pattern generating forces. Translational force includes error-correcting force controlled by feedback through visual system to keep the text line along the direction of script as far as possible.

The translational force usually is a combination of two forces guided by (i) writing direction of script along with writing speed and (ii) orientation and stretch of writing hand. Readers may have noticed that handwritten lines are more undulated if writing hand is in uncomfortable pose. Because of our habit, handwritten lines on the vertical surface have higher variability in the baseline-skew than that on the horizontal surface. So here we show that the non-intersecting nature of text lines for writing on vertical surface explicitly similar nature of lines written on horizontal surface may be shown by corresponding modifications.

Let us suppose the pen-tip is represented as a ball attached to a vertical surface. Now we have to study the motion of the ball while writing on a board. To show non-intersecting nature of two successive text-lines, let us consider that two balls placed at two vertically separated locations on the board are initially at rest. During writing, the gravitational force ($\vec{G}$), the translational force ($\vec{T}$) and the pattern generating force ($\vec{p}$) are acted upon these balls to determine the path followed by them. If the writer’s hand is in uncomfortable position then he or she also tries to move his or her hand gradually to achieve comfort. As a result, prominent baseline-skew is noticed. Now let us see the effect of these $\vec{G}$ and $\vec{T}$ on the balls. First we assume that the writer writes in comfortable pose. Thus we take angle between $\vec{G}$ and $\vec{T}$ is $90^0$ [Fig. 2.2 (a)].

$$\vec{R'} = \vec{G} + \vec{T}$$

where $|\vec{R'}| = \sqrt{|\vec{G}|^2 + |\vec{T}|^2}$ and $\tan \theta = \frac{|\vec{T}|}{|\vec{G}|}$.

It may be noted that a frictional force opposes this $\vec{R'}$ due to pen pressure. Let the resultant force, incorporating this opposing frictional force, is denoted by $\vec{R}$. For writing on horizontal plane this resultant force $\vec{R}$ is a vector sum of horizontal component of the diagonal force applied to writer hand, translational force and frictional force. Also note that when the person is not writing, the
pen-tip lies at rest because of this frictional force due to pen-pressure. If the mass of the each of two balls are \( m \), then we can say that due to the application of \( \vec{R} \) on them, they will move with same acceleration \( \vec{a} = \frac{\vec{R}}{m} \) and same velocity \( \vec{v} = \vec{a}t \) at angle \( \theta \) with respect to \( \vec{G} \). As the initial positions of balls are vertically spaced, so the paths described by these balls are two parallel lines as shown in fig. 2.2(b).

Now we consider the pattern generating force \( \vec{p} \). The magnitude and direction of \( \vec{p} \) depends on the pattern to be generated, i.e., the characters to be written. Usually the overall amplitude of \( \vec{p} \) is much smaller than \( \vec{T} \) and \( \vec{G} \). Thus when we combine the pattern generating force \( \vec{p} \) with \( \vec{R} \) the paths described by the balls are no longer smooth parallel lines, but have some zigzag nature which depend on patterns present in the text. Study on alphabet patterns reveals that the text is written about a baseline. That means the force \( \vec{p} \) is applied in such a manner that the patterns are formed about the two parallel lines as shown in Fig. 2.2(c).

Let us now assume that the maximum deviation of the pen-tip from the base line due to pattern generation is \( d \). Thus it can be said that the path described by the balls are always nonintersecting if the vertical separation of starting point of the balls is greater than \( 2d \).

Now if the writer’s hand is in uncomfortable pose, the angle between translational force and gravitation force may not remain 90°. Second, the discomforting feeling of the writer may also depend on position. For example, a writer has to stretch his/her hand more at the end of a line or he/she may write the whole text line in an uncomfortable pose. The latter situation may occur when a short height writer writes something on the upper portion of a vertical plane, or a tall writer on its lower portion. Usually these uneven forces are more or less same for the above mentioned two parallel base lines. Moreover, magnitude of pattern generating force is also non-uniform, both time and position variant. But variation in this force is significantly less than the variation in the previous forces. So if we take all these forces together, there will be a change in slope of the parallel lines as shown in Fig. 2.2(d). The change in slope of the baseline then depends on the orientation of hand in the comfortable zone. In this case the path followed by the balls are again non-intersecting if the vertical separation of starting point of the balls is greater than \( 2(d + b) \), where \( b \) is the deviation in slope. Fig. 2.2(d)
shows the path followed by the balls in case of a writer’s uncomfortable hand position at the end of line. However, two text lines may touch or intersect which sometimes happens, if this condition does not hold.

Here the balls represent pen-tips. Thus the paths described by the balls are two successive handwritten lines. Finally, an error correcting force to modify the slope of baseline is always in action trying to keep successive text lines non-intersecting as far as possible. For example, text lines written blind-folded, where this correcting force is absent, may intersect significantly. So it is found that successive handwritten lines are mostly non-intersecting if the vertical separation of the starting positions of two balls is greater than 2d for comfortable writing and 2(d+b) for uncomfortable writing. Thus two successive handwritten lines are always nonintersecting. Generally, the stability of a body puts resistance to

Figure 2.2: (a) Resultant vector $R'$ of gravitational force(G) and translational force(T). (b) Path described by the pen-tips under resultant force are parallel. (c) Comfortable writing: Path described by the balls under resultant force and pattern generating force are always nonintersecting if the starting point separation is greater than two times of maximum possible deviation(d). (d) Uncomfortable writing: Path described by the balls are always nonintersecting if the starting point separation is greater than twice the maximum possible deviation(d) plus deviation due to slope change(b).
a disturbance for the body’s equilibrium. When an object is in equilibrium, the weight of one side of its centre of gravity is counter balanced by the weight of other side. Centre of gravity (COG) of a body is the point at which entire mass or weight of the body is equally balanced. The whole weight of the body, or body segment, is considered to act vertically downwards through the COG of the body or body segment. A body is in stable equilibrium when this force passes through the base of support, i.e., the supporting area beneath the body. A body is in stable position depending on where the COG is in relation to the base of support. The stability is less if the vertical line passing through COG is near the edge of the base of support. If this line is outside the base of support, the body will lose its equilibrium due to a turning effect. Now let us consider the image of a word as an object, whose pixel value is treated as mass of that position. Thus the centroid behaves like centre of gravity. Here the base of support is the base line on which the word is written. According to above explanation, the word is in most stable position when the vertical line through centre of mass intersect the base of support. It is evident that in this position, the ascenders and descenders of a word are vertical. Thus before feature extraction, the base line of handwritten word is made horizontal with its ascenders and descenders vertical.

**Proposed method:**

The proposed text-line segmentation algorithm has two main steps. In the first step the system takes the raw image of the given text as an input and all the connected components are labeled. In the second step, all the connected components are linked to form text lines.

After accepting the raw image, all the connected components of it are identified by using two-pass algorithm [95]. Then for each connected component a bounding box is found. Now if two characters, one from a text line and other from the adjacent text line touch each other, a bounding box with large height is formed. To avoid this problem we set a threshold height $(H_T)$ using the equation, $H_T = \mu + 3\sigma$, where $\mu$ is the mean and $\sigma$ is the standard deviation of the height of the boundary boxes respectively. Now the bounding boxes with height greater
than $H_T$, are broken into two boxes having half of its previous height as shown in Fig. 2.3. Their widths are refined then and the centroids of all boundary boxes are determined.

![E]ighty
![f]ive

Figure 2.3: (a) Touching component of a handwritten text. (b) Bounding box with height greater than threshold-height is formed. (c) Bounding boxes with half of its previous height are formed and widths are refined.

Now as the handwritten text-lines are always nonintersecting, centroids of all the components of a line can be arranged properly by using some simple rules based on the following two assumptions:

1. Two successive bounding boxes of a text line must have sufficient vertical overlap.

2. The difference in vertical coordinates between two successive centroids is small.

Here we define a set of vertical coordinates as $h_i = [Y_{i1}, Y_{i2}]$ of the left side of bounding box and consequently the over-lapping area as

$$\text{over-lapping area} = \frac{|h_1 \cap h_2|}{\min\{|h_1|, |h_2|\}}$$

where $|h_1|$ and $|h_2|$ are the heights of two successive bounding boxes respectively, as shown in Fig. 2.4.

Following example illustrate the text line segmentation / extraction algorithm. Let us consider Fig. 2.5. Here the nine rectangles represent the bounding boxes of nine components and a, b, c, d, e, f, g, h and i represent the corresponding
Figure 2.4: Vertical overlap between of two boundary boxes: \((h_1 \cap h_2)/h_1\).

Figure 2.5: Text having nine connecting boxes with their corresponding centroids.

centroids. Now the arrangement procedure followed by text line extraction is explained clearly by using tree shown in Fig. 2.6. First, all the centroids are arranged in ascending order with respect to the vertical coordinate. Thus the arrangement becomes \([c, b, a, d, e, g, f, i, h]\). Here the first centroid indicates the top most bounding box of the given text and this is set as a Root of the tree as shown in Fig. 2.6(a). Thus the Root denotes the top most bounding box of a line, the left sub tree of the root contains the possible boundary boxes or components, which are on the left side (with respect to horizontal coordinate) of their Root and the right sub tree of root contains the boxes on the right side of their Root. Now among the nodes of the left sub tree which satisfy the following two conditions first are selected as the previous boxes.

1. The vertical overlapping area between a node (bounding box) and a selected node (bounding box) on its immediate right is greater than a threshold.

2. The horizontal distance between two bounding boxes (centroids) is less than a threshold value.

Proceeding in this way the portion of the line which is on the left side of the root
Figure 2.6: Centroids arrangement. (a) Tree for the first line identification. (b) Centroids present in the first line are connected. (c) Tree for second line identification. (d) Centroids present in the second line are connected.

is found. Similarly the portion of the line which is on the right side of root is also found. Though two thresholds are to be set, their values are not critical and can be selected from a wide range. After selecting the first line all the centroids of the line, are deleted from the tree. Fig. 2.6(c) shows the modified tree. And again in similar fashion second line is identified as shown in the Fig. 2.6d, and so on. In Fig. 2.6(a) and (c) the selected centroids are shown in large font.

Experimental Results:

At first a database of offline handwritten text images is generated from the answer script of students to train the system. We have deliberately chosen such text images as the text is written hurriedly without almost any constraint. Then
the text images are scanned at 300 dpi. We have trained our algorithm on 40 such handwritten pages in English script written by different writers. The database contains 300 text lines with an average of 20 connected components in each. The pages contain pure text blocks including different mathematical symbols and punctuation. We have implemented our algorithm in MATLAB 6.5 software. As a result of straightforward implementation of the proposed algorithm, out of 300 text lines 279 lines are correctly identified. Figs. 2.7(a, b, c, d) show some of the handwritten texts with correct line segmentation. The peak value of the distribution of existing line gaps is set as the threshold for vertical coordinate difference between two successive centroids. Here the existing line gaps are measured by counting the zero-run which are greater than 10 for each column. The threshold for vertical overlap between two successive connecting components as 0.25 and the threshold for horizontal separation of bounding boxes five times of the mean horizontal separation between adjacent components.

The failure of our algorithm is mainly due to insufficient overlap between two adjacent connected components. An example is shown in Fig. 2.8. Sometimes the vertical overlap between two successive connected components becomes very small or even zero. Such incident occurs if one of the connected components is a punctuation, such as ‘,’ ‘.’ etc., or dots of ‘i’ and ‘j’. In that case we ignore small bounding boxes. With improper threshold for the size of the bounding boxes, this constraint may also eliminate small bounding boxes containing part of letter or even a small letter.

Now to improve the performance of our system, we first determine the average altitude of all segmented text lines. Here by the term ‘average altitude’ we mean the average of the vertical coordinate of the centroids present in a text-line. Then we merge the lines whose average altitude differences are less than a threshold value. Here we have taken the threshold for altitude difference equal to 20. As a result performance of the system is improved.

We have also tested our algorithm on the IAM database for offline handwriting recognition research. This database contains various forms of unconstrained handwritten text which is scanned at a resolution of 300 dpi and saved as tiff images with 256 gray levels. We have tested our algorithm on 100 pages containing about 1000 text lines written by 40 different writers. First, the gray images are
converted to binary images and then proposed algorithm is applied to measure the system performance. Considering both databases the overall performance of our scheme becomes 99.6%. Whereas for the smearing approach [65] the overall performance of line-segmentation is 85.6% and for the block based Hough transform mapping [71] is 96.87%. Thus our proposed line segmentation algorithm works good for natural writing and also complicated speed writing (exam sheet).

We have extracted text line of ISIHWD database using the above algorithm under manual supervision. Line segmented text of ISIHWD database is shown in Fig. 2.9. After line segmentation, handwritten words are extracted from text line. First vertical projection of text line is estimated. Then the location of columns having no black pixel are identified. The width of all zero runs are estimated and the mean of zero run widths is set as threshold width. Finally we have set cutters at the center of those zero runs whose widths are greater than the thresh-
2. DATABASE

Figure 2.8: Results: (a) Segmented text lines at first attempt. Errors occur in first line after ‘.’ and in second line near the dot of ‘i’ of window. (b) Corrected text lines after merging the lines whose average height difference is less than 20.

2.3.2 Slope correction

Slope of handwritten word is defined as an angle between the intended (horizontal) direction of writing and the actual direction of the implicit line along which the word is aligned. This implicit line is called base line of word. For natural writing (in targeted languages) the baseline should be horizontal. But in practice rarely this situation occurs. Thus slope correction is needed. In this thesis we have proposed two new slope correction techniques. Slope of the base line is taken as the slope of the word. In slope correction, the word is rotated such that the lower baseline is aligned to the horizontal axis of the image.

Method-1
Figure 2.9: Example of line segmented image of 62 legal words written by one writer.

Figure 2.10: Example of word segmented text-line following projection method.
In the proposed method slope of word is estimated from the slope of base line. As ascender and descender has no contribution in base-line formation, we first select the core region of the word. Then from the lower boundary of the core region, best-fit line is estimated. This is done in two steps. First, a rough core region of the word is determined from the distribution of horizontal black to white transition based on the fact that the number of black to white transitions in a row is higher in the core region than it is in the ascender and descender regions. Then the lower boundary of the core region is selected. Next the selected part is divided into small narrow vertical strips. For each strip the centroid is determined and then the best-fit straight line (base line) is drawn considering these centroids.
which have vertical coordinates less than the leverage [7] of the vertical coordinate of all centroids. Leverage is a measure of how far an observation deviates from the mean of that variable. These leverage points can have an unusually large effect on the estimate of regression coefficients. Mathematically, leverage (Le) is the average of \( le_i \), where \( le_i \) is calculated using the following equation.

\[
le_i = \frac{1}{N_i} + \frac{(y_i - \bar{y})^2}{\sum_{i=1}^{N_i}(y_i - \bar{y})^2}
\]  

where \( N_i \) is total number of centroids, \( y_i \) is vertical coordinate of \( i^{th} \) centroid and \( \bar{y} \) is mean value of \( y_i \). By this we eliminate some outlier centroids contributed by ascender or descender or sides of characters in a word. Then the slope of the baseline is calculated based on these centroids. Finally, the slope is corrected by rotating the word about its centroid by negative of the slope angle. The results of different steps of slope correction technique is shown in Fig. 2.11.

**Method-2**

In this approach the spacial distribution of black pixels in text word image is studied. In most of the script including English, the text-words are written along a baseline. In the proposed method, slope of text word is estimated from the slope of base line. As ascender and descender have no contribution to baseline formation, they are discarded as much as possible and then a best fit straight line is estimated from the remaining part of the word. Slope of the baseline is found and the slope is then corrected by rotating the word about its centroid by the slope angle. The ascender and decender are eliminated in two steps.

In the first step, coordinates of all the black pixels of text-word image is stored in a two dimensional array. Let the total number of black pixels in an image is \( n \). \( X \) stores all the x coordinates and \( Y \) stores all the y-coordinates of black pixels.

\[
X = [x_1, x_2, x_3...x_n] \quad Y = [y_1, y_2, y_3...y_n]
\]

\( \bar{x} \) and \( \bar{y} \) are found as mean(X) and mean(Y) respectively. Then using a transformation (Tr), \( X' \) and \( Y' \) are created as follows.

\[
x'_i = x_i
\]  

(2.3.3)
\[
y'_i = \begin{cases} 
y_i + 2(\bar{y} - y_i) & \text{if } y_i > \bar{y} \text{ and } x_i < \bar{x} \\
y_i & \text{or } y_i < \bar{y} \text{ and } x_i > \bar{x} \\
y_i & \text{otherwise}
\end{cases} \tag{2.3.4}
\]

The effect of the transformation can be visualized from Fig. 2.12. Fig. 2.12(b) is the transformed image of Fig.2.12(a). Then the modified word (set of black pixels) is shifted to touch the top and left edge of the image. Thus \(X'\) and \(Y'\) are modified to \(X''\) and \(Y''\) as follows.

\[
X'' = X' \tag{2.3.5}
\]

\[
Y'' = Y' - \min_i(Y') + 1 \tag{2.3.6}
\]

Subsequently \(M''\) is computed as simple term-wise algebraic product:

\[
M'' = X''Y'' \tag{2.3.7}
\]

Now the black pixels of the modified text word that are outside of first and forth quartiles of \(M''\) are found and discarded. Then the best fitted straight line is estimated from the remaining black pixels. In the next step perpendicular distance \(d_{pi}\) of each black pixel of the original word to the estimated line is calculated. Mean of these distances \(\bar{d}_{p}\) is calculated and is set as threshold. The black pixel of the original word whose perpendicular distance from the estimated line is greater than this threshold are discarded. Then again the best fit line is estimated from the remaining black pixels. This line is the baseline and the slope of the baseline is found. Then slope correction of the text word is done. Result of each steps of our algorithm is explained with an example in Fig. 2.12. It is clear from the above description that the first step provides a reasonable gauss of the base. However, the second step determines the baseline by ignoring ascenders and descenders (by discarding black pixels at a distance greater than \(\bar{d}_{p}\)). The algorithm of proposed method is summarised as follows.

**Algorithm:** Slope correction of handwritten word  
**Input:** Word image.
Output: Slope corrected word image.

step 1: Form X and Y using equation 2.3.2.

step 2: Keep a copy of X and Y: Store $X_0 = X$ and $Y_0 = Y$.

step 3: Compute $\bar{X} = mean(x)$ and $\bar{Y} = mean(Y)$.

step 4: Compute $X'$ and $Y'$ using equation 2.3.3 and 2.3.4 respectively.

step 5: Compute $X''$ and $Y''$ using equations 2.3.5 and 2.3.6 respectively.

step 6: Determine $M''$ using equation 2.3.7.

step 7: Discard the pixels $(x,y)$ from X and Y which corresponding to first and forth quartile of image $M''$.

step 8: Determine best fitted straight line L over the remaining pixels of X and Y.

step 9: Recover X and Y: $X = X_0$ and $Y = Y_0$.

step 10: Calculate perpendicular distance, $dp$ of the estimated line L from (X,Y).

step 11: Estimate $\bar{dp}$, the mean distance of dp’s.

step 12: Discard pixels $(x, y)$ of $(X,Y)$ whose dp is greater than $\bar{dp}$.

step 13: Estimate the final best fitted straight line $L'$ over the remaining pixels as the baseline of word.

step 14: Estimate the slope $\theta$ of this baseline $L'$ as slope of the word.

step 15: Give equal and opposite rotation by the angle $\theta$ to word image.

Result: Since this belongs to a class of computer vision problem we argue that best slope estimation of handwritten word can be done by human beings. Thus slope estimation by a method should be compared with manual ground truth. Root mean square of difference of estimated slope angle and the ground truth is taken as evaluation of performance. Here manual ground truthing of
Figure 2.12: (a) A sample text-word. (b) Modified word after applying transformation (Tr). (c) Word after discarding the Black pixels present in first and forth quartile of modified word. (d) Best fit straight line estimation. (e) Text word after eliminating the black pixels whose perpendicular distance from the estimated line is greater than threshold. (f) Estimation of Base line of the text word. (g) slope corrected word.

slope is done by drawing the base-line over text word by five persons and finding the average.
We have applied proposed two slope correction techniques to all handwritten words of our database and have compared the result with two well-known slope correction techniques, entropy method and eigen vector method. In the former case, the entropy of the horizontal projection \cite{63} is computed while in eigen vector method principal axis is determined \cite{25}. The comparative results are shown in Table 2.1. The study shows that our proposed Method-2 incurs less error. Thus for slope correction of handwritten words we have used Method-2.

Here we have implemented the algorithms using MATLAB 2014a software on Intel Xeon®CPU W3350 @3.07 GHz, with 24GB RAM runing on Windows7. Fig. 2.13 shows some handwritten text words before and after slope correction.

![Figure 2.13: Sample words and Slope corrected words.](image)

### 2.3.3 Slant correction

Here we propose three new slant correction techniques for off-line handwritten words. In Method 1 the global slant estimation is done exclusively by Gabor filter while in Method 2 global slant is estimated by Fourier transform of word image...
Table 2.1: Performance comparison of different methods for slope correction.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>error(rms)</td>
<td>7.79</td>
<td>10.38</td>
<td>5.23</td>
<td><strong>3.99</strong></td>
</tr>
</tbody>
</table>

and then local slant is estimated by using Gabor filter and in Method 3 global slant is estimated using Hough transform.

**Method 1**
Before going into details of this method, a brief description of Gabor filter follows. It is well known that directional texture can be detected and analyzed efficiently by means of Gabor filter. This ability is inherent to Gabor filter as it can provide a filter bank depending on variation in length as well as angle parameters. Here we intend to utilize this capability of Gabor filter for slant correction. A Gabor filter is a linear filter whose impulse response is the product of a Gaussian function and a harmonic function.

\[
 h(x, y) = g(x, y)s(x, y) \quad (2.3.8)
\]

where \( s(x, y) \) is a complex sinusoidal known as the carrier and \( g(x, y) \) is a 2-D Gaussian-shaped function known as the envelop. The real valued response of 2-D Gabor filter in spatial domain may be obtained by convolving the 2-D function of image with

\[
 Ga(x, y; \lambda, \theta, \sigma_x, \sigma_y, \phi) = \exp\{-0.5(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2})\} \cos(2\pi \frac{x \theta}{\lambda} + \phi) \quad (2.3.9)
\]

where,

\[
 \sigma_x = \frac{\lambda}{\pi} \sqrt{\frac{\log(2)}{2}} [1 + 2BW] \quad (2.3.10)
\]

\[
 \sigma_y = \frac{\sigma_x}{aspect ratio} \quad (2.3.11)
\]

\[
 x_\theta = (x \cos \theta) + (y \sin \theta) \quad (2.3.12)
\]

\[
 y_\theta = (-x \sin \theta) + (y \cos \theta) \quad (2.3.13)
\]
where $\sigma_x$ and $\sigma_y$ control the spatial spread and are the standard deviations of the Gaussian envelope along $x$ and $y$ direction, $x_\theta$ and $y_\theta$ are the $x$ and $y$ coordinates in the rotated rectangular coordinate system and $\theta$ is the rotation, $\phi$ is the angular phase shift, $\lambda$ is the wavelength of carrier and $BW$ is the bandwidth of Gaussian envelope.

To estimate the global slant angle of a word, a Gabor filter is chosen whose $\lambda$ is equal to the height of the core region of slope corrected word. First, the core region of text-word is determined from the horizontal count of black to white transitions. Then Gabor filter with suitable $\sigma_x$ and $\sigma_y$ is chosen. For each angle in an interval, Gabor filter is rotated and convolved with the word image. The angle for which the absolute sum of all pixel values of the convolved image is maximum is taken as the global slant angle of the word. The global slant is corrected by shearing effect in the reverse direction. An example is shown in fig 2.14.

![Figure 2.14](image)

Figure 2.14: a. Slope corrected word. b. Suitable Gabor filter estimates the global slant angle. c. slant corrected word.

**Method 2**

Generally Fourier transform, converts spatial domain signal to frequency domain information. Thus in Fourier spectrum of a word image all dominant spatial frequencies present in the image are highlighted. Binary image of text-word is a distribution of black points on white background. Repetition of black points along a direction gives rise to spatial frequency along its perpendicular direction. Therefore in Fourier spectrum domain, two prominent spatial frequencies are noticed: One is for slant of the word and other is for slope. As our input is
slope corrected word image, spatial frequency for slope of the word is along the
vertical axis. Thus in the frequency domain, the spatial frequency due to global
slant of the word produces a strong non-vertical line. From the orientation of this
dominant component, global slant can be estimated. Thus slant correction is done
by shearing the image at an angle equal to $\theta$ which is $180^\circ$ minus the orientation
of spectral component responsible for slant angle($\alpha$) as shown in fig 2.15.

This new slant estimation technique is done in two steps. First Fourier spectrum
of the word image is found by FFT. Then the orientation of most prominent line
in the spectrum is estimated using Hough transform. As slant of the a word is
global as well as very local, we calculate the average angle due to most prominent
three Hough peaks. Thus the global slant angle $\theta_s$ is estimated and corrected by
shearing the word at an angle $180^\circ - \alpha$. Then local slant correction is done based
on the assumption that the local slant information is prominent in the ascender
and descender regions only. Thus for local slant estimation, the core region of
the global slant corrected word is eliminated and all connecting components of
the remaining image are labeled. The connecting components of sufficient height
is under gone through local slant correction. This is done by similar way as
discussed in Method 1 (of slant correction). For this the $\lambda$ of Gabor filter is taken
to be equal to the half of the height of the core region of word. Then local slant
is eliminated by shearing again these connecting components. The result of slant
correction technique (Method 2) are shown in fig 2.15.

**Method-3**

Here we correct the slant of word image by applying shear to its connected compo-
nents in the opposite direction of average orientation of its near-vertical strokes.

Average stroke width of text-word is calculated as

\[
stroke \ width = \frac{\#[\text{Word}]}{\#[\text{Thinword}]} \tag{2.3.14}
\]

where $\#[\text{Thin word }]$ is black pixel count of one pixel thick word obtained by
thinning and $\#[\text{Word }]$ is black pixel count of original word. First, the connected
components of a word image are labeled. Then each component is opened by a
vertical line structural element of length, say $L_s$ so that the strokes non-useful for
Figure 2.15: a. Slope corrected word. b. Spectrum of image word. Vertical spacial frequency is for slope. Global slant correction is done by shearing image at angle $\theta$. c. Word after global slant correction. d. Core region of word is eliminated. e. Word after local slant correction.

Slant estimation are eliminated and each component is splitted into near-vertical strokes. The choice of $L_s$ is, in this case, a bit critical. Let us assume that the maximum slant (or orientation of near-vertical strokes) is $\theta_m$ and the stroke width is $w_s$ as shown in Fig. 2.16. Then

$$L_s = \frac{w_s}{\tan \theta}$$  \hspace{1cm} (2.3.15)

Small length strokes (if it is not only stroke) are omitted. Then for each of the
remaining strokes of a connected component most dominating stroke orientation
is determined by Hough transform. In general, it is noticed that when a writer
writes in continuous stroke the variation in local slant of those stroke is small,
we correct the slant angle of a connected component by shearing the component
in the opposite direction of weighted average of slant angles present in those
strokes.

The weight is determined based on the height of a stroke. Suppose the set \( Y \)
contains the y-coordinate of all black pixels of j-th stroke. Then weight \( W_j \) may
be defined as

\[
W_j = \max\{Y_j\} - \min\{Y_j\} \quad (2.3.16)
\]

Finally, slant angle \( \theta_s \) of the connected component is given by

\[
\theta_s = \sum W_j \theta_j \quad (2.3.17)
\]

where \( \theta_j \) is the slant angle of the j-th stroke of the connected component. The
slant correction method is explained in Fig. 2.17 with an example. Note that
after slant correction the dot of ‘i’ remains in its old position, which is now away
from the vertical part of ‘i’. This however, is not a significant factor in the holistic
approach for word recognition.

Figure 2.16: Length of structural element (L) selection if \( \theta \) is maximum slant
angle of stroke and \( b \) is stroke width.

**Selection of the best slant correction method**

To select the best slant correction method first results obtained by all slant cor-
rection methods are thinned to single pixel scaled to a predefined height and
then each slant corrected scaled word is opened by a series of vertical structural elements of length $L_s$, where $L_s$ varies from two to predefined height of the word. Then accuracy is computed as,

$$\text{accuracy} = \text{average}(B(L_s)\text{count} \times L_s) \quad (2.3.18)$$
Table 2.2: Performance comparison of different methods for slant correction.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>average accuracy</td>
<td>520.50</td>
<td>508.28</td>
<td>525.79</td>
<td>530.20</td>
<td><strong>531.80</strong></td>
</tr>
</tbody>
</table>

where $B(L_s)_{\text{count}}$ is the black pixel count of the word opened by structural element of height $L_s$. So higher accuracy means better slant correction.

**Results:**

We have taken slope corrected text-words of our database. Here we have taken the height of structural element for opening as 1.5 times of stroke width. The performance of three slant correction methods are compared with two standard slant correction methods: entropy based method and variance based method [43],[89]. A comparative study is shown in Table 2.2. This study reveals that our proposed Method-3 is better than other. Thus we have corrected the slant of all slope corrected word of our database using Method-3. Example of some slant corrected words are shown in Fig.2.18.

![Sample text-word](image1.png)

Figure 2.18: (a) Sample text-word whose slant to be corrected. (b) slant corrected word.
2.3.4 Space compaction

Usually a handwritten word in cursive writing remains a single connected since the pen is not lifted while writing. However, sometimes it happens that, because of lifting the pen up, a handwritten word gets fragmented into a several connected components. As a result, length of the word image increases along the horizontal direction without providing any extra information. To reduce this extra vertical space, each column of slope and slant corrected word image is scanned and is discarded if there is no black pixel present. Results of all three preprocessing steps are shown in Fig. 2.19. The gray columns in Fig. 2.19(c) contain no black pixel and discarded in Fig. 2.19(d).

**Partition:** Finally, to provide a partition marking training and test set for writer independent word recognition experiment, all words of 20% randomly selected writers from our database and are placed in test set, and the rest are kept as training set. Our test set contains 6200 handwritten words where training set contains 24924 words. Note that handwritten words due to each writer are not present in both training and test sets. Our database can be available on request for research purpose only.

2.4 Summary

In this chapter, we have given detail description of Databases which are used in our experiment to analyze the performance of word recognition system. A
set of proposed methods for handwritten document image preprocessing is also presented along with their performance evaluation. We have also generated a new handwritten word database named ISIHWD. This database contains 31124 handwritten words written by 105 different writers. To provide common platform of all researchers a fixed partition of training and test set is given. It can also be used in writer identification experiments. Here we have also proposed two new methods for slope correction and three novel methods for slant correction of off-line handwritten word images.