A

Project Report

On

**SEGMENTATION AND ALIGNMENT OF TEXT SYMBOLS FROM PRINTED GUJARATI DOCUMENT IMAGE**

Developed at

**Dharmsinh Desai University**

**College Road, Nadiad-387001**

Developed by

**ADESARA HARDIK - IT Department, DD University**

**Guided By**

**PROF. M. M. GOSWAMI**

**Department of Information Technology**

**Faculty of Technology**

**DD University**



**Department of Information Technology**

**Faculty of Technology, Dharmsinh Desai University**

**College Road, Nadiad-387001**

**April – 2015**

**CANDIDATE’S DECLARATION**

I declare that final semester report entitled “**SEGMENTATION AND ALIGNMENT OF TEXT SYMBOLS FROM PRINTED GUJARATI DOCUMENT IMAGE**” is my own work conducted under the supervision of the guide **Prof. M. M. Goswami** from **IT Dept., D. D. University.**

I further declare that to the best of my knowledge the report for B.Tech final semester does not contain part of the work which has been submitted for the award of B.Tech degree either in this or any other university without proper citation.

Candidate’s Signature

Hardik H Adesara

Branch: IT

Student ID: 11ITUOS001

**DHARMSINH DESAI UNIVERSITY**

**NADIAD-387001, GUJARAT**



**CERTIFICATE**

This is to certify that the report entitled “**SEGMENTATION AND ALIGNMENTOF TEXT SYMBOLS FROM PRINTED GUJARATI DOCUMENT IMAGE**” is a bonafied report of the work carried out by **Mr. Hardik Adesara,** Student ID No: **11ITUOS001** of Department of Information Technology, semester VIII, under the guidance and supervision for the award of the degree of Bachelor of Technology at Dharmsinh Desai University, Nadiad (Gujarat). He was involved in Project training during academic year 2014-2015.

Prof. M. M. Goswami

(Project Guide)

Department of Information Technology,

Faculty of Technology,

Dharmsinh Desai University, Nadiad

Date:

Prof. R.S.Chhajed

Head, Department of Information Technology,

Faculty of Technology,

Dharmsinh Desai University, Nadiad

Date:

**ACKNOWLEDGEMENT**

This work would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, I would like to forward my deepest appreciation and gratitude to my project guide **Prof. Mukesh M. Goswami** Associate Professor, Information Technology Department, Dharmsinh Desai University. I sincerely thank my guide for their patience and steadfast encouragement to complete this work.

Special thanks to **respected faculty members** of B.Tech., Information Technology Department, Dharmsinh Desai University for their support and encouragement throughout this work. **Prof. Rajendra S. Chhajed**, Head of the Department, Information Technology Department, Dharmsinh Desai University, who had kind concern and consideration regarding my academic requirements.

Thanks to all of my **friends, classmates & colleagues** for sharing their valuable insights in the relevance of my report and always helping in all possible ways. And also Thanks to **Dharmsinh Desai University** for providing me the platform by which I have been able to precede my final semester training.

With sincere regards,

HARDIK ADESARA

Table of Contents

[ABSTRACT i](#_Toc415868973)

[LIST OF FIGURES ii](#_Toc415868974)

[LIST OF TABLES iii](#_Toc415868975)

[ABBREVIATIONS iv](#_Toc415868976)

[1.0 INTRODUCTION 1](#_Toc415868977)

[1.1 PROJECT DETAILS 1](#_Toc415868978)

[1.2 PURPOSE 1](#_Toc415868979)

[1.3 SCOPE 2](#_Toc415868980)

[1.4 OBJECTIVE 2](#_Toc415868981)

[1.5 TECHNOLOGY AND LITERATURE OVERVIEW 2](#_Toc415868982)

[1.5.1 Optical Character Recognition (OCR) 2](#_Toc415868983)

[1.5.2 Components of an OCR System 3](#_Toc415868984)

[1.5.2.1 Binarization 3](#_Toc415868985)

[1.5.2.2 Noise Removal 5](#_Toc415868986)

[1.5.2.3 Skew Detection and Correction 5](#_Toc415868987)

[1.5.2.5 Segmentation 6](#_Toc415868988)

[1.5.2.6 Recognition 7](#_Toc415868989)

[1.5.2.7 Document Reconstruction / Layout Restoration 7](#_Toc415868990)

[1.5.3 Tools and Technology 8](#_Toc415868991)

[2.0 PROJECT MANAGEMENT 9](#_Toc415868992)

[2.1 FEASIBILITY STUDY 9](#_Toc415868993)

[2.1.1 Technical Feasibility 9](#_Toc415868994)

[2.1.2 Time Schedule Feasibility 9](#_Toc415868995)

[2.1.3 Implementation Feasibility 9](#_Toc415868996)

[2.2 PROJECT PLANNING 9](#_Toc415868997)

[2.2.1 Project Development Approach and Justification 9](#_Toc415868998)

[2.2.2 Milestones and Deliverables 11](#_Toc415868999)

[2.2.4 Project scheduling 12](#_Toc415869000)

[3.0 SYSTEM REQUIREMENT STUDY 13](#_Toc415869001)

[3.1 USER CHARACTERSTICS 13](#_Toc415869002)

[3.2 HARDWARE AND SOFTWARE REQUIRMENTS 13](#_Toc415869003)

[4.0 SYSTEM ANALYSIS 14](#_Toc415869004)

[4.1 REQUIREMENT OF SYSTEM 14](#_Toc415869005)

[4.1.1 User requirement 14](#_Toc415869006)

[4.2.2 System Requirements (SRS) 14](#_Toc415869007)

[4.3 FEATURES OF NEW SYSTEM 16](#_Toc415869008)

[4.4 DATA FLOW DIAGRAM 16](#_Toc415869009)

[5.0 SYSTEM DESIGN 18](#_Toc415869010)

[5.1 INTERFACE DESIGN 18](#_Toc415869011)

[5.2 SYSTEM STRUCTURAL CHART 19](#_Toc415869012)

[5.2.1 Structural Chart Diagram 19](#_Toc415869013)

[5.3 ALGORITHMIC EXPLANATION OF MODULES 19](#_Toc415869014)

[6.0 IMPLEMENTATION PLANNING 26](#_Toc415869015)

[6.1 IMPLEMENTATION ENVIRONMENT 26](#_Toc415869016)

[6.2 PROGRAM SPECIFICATION 26](#_Toc415869017)

[6.3 CODING STANDARDS 27](#_Toc415869018)

[6.4 SAMPLE OF CODE 27](#_Toc415869019)

[7.0 TESTING 29](#_Toc415869020)

[7.1 TESTING PLAN 29](#_Toc415869021)

[7.2 TESTING STRATEGY 29](#_Toc415869022)

[7.3 TESTING METHODS 30](#_Toc415869023)

[7.4 TEST CASES: 30](#_Toc415869024)

[8.0 USER MANUAL 33](#_Toc415869025)

[8.1 WHY WE USE OCR: 33](#_Toc415869026)

[8.2 HOW TO PERFORM ALL TASKS DIRECTLY: 33](#_Toc415869027)

[8.2 INDEXING 39](#_Toc415869028)

[8.4 HOW TO CUSTOMIZE CODE: 39](#_Toc415869029)

[9.0 LIMITATION AND FUTURE ENHANCEMENT 41](#_Toc415869030)

[9.1 LIMITATION 41](#_Toc415869031)

[9.2 FUTURE ENHANCEMENT 41](#_Toc415869032)

[10.0 CONCLUSION AND DISCUSSION 42](#_Toc415869033)

[References 43](#_Toc415869034)

# ABSTRACT

**SEGMENTATION AND ALIGNMENTOF TEXT SYMBOLS FROM PRINTED GUJARATI DOCUMENT IMAGE**

B.Tech. Report by Mr. Hardik. H. Adesara

At

Dharmsinh Desai University, Aprl 2015

OCR has gained popularity because of its wide applicability in automation of the printed documents, processing system or digital processing system. From OCR perspective, the character recognition accuracy of the glyph in the text document depends upon how well the document is segmented.

Gujarati is an Indic script derived from Devnagari script with certain amount of modification. Gujarati has some special features, including combination of Middle Zone character with other Upper Zone character and Lower Zone character together forming one text word. The identification and separation of the Zone in Gujarati text is a huge challenge for OCR. The solution to this problem will result into the increased accuracy of the Gujarati OCR system.

This report presents the various methods of Zone Segmentation and Alignment on Gujarati text. Based on our implementation, we have identified some different methods for segmentation and Alignment, namely Collapsed Horizontal Profile (HP), Vertical Profile (VP), Connected Component (CC), Vertical Bar (VB), Least Distance components (LDC) based methods (applied on Gujarati Script). In this research work, we have implemented VP, HP, CC, VB, LDC methods and evaluated their performance on actual and generated zone and alignment samples.

# LIST OF FIGURES

[Figure 1: Components of OCR System 3](file:///E:\sem8_final_report.docx#_Toc415867521)

[Figure 2: (a) Input Document for Binarization (b) Output of Binarization 4](file:///E:\sem8_final_report.docx#_Toc415867522)

[Figure 3: Input document for Noise Removal 5](file:///E:\sem8_final_report.docx#_Toc415867523)

[Figure 4: Output of Noise Removal 5](file:///E:\sem8_final_report.docx#_Toc415867524)

[Figure 5 : (a) Input Document for Skew Correction (b) Output of Skew Correction 6](file:///E:\sem8_final_report.docx#_Toc415867525)

[Figure 6: (a) Input for thinning (b) After Thinning 6](#_Toc415867526)

[Figure 7: Line segmentation using HP 7](#_Toc415867527)

[Figure 8: Word segmentation using VP 7](#_Toc415867528)

[Figure 9: Iterative waterfall Model 10](#_Toc415867529)

[Figure 10: DFD Level 0 16](#_Toc415867530)

[Figure 11: DFD Level 1 16](#_Toc415867531)

[Figure 12: DFD Level 1 17](#_Toc415867532)

[Figure 13: System Structural Chart Diagram 19](#_Toc415867533)

[Figure 14: (a) Original Image (b) Binarized image 20](#_Toc415867534)

[Figure 15: Image inversion 21](#_Toc415867535)

[Figure 16: Line Segmentation 21](file:///E:\sem8_final_report.docx#_Toc415867536)

[Figure 17: Word Segmentation 22](#_Toc415867537)

[Figure 18: Vertical bar present in a text region 24](#_Toc415867538)

[Figure 19: Zone Segmentation 24](#_Toc415867539)

[Figure 20 : Original Zone images 25](#_Toc415867540)

[Figure 21: Alignment of zones 25](#_Toc415867541)

[Figure 22: Sample code for alignment 27](#_Toc415867542)

[Figure 23: Sample code for Vertical Bar 28](#_Toc415867543)

[Figure 24: Sample code for Zone Segmentation 28](#_Toc415867544)

[Figure 25 : Structure of system 34](#_Toc415867545)

[Figure 26: Flow of functions 35](#_Toc415867546)

[Figure 27: Output in "Bwimage" folder 36](#_Toc415867547)

[Figure 28: Output in "Lines" folder 36](#_Toc415867548)

[Figure 29: Output in 'Words' folder 37](#_Toc415867549)

[Figure 30: Output in 'Zones' folder 38](#_Toc415867550)

[Figure 31: Output in 'SkipWords' folder 38](#_Toc415867551)

[Figure 32: Output in 'Alignment' folder 39](#_Toc415867552)

[Figure 33: Sub function of Main task Function 40](#_Toc415867553)

# LIST OF TABLES

Table 2.1 Milestones and Deliverables 11

Table 2.2 Schedule of project 12

Table 7.3 Test Case 1 31

Table 7.4 Test Case 2 31

Table 7.5 Test Case 3 31

Table 7.6 Test Case 4 32

Table 7.7 Test Case 5 32

# ABBREVIATIONS

ANN Artificial Neural Network

ASCII American Standard Code for Information Exchange

CCA Connected Component Analysis

HP Horizontal Projection

HPP Horizontal Projection Profile

LDC Least Distance components

OCR Optical Character Recognition

VB Vertical Bar

VP Vertical Projection

VPP Vertical Projection Profile

# 1.0 INTRODUCTION

# 1.1 PROJECT DETAILS

This project work tries to implement the Segmentation and Alignment approaches applicable for Gujarati Optical Character Recognition (OCR). In Segmentation portion we perform Line, Word, and Zone Segmentation. Before the Segmentation, some Pre-processing steps are involved of course. Zone Identification is also performed on word level.

## 1.2 PURPOSE

Enormous amount of information is available in the world, in the form of printed text. Operations such as searching, transposing and processing on required information in a printed form are difficult, time consuming and costly. To transform the information to electronic form either one has to enter it manually or scan the pages and store in the form of images. Both the methods are not good as entering manually such a large amount of information is impractical and error prone and storing in the form of images requires lot of disk space, and also transformation, searching are very difficult tasks. This led to need of OCR development.

Optical Character Recognition (often abbreviated as OCR), in the context of computers, means converting a digital image of a machine printed document in to a computer file which can be stored, edited, indexed and searched more efficiently than the image of the document itself. Although there has been a significant number of improvements in languages such as English, but recognition of Gujarati scripts is still in its preliminary level because Lake of Resources.

Gujarati is one of the official languages of India, which belongs to the group of Indo-Aryan languages and is written in Gujarati script. Gujarati script is used to write the Gujarati language spoken by about 50 million people in the western part of India [18]. However, it is a surprising fact that, very little of this wealth of Gujarati literature is available in an electronic form which can allow searching and indexing. For recreating this wealth of existing literary work in machine readable form, it is impractical to think of getting it typed again as it is time consuming and prone to errors. The solution to this problem lies in building robust Optical Character Recognition (OCR) Systems for Gujarati script. However, complexity of Gujarati script poses several challenges in building such a system.

The segmentation phase will not be able to identify the Upper and lower zone component separately as they are conjunct at Middle zone component.

## 1.3 SCOPE

The scope of our work is only for the Machine printed Gujarati Text Documents not for Handwritten Gujarati Text. It’s doesn't work on classifier, complex newspaper document. The text documents taken under experiments are assumed to be skew corrected and graphics free.

## 1.4 OBJECTIVE

The objective of our work is Zone Identification, Segmentation, symbols collection and alignment of symbols on Gujarati text. Alignment is performed after symbols are segmented.

## 1.5 TECHNOLOGY AND LITERATURE OVERVIEW

### 1.5.1 Optical Character Recognition (OCR)

Optical Character Recognition (OCR) is the text recognition system that allows hard copies of written or printed text to be rendered into editable, soft copy versions. It is the translation of optically scanned bitmaps of printed or written text into digitally editable data files. It is widely used as a form of data entry from some sort of original paper data source, whether documents, sales receipts, mail, or any number of printed records. It is a common method of digitizing printed texts so that they can be electronically searched, stored more compactly, displayed on-line, and used in machine processes such as machine translation, text-to-speech and text mining.

OCRs are of two types,

1. OCRs for recognizing printed characters
2. OCRs for recognizing hand-written text.

OCRs meant for printed text recognition are generally more accurate and reliable because the characters belong to standard font files and it is relatively easier to match images with the ones present in the existing library. As far as hand writing recognition is concerned the vast variety of human writing styles and customs make the recognition task more challenging. An OCR for hand writing is still in the research and development stage to have more result accuracy.

Optical Character Recognition (OCR) is one of the most common and useful applications of machine vision, which is a sub-class of artificial intelligence, and has long been a topic of research, recently gaining even more popularity with the development of

Prototype digital libraries which imply the electronic rendering of paper or film based documents through an imaging process.

## 1.5.2 Components of an OCR System

Various components of an OCR system. Can be broadly divided into four groups as shown in fig 1:

* 1. Image Acquisition and Pre-processing
  2. Segmentation
  3. Recognition
  4. Document Reconstruction / Layout Restoration

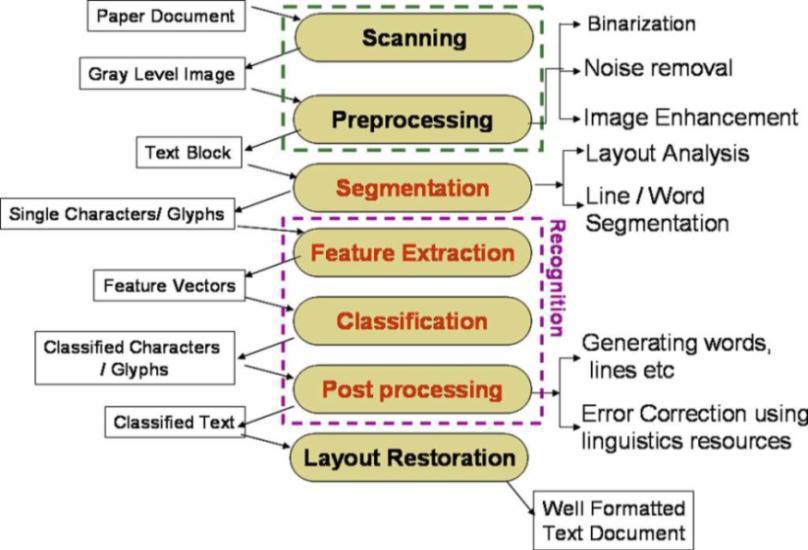


Figure 1: Components of OCR System

### 1.5.2.1 Binarization

Binarization or Thresholding is the operation performed upon gray-scale document images which contain text or graphics. The objective of binarization is to

automatically choose a threshold that separates the foreground and background information. The output of the thresholding operation is a binary image whose one state will indicate the foreground objects, that is, printed text, a legend, a target, defective part of a material etc. while the complementary state will correspond to the background. Figure 2 indicates the result of a binarization method applied on a document.

There are certain categories available for Binarization methods,

1. Histogram shape-based methods
2. Clustering-based methods
3. Entropy-based methods
4. Object attribute-based methods
5. The spatial methods
6. Local methods

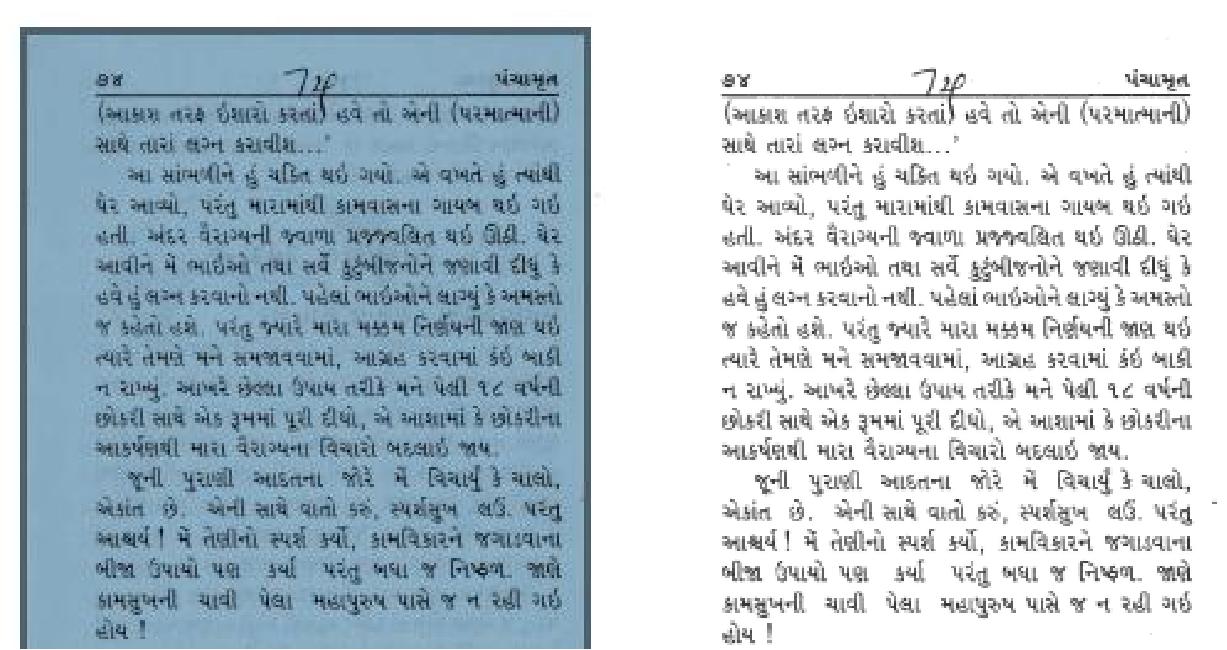


Figure 2: (a) Input Document for Binarization (b) Output of Binarization

#### 1.5.2.2 Noise **Removal**

### 1.5.2.2 Noise Removal

#### There can be different type of noises that can deteriorate quality of the image. Digital capture of images can introduce noise from scanning devices and transmission media. Noise Removal operations have been used to eliminate the artifacts introduced during image capture. Figure 3 & 4 shows the results of Noise removal Process.

There are many noise removal and smoothing methods.

* Median filters
* Gaussian filters
* Morphological filters
* Low pass filters

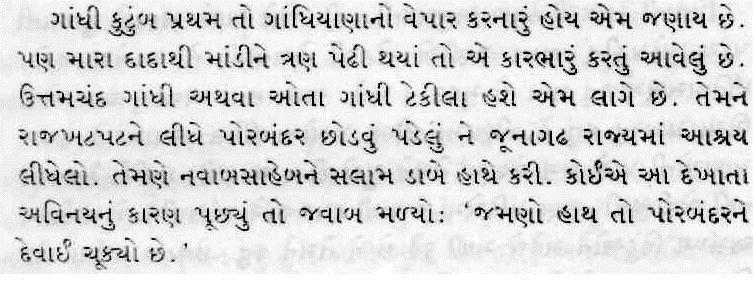


Figure 3: Input document for Noise Removal

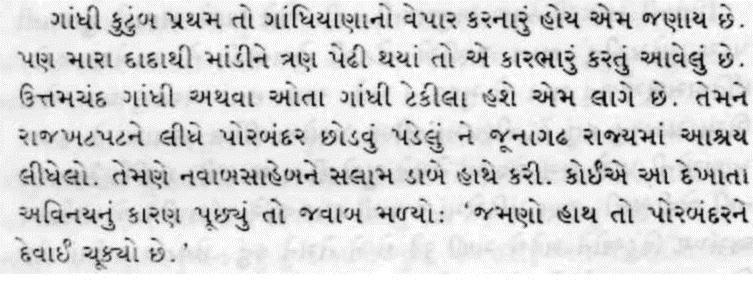


Figure 4: Output of Noise Removal

### 1.5.2.3 Skew Detection and Correction

When a document is scanned, due to the improper scanning, scanned document may contain certain degree of skew present into it. .Skew of the scanned

document image is a problem that reduces accuracy of any OCR system. To get better recognition score skewed images should be de-skewed first.

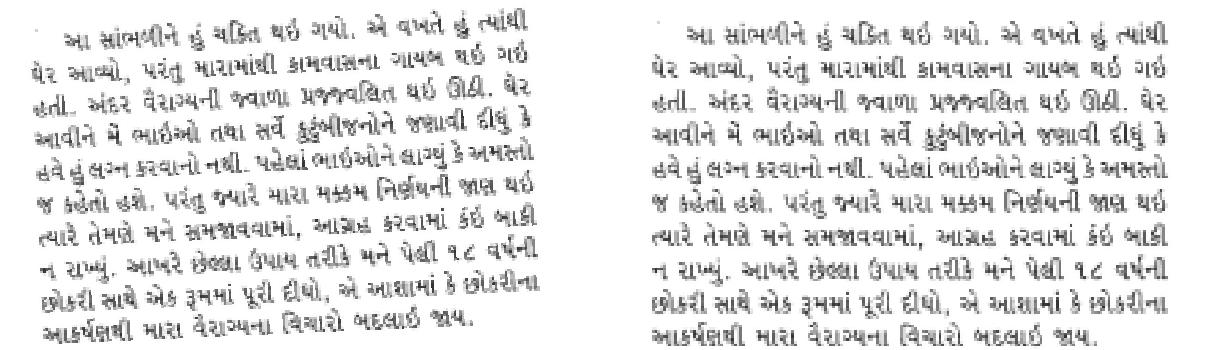


Figure 5 : (a) Input Document for Skew Correction (b) Output of Skew Correction

#### 1.5.2.4 Thinning

Thinning is a process that deletes the unwanted pixels and transforms the image pattern one pixel thick .i.e. the thinning operation is typically applied repeatedly, leaving only pixel-wide linear representations of the image objects.

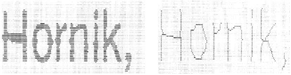


Figure 6: (a) Input for thinning (b) After Thinning

### 1.5.2.5 Segmentation

Segmentation process subdivides an image into its constituent regions or objects. Segmentation phase will separate the lines from the text block then segment various words present in single text line and then separate each character of all the available words. The objective of segmentation phase is to identify each glyph separately so that they can 4 indicates the line

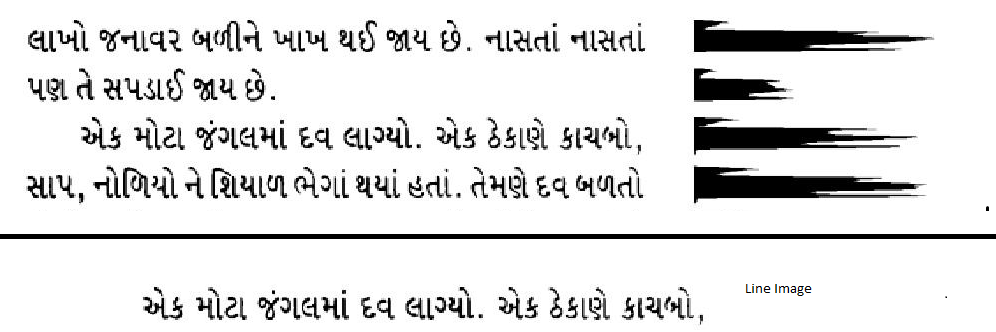


Figure 7: Line segmentation using HP

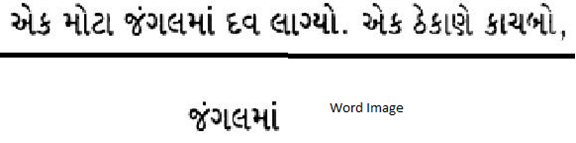


Figure 8: Word segmentation using VP

### 1.5.2.6 Recognition

Segmented glyphs are recognized during this phase. The precision of Recognition phase depends upon how well each text component is segmented during segmentation phase. The decision about the unit of recognition will drive the segmentation process, the depth of the segmentation to be precise. Recognition can be divided into three subtasks viz. Feature Extraction, Classification and Post Processing. Based on the selected features, each glyph is classified during this phase. Post-processing is a process that identifies and rectifies the errors in the output of the recognition phase.

### 1.5.2.7 Document Reconstruction / Layout Restoration

The output of the previous task is text blocks which corresponds to their images extracted during the layout analysis. During this step those text blocks are arranged appropriately to restore the layout of the printed document in the soft copy, using the information acquired at the layout analysis subtask of the first step. At the end of this

Process the OCR system is expected to produce editable copy of the document image scanned at the first step keeping the formatting intact.

### 1.5.3 Tools and Technology

Our entire work is implemented in Matlab R2012b. We have used NNTool- inbuilt toolbox for implementation of MLP based method. NNTool provides Neural Network Toolbox graphical user interface. NNTool opens the Network/Data Manager window which allows us to import, create, use, and export neural networks and data.

# 2.0 PROJECT MANAGEMENT

## 2.1 FEASIBILITY STUDY

### 2.1.1 Technical Feasibility

* System provides the hardware and software requirements at the time of installation.
* So that, it will be feasibility to the system if it have the required technical support.

### 2.1.2 Time Schedule Feasibility

Projects are initiated with a specific deadline. We need to evaluate whether the deadlines are mandatory or desirable. Time is one of the critical factors in the development of any system and hence proper scheduling is very essential for timely completion of project.

### 2.1.3 Implementation Feasibility

Implementing our system is feasible as it just needs the Matlab to be installed on the Computer. Once we are ready with the installation, we can implement our tool quiet easily and sufficiently.

## 2.2 PROJECT PLANNING

## 2.2.1 Project Development Approach and Justification

* For Project Development Iterative Waterfall Model is used.

**Iterative waterfall model**

The Iterative waterfall model approach the problems associated with the waterfall model approach. If any difficulty or problem encounter in any phase may require going back to the previous phase and performing the required modifications and proceeds sequentially .This backtracking allows modifying any corrections or modifications required in the previous phase.

As illustrated in fig, this model divides the cycle into the phases mention below:

1. Feasibility study
2. Requirements analysis and specification
3. Design
4. Coding and unit testing
5. Integration and system Testing
6. Maintenance

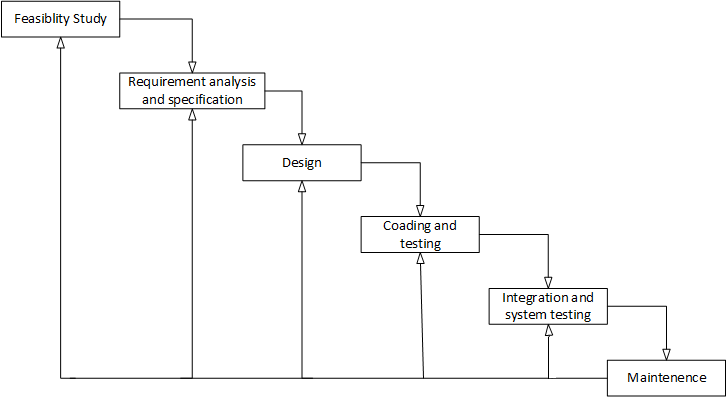


Figure 9: Iterative waterfall Model

**Advantages of using Iterative waterfall model**

* You are provided the changes to see the potential outcomes of every stage and make changes to area of concern if necessary. This is one of the reasons that make iterative model useful.
* Iterative development is more adjustable to changes as it considers each stage like a vital portion of the development.
* The time spent on each successive interval may be less end depending on how this last stage went and what knowledge was gained through adding new functionalities in the development part of all iterations.

**Disadvantages of using Iterative waterfall model**

* When using the iterative model people working on the project can get stuck in a loop. Always finding problems than having to go back and design a fix implement it, than test the system again and finding another problem can mean that the project can run over time and budget.
* Informal request for improvement after each phase may lead to confusion and may also create scope creep, since user feedback following each phase may lead to increased demands. As users see the system develop, they may realize potential of other system capabilities which would enhance their work, this can be advantage as much as in can be disadvantage.

### 2.2.2 Milestones and Deliverables

Timely directions are always required to run a project successfully. Milestones tell the developers how far he has reached and also tell him what things are still left and how to fulfill them. Milestones may be the short report of achievement in project activity that are used by the project manager to check project progress but which are not delivered to the clients. The delivered at the end of some major project phases.

Table 2.1 Milestones and Deliverables

|  |  |  |
| --- | --- | --- |
| MILESTONES | DELIVERABLES | PURPOSE |
| System feasibility study, requirement and analysis | Functional specification | It gives exact understanding of the tool. |
| System Design | Data flow diagram  Structure chart diagram | It gives the logical structure that describes the system |
| Coding and unit testing and corrections if any | Individually tested and functional models. | It gives the required module |
| Integration and Tools testing | The output obtain for the required functionality after implementation | Integration tools is ready |

### 2.2.4 Project scheduling

Table 2.2 Schedule of project

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Task name** | **Start** | **Finish** | **Duration** | **Des-2014** | **Jun-2015** | **Feb-2015** | **Mar-2015** |
|  |  |  |
| 1 | Brief Knowledge about Matlab and OCR | 05/12/2014 | 19/12/2014 | 2week |  | | | |
| 2 | Line Segmentation | 21/12/2014 | 16/01/2015 | 4week |  | | | |
| 3 | Word Segmentation | 18/01/2015 | 31/01/2015 | 2week |  | | | |
| 4 | Zone Segmentation | 01/02/2015 | 14/02/2015 | 2week |  | | | |
| 5 | Symbols collect and alignment of zone | 16/02/2015 | 21/03/2015 | 4week |  | | | |

# 3.0 SYSTEM REQUIREMENT STUDY

## 3.1 USER CHARACTERSTICS

User is familiar with Matlab Built-In Library. And also basic knowledge about OCR. Built-In Library like,

* C callable versions of MATLAB built-in functions
* C routine versions of the MATLAB operators
* Routines that initialize and control how the library operates

## 3.2 HARDWARE AND SOFTWARE REQUIRMENTS

**Software requirement**

 OS: Windows 7 Ultimate

|  |  |  |
| --- | --- | --- |
|  | System type : | 32 bit OS |
|  | Front End : | Matlab R2012b |

**Hardware requirements**

 Processor: Intel Core i3 CPU M430 2.27 GHz

 RAM: 4 GB RAM

# 4.0 SYSTEM ANALYSIS

## 4.1 REQUIREMENT OF SYSTEM

### 4.1.1 User requirement

User requirement include not many things, but important things is user must be aware, how to tool works properly with full availability, reliability, security and safety. The user responsibilities are as follows:

* Should know how to Matlab work.

### 4.2.2 System Requirements (SRS)

For these tools System Requirements means Preprocessing state Requirements and this is dividing into module. Here all images are machine printed documents. Now we see module wise Requirements

**Function Recruitments:**

**R1: Input image (Module 1)**

**R.1.1: I/P image convert into grey scale**

**Pre Requirement:** machine printed Input image.

**Input**: Guajarati script image.

**Output**: Binary image.

**Post Requirement:** Input image perfectly convert into 0’s and 1’s.

**R.1.2: Binary to skew corrected**

**Pre Requirement:** machine printed Input image.

**Input**: Binary or simple image

**Output**: skew corrected image.

**Post Requirement:** Input image perfectly skew corrected.

**R.1.3: Noise Remove**

**Pre Requirement:** Machine printedBinary image.

**Input**: binary image.

**Output**: Noisefree image.

**Post Requirement:** Noise free image.

**R2: Segmentation (Module 2)**

**R.2.1: Line Segmentation**

**Pre Requirement:** Binary, skew corrected, noise free image.

**Input**: Guajarati script image.

**Output**: separate images of line.

**Post Requirement:** wholeimage dividing into lines image.

**R.2.2: Word Segmentation**

**Pre Requirement:** I/P image are single line images.

**Input**: Line images

**Output**: Word images.

**Post Requirement:** All lines dividing into word.

**R.2.3: Zone Segmentation**

**Pre Requirement:** I/P image are single word images.

**Input**: word image.

**Output**: Upper, Middle, Lower zones image.

**Post Requirement:** word image dividing into separate zones.

**R3: Alignment (Module 3)**

**R.3.1: Symbols collect**

**Pre Requirement:** Input image are zones image.

**Input**: Zone image.

**Output**: All symbols collect from zones...

**Post Requirement:** Symbol of zone is separate.

**R.3.2: Alignment of Symbols**

**Pre Requirement:** Symbol of zone is separate

**Input**: Symbols

**Output**: symbols align with zone.

## 4.3 FEATURES OF NEW SYSTEM

Using this system user can perform some preprocessing task like,

* Binarization
* Line Segmentation
* Word segmentation
* Zone Identification.
* Zone segmentation
* Symbol collect
* Alignment of symbols

## 4.4 DATA FLOW DIAGRAM

**LEVEL 0, 1, 2:**

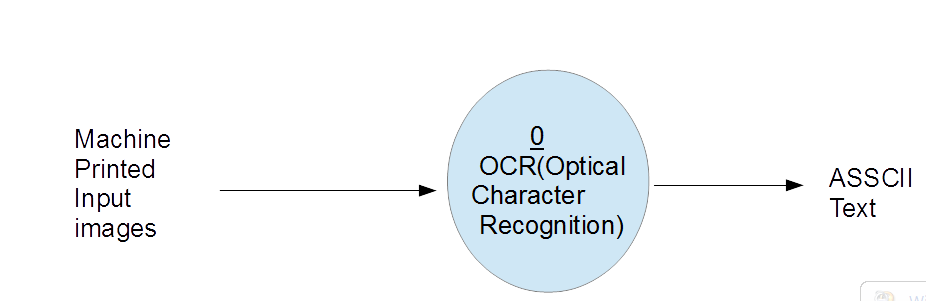


Figure : DFD Level 0

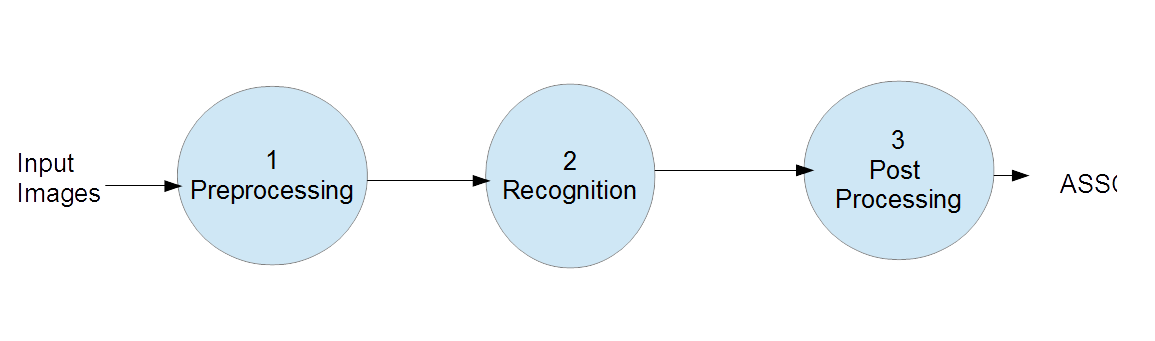


Figure : DFD Level 1

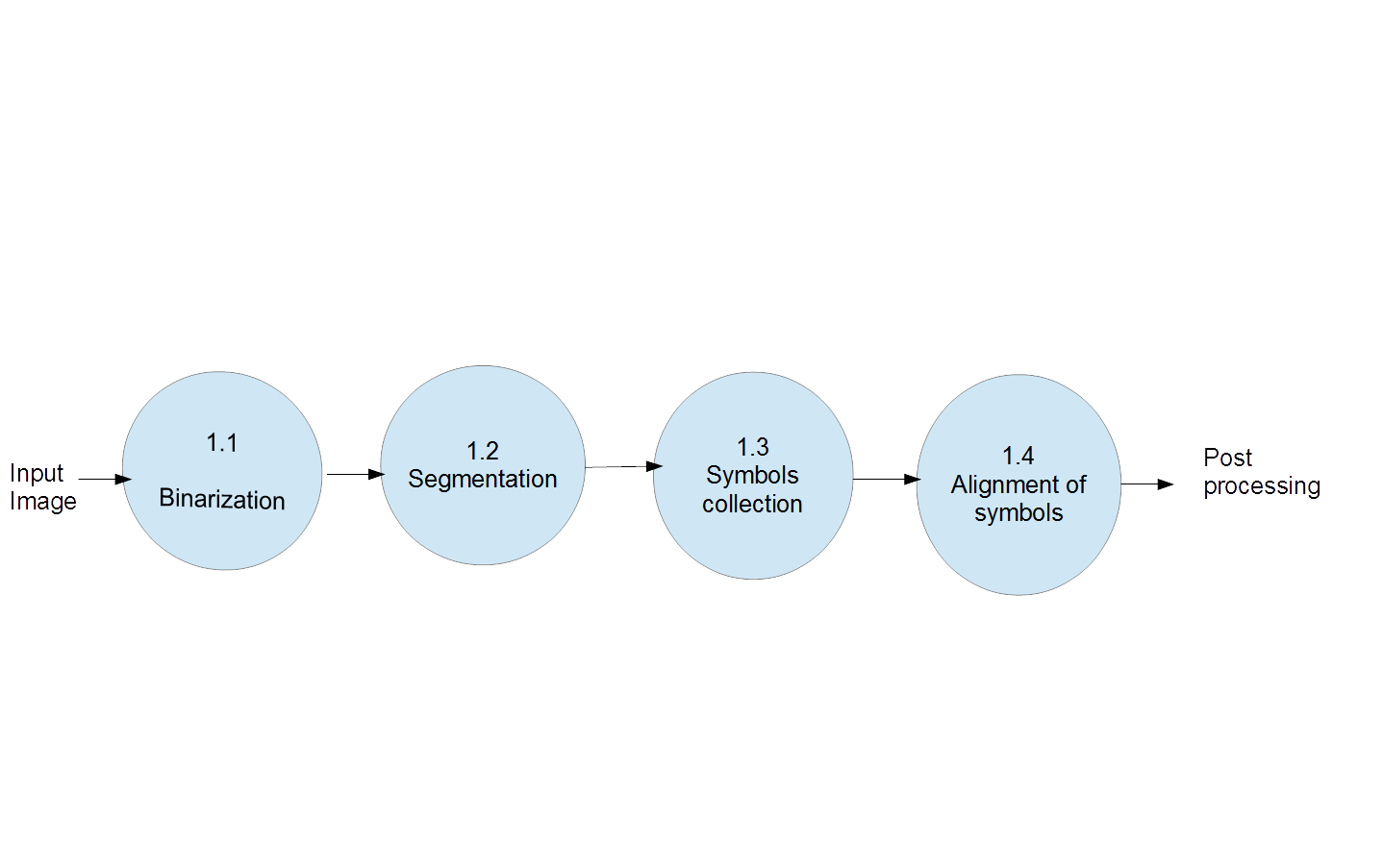


Figure : DFD Level 1

# 5.0 SYSTEM DESIGN

# 5.1 INTERFACE DESIGN

**Structure of Tool:**

1: Documents folder: Machine printed Guajarati script Image.

.

2: Output folders: Bwimage, Lines, Words, Zones, Alignment.

**Flow of Functions:**

* OCRMain
* Binarization
* MY\_Binarization
* LineSegment
* MY\_Linesegment
* WordSegment
* MY\_Wordsegment
* ExtrackBlock
* MarkPoint
* ZoneSegment
* SkipLine
* Skip\_word\_zone
* Alignment1
* Alignment2
* Alignment3
* Alignment4

## 5.2 SYSTEM STRUCTURAL CHART

### 5.2.1 Structural Chart Diagram

Figure : System Structural Chart Diagram

## 5.3 ALGORITHMIC EXPLANATION OF MODULES

Our whole work is mainly divided into 3 modules as indicated in fig 13.

* **Module 1:** During the very first phase of our work, we are acquiring the inputimage of the document and apply certain processing like Binarization, Skew correction, Noise remove.
* **Module 2:** This phase involves line segmentation from input image, then wordsegmentation from the extracted lines. Zone identification method is applied on word level. And now separate the Upper, Middle and Lower zones.
* **Module 3:** During this phase, we perform symbols collection and then Alignment of symbols. Here we applying one new method LDC for alignment.

**Module 1: Input Document Image Acquisition**

The text documents taken under experiments are assumed to be skew corrected and graphics free.

**Function: Binarization**

**Algorithm:**

*Input****:*** Image of Gujarati script

*Output:* Binarized Image.

*Step 1:* Input image is logical or not.

*Step 2:* If yes then directly convert into inverse image. If No then compute one threshold value.

*Step 3:* If values of pixel is more than threshold value so convert into 1’s. If less than convert into 0’s.

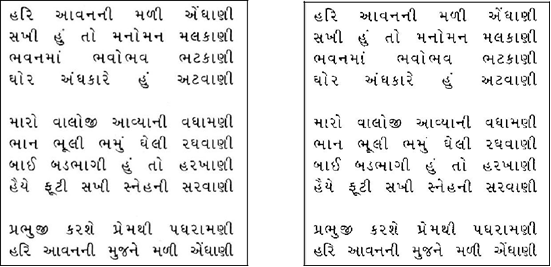
****

Figure : (a) Original Image (b) Binarized image



Figure : Image inversion

**Module 2: Segmentation**

**Function: LineSegmentation**

**Algorithm:**

*Input****:*** Binarized Image.

*Output:* Separate Images of line.

*Step 1:* Perform HP method on Guajarati script image.

*Step 2:* Compute Maxima of each lines. Then find Valley between these maxima.

*Step 3:* Compute the height and width of each line.

*Step 4:* Now, based on the height of the line, separate each line as separate image and stored in “LINE” folder.

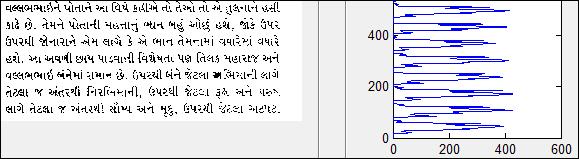




Figure : Line Segmentation

**Function: WordSegmentation**

**Algorithm:**

*Input****:*** Image of lines

*Output:* Separate Image of word.

*Step 1:* Apply VP method on line image.

*Step 2:* Compute Valley between each word and character. Then find the mean (Average) of Valley. This is our threshold value.

*Step 3:*  Now check all Valley between words and character. If Valley is more than threshold value so this is words space.

*Step 4:* Now, compute Height and width of words. And based on this height and width of the word, separate each word as separate images and stored in “WORD” folder.

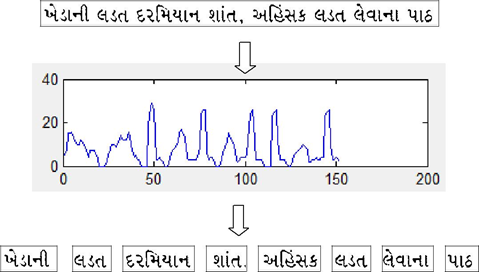
****

Figure : Word Segmentation

**Function: Zone Segmentation**

**Algorithm:**

*Input****:*** Image of words

*Output:* Separate Image of zones.

*Step 1:* Perform labeling operation on text line to identify the Connected Components (CC) separately.

*Step 2:* Compute total number of black pixels in each CC. Then derive the heightand width of each CC.

*Step 3:* Compute Aspect ratio for each CC.Aspect ratio = (height/width) of CC

*Step 4:* Identify the CC that gives highest Aspect ratio.

The vertical bar is identified based on a fact that, the height of Vertical bar is nearly three times of its width. So, the Aspect ratio of Vertical bar will be highest in comparison with all other consonants, modifiers and symbols.

*Step 5:* Now, based on the height of the Vertical bar, separate Upper and Lowerzones and store Upper, Middle and Lower zones as separate images at word level.

*Step 6:* If VB not identify, first consider identified VB cases and then compute frequently occur Height and width.

.

*Step 7:* Using this Height and width, check some conditions.

*Step 8:* Now separate each zones as separate images and stored in “ZONES” folder.

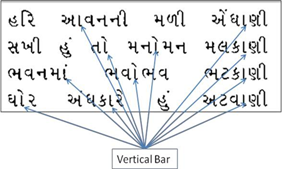


Figure : Vertical bar present in a text region

1. Original Text Line

(b) Upper Zone

* 1. Middle Zone



(d)LowerZone

Figure : Zone Segmentation

**Module 3: Alignment of Symbols**

**Function: Alignment**

**Algorithm:**

*Input****:*** Image of upper, middle, lower zone.

*Output:* Separate Images of upper, lower, middle zone component with indexing.

*Step 1:* Perform labeling operation on each zone to identify the Connected Components (CC) separately.

*Step 2:* Compute total number of black pixels in each CC. Then derive the height*,* width, X, Y of each CC.

*Step 3:* Now, compare distance between upper zone component and middle zone component. Same process for lower zone

*Step 4:* compute two least distance from upper zone components to middle zone component. Same process for lower zone

*Step 5:* Now, check X of upper or lower zone component is between widths of this least distance middle zone component

*Step 6:* If yes then index value of upper zone component and middle zone component are same. Same process for lower zone component.

*Step 7:* This method follow for all upper zone component as well as lower zone component.

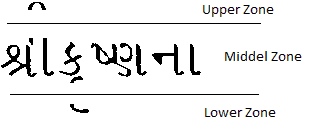


Figure : Original Zone images

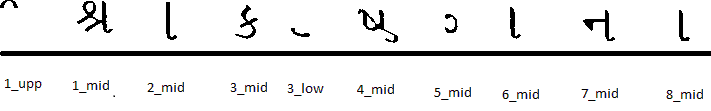


Figure : Alignment of zones

# 6.0 IMPLEMENTATION PLANNING

## 6.1 IMPLEMENTATION ENVIRONMENT

**Hardware Specification**

 Processor: Intel Core i3 CPU M430 2.27 GHz

 RAM: 4 GB RAM

**Software Specification**

 OS: Windows 7 Ultimate

|  |  |  |
| --- | --- | --- |
|  | System type : | 32 bit OS |
|  | Front End : | Matlab R2012b |

## 6.2 PROGRAM SPECIFICATION

We had divided our project onto modules, so that it would be easy when it comes to implementation. This division of modules is based on the task that we accomplished at regular interval of time. Several functions were implemented one after the other at different times and are considered to be under different module as describe below:

* The first module ‘Input Document Image Acquisition**’** is created for the Binarization process. This includes functions like, binarization, My\_binarization.
* The second module ‘Segmentation’ Is created for segmentation process. This includes LineSgmentation, My\_lineSegmentation, WordSegmentation, My\_wordSegmentation, ExtrackBlock, ZoneSegment, Skip\_word\_zone, SkipLine, MarkPoint and ZoneBlock functions.
* The third module ‘Alignment’ is created for Symbols collection and alignment of symbols. This includes Alignment1, Alignment2, Alignment3, and Alignment4 functions.

## 6.3 CODING STANDARDS

The coding standard provides the guideline for coding a module during development. When these standards are strictly followed the code becomes more readable and understandable making the process of debugging easy.

To develop reliable and maintainable tool, one must follow coding standards and best practices. The naming conventions, coding standards and best practices described in this document are those referred by us. There are several standards that exist in the programming industry. None of them are wrong or bad and one may follow any of them. What is more important is, selecting one standard approach and ensuring that the set of standards defined are well adopted. Some Coding standards describe here.

* Specifications for coding loops and other control structures
* Specifications for defining the variables etc. and defining the functions
* Naming Conventions are followed as in Matlab.
* Class name always starts with Capital Letters.
* Sub functions name always start with small letters followed by camel notation.

## 6.4 SAMPLE OF CODE

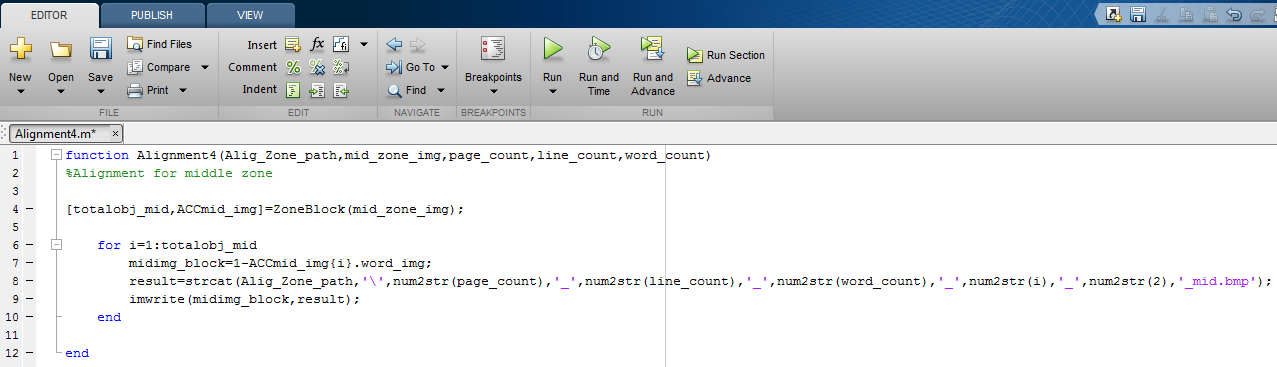


Figure : Sample code for alignment

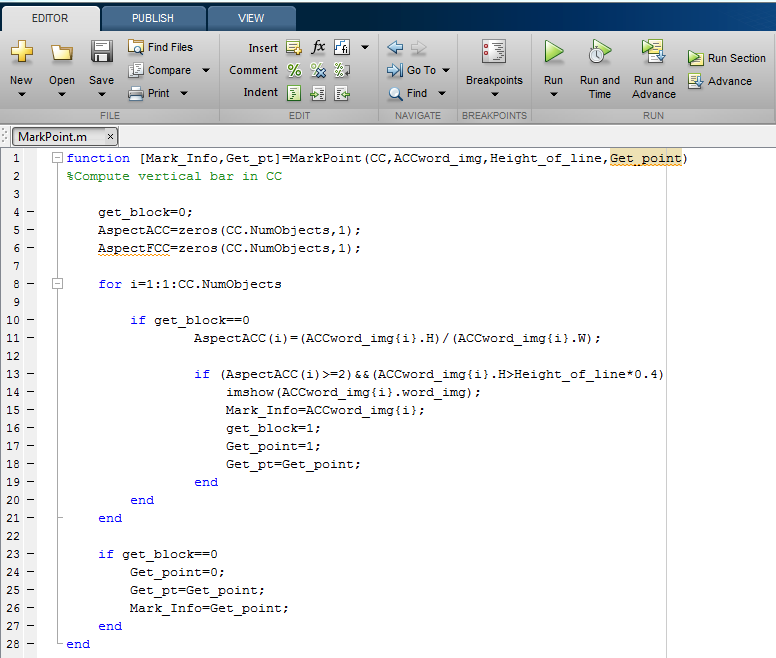
****

Figure : Sample code for Vertical Bar

****

Figure : Sample code for Zone Segmentation

# TESTING

Testing is the process carried out on the tools to detect the difference between its behavior and the desired behavior as stipulated by the requirement specification. Testing is advantageous in several ways Firstly, the defects found help in the process of making the tools reliable. Secondly even if the defects found are not corrected, testing overtime, the record of defect found reveals the most common kinds of defects, which can be used for developing appropriate preventive measures such as training, proper design and reviewing.

## 7.1 TESTING PLAN

The testing technique that is going to be used in the project is white box testing. This testing is based on knowledge of the internal logic of an application’s code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

## 7.2 TESTING STRATEGY

The development process repeats this testing sub-process a number of times for the following phases.

1. Unit Testing
2. Static and Dynamic Analysis
3. Statement Coverage
4. Branch Coverage

**Unit Testing**

The developer carries out unit testing in order to check if the particular module or unit of code is working fine. The unit testing comes at the very basic level as it is carried out as and when the unit of the code is developed or a particular functionality is built.

**Static and Dynamic Analysis**

While static analysis involves going through the code in order to find out any possible defect in the code, dynamic analysis involves executing the code and analyzing the output.

**Statement Coverage**

In this type of testing, the code is executed in such a manner that every statement of the application is executed at least once. It helps in assuring that all the statements are executed without any side effect. Different coverage management tools are used to assess the percentage of the executable elements, which are currently been tested. (These tools are used for both statement as well as branch coverage.)

**Branch Coverage**

No software application can be written in a continuous mode of coding. At some point we need to branch out the code in order to perform a particular functionality. Branch coverage testing helps in validating of all the branches in the code, and helps make sure that no branching leads to abnormal behavior of the application.

## 7.3 TESTING METHODS

**White box testing:**

White-box testing is a method of testing the application at the level of the source code. These test cases are derived through the use of the design techniques mentioned above: control flow testing, data flow testing, branch testing, path testing, statement coverage and decision coverage as well as modified condition/decision coverage. White-box testing is the use of these techniques as guidelines to create an error free environment by examining any fragile code. These White-box testing techniques are the building blocks of white-box testing, whose essence is the careful testing of the application at the source code level to prevent any hidden errors later on. These different techniques exercise every visible path of the source code to minimize errors and create an error-free environment. The whole point of white-box testing is the ability to know which line of the code is being executed and being able to identify what the correct output should be.

## 7.4 TEST CASES:

**Module 1: Input Document Image Acquisition**

**Function: Binarization**

* **Test Case #1:**

Table 7.3 Test Case 1

|  |  |
| --- | --- |
| Test Type | Unit Testing. |
| Action | Binarization |
| Expected Result | Pixel value 0’s or 1’s. |
| Actual Result | Pixel value converts into 0’s and 1’s. |
| No of Sample | 31 Guajarati script images |
| Accurate Result | 31 |
| Less Accurate Result | 0 |
| Accuracy | 100% |

**Module 2: Segmentation**

**Function: LineSegmentation**

* **Test Case #2:**

Table 7.4 Test Case 2

|  |  |
| --- | --- |
| Test Type | Unit Testing. |
| Action | Line Segmentation |
| Expected Result | Separate each line as separate image |
| Actual Result | Mostly all lines are perfectly separate but very few lines are Not separate. |
| No of Sample | 775 Lines |
| Accurate Result | 770 |
| Less Accurate Result | 10 |
| Accuracy | 98.30% |

**Function: WordSegmentation**

* **Test Case #3:**

Table 7.5 Test Case 3

|  |  |
| --- | --- |
| Test Type | Unit Testing. |
| Action | Word Segmentation |
| Expected Result | Separate each Word as separate image |
| Actual Result | All most words are perfectly separate. |
| No of Sample | 5425 Words |
| Accurate Result | 5400 |
| Less Accurate Result | 25 |
| Accuracy | 99.54% |

**Function: Zone Segmentation**

* **Test Case #4:**

Table 7.6 Test Case 4

|  |  |
| --- | --- |
| Test Type | Unit Testing. |
| Action | Zone Segmentation |
| Expected Result | Separate each Zone as a separate image |
| Actual Result | Mostly zones are perfectly separate.  But some zones are not separate. |
| No of Sample | 13562 zones |
| Accurate Result | 1300 |
| Less Accurate Result | 562 |
| Accuracy | 95.55% |

**Module 2: Alignment of Symbols**

**Function: Alignment**

* **Test Case #5:**

Table 7.7 Test Case 5

|  |  |
| --- | --- |
| Test Type | Unit Testing. |
| Action | Alignment of Symbols |
| Expected Result | Separate each Symbols as separate image With proper indexing. |
| Actual Result | Mostly all Symbols are perfectly separate but some symbols are Not separate with proper indexing. |
| No of Sample | 67810 Symbols |
| Accurate Result | 60000 |
| Less Accurate Result | 7810 |
| Accuracy | 89.50% |

# 8.0 USER MANUAL

## 8.1 WHY WE USE OCR:

Many times we want to have an editable copy of the text which we have in the form of a hard copy like a fax or pages from a book or a magazine. The system employs the use of an optical input device usually a digital camera or a scanner which pass the captured images to a recognition system that after passing it through a number of processes convert it to a soft copy like an MS Word document. When we scan a sheet of paper we reformat it from hard copy to a soft copy, which we save as an image. The image can be handled as a whole but its text cannot be manipulated separately. In order to be able to do so, we need to ask the computer to recognize the text as such and to let us manipulate it as if it was a text in a word document. The OCR application does this; it recognizes the characters and makes the text editable and searchable, which is what we need. The technology has also enabled such materials to be stored using much less storage space than the hard copy materials. OCR technology has made a huge impact on the way information is stored, shared and communicated.

In this OCR System for Guajarati Script we implemented preprocessing state. In preprocessing state we done,

* Binarization
* Line segmentation
* Word segmentation
* Zone identification
* Zone segmentation
* Symbols collection
* Alignment of symbols

## 8.2 HOW TO PERFORM ALL TASKS DIRECTLY:

**Structure of Tool:**

**Main folder:** My\_OCR

**Sub folders:**

1:-Documents\_image

2:-Outputs

* Bwimag
* Lines
* Words
* Zones
* Alignment

**Steps:**

**1:** Put machine printed Guajarati script images in ‘Documents\_image’ folder. See the Fig. 25.

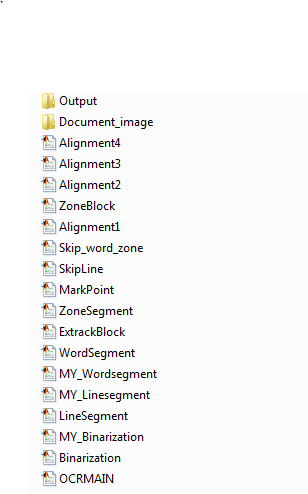
****

Figure : Structure of system

**2:** Open this ‘OCRMAIN.m’ files in matlab. Then select our current directory. After this run only ‘OCRMAIN.m’ file.

Commands:

**>> addpath(genpath(pwd));**

**>> cur\_dir=pwd;**

3: All Functions called by OCRMAIN.m file can be Seen in Fig.26.

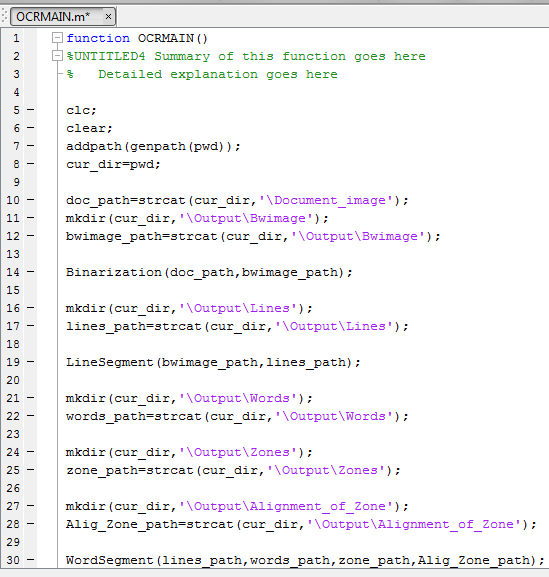


Figure : Flow of functions

**4:** First of all ‘Binarization ()’function will be called by OCRMAIN and ‘bwimage’ folder is created.

Commands**:**

**>> Binarization(doc\_path,bwimage\_path);**

**>> orgimage=im2bw(orgimage);**

**Output:**

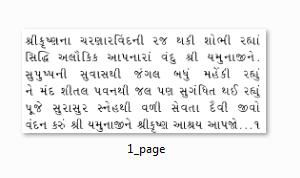


Figure : Output in "Bwimage" folder

**4:** Second ‘LineSegmentation()’ function will called and‘Lines’ folder is created.

Here, Two arguments’bwimage\_path’ and ‘lines**\_**path’ is used.

Command**:**

**>>LineSegment(bwimage\_path,lines\_path);**

**Output:**

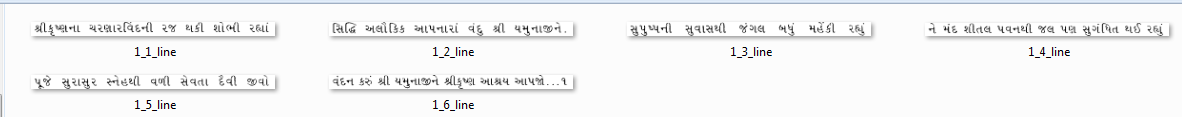


Figure : Output in "Lines" folder

5: Third ‘WordSegmentation’ function will call and ‘Words’ folder is created. Zone Segmentation is performed on word level. Symbols collection and Alignment of symbols

Is performed on zone level.

Commands**:**

**>> WordSegment(lines\_path,words\_path,zone\_path,Alig\_Zone\_path);**

**Output:**



Figure : Output in 'Words' folder

6: Here, Zone identification and Zone Segmentation is performed on word level so Zone related functions will be called by ’WordSegmentation()’.

Commands**:**

**>> ExtrackBlock(line\_img1,Get\_point)**

**>> bwconncomp(line\_img)**

**>> MarkPoint(CC,ACCword\_img,Height\_of\_line,Get\_point)**

**>>ZoneSegment(wordimg,Mark\_point\_Info,zone\_path,Alig\_Zone\_path,page\_count,line\_count,wm)**

**>> SkipLine(Skip\_word\_info,Total\_lines,zone\_path,Alig\_Zone\_path)**

**>>Skip\_word\_zone(wordimg,Mark\_point\_Info,zone\_path,Alig\_Zone\_path,page\_count,line\_count,wm)**

**Outputs**:

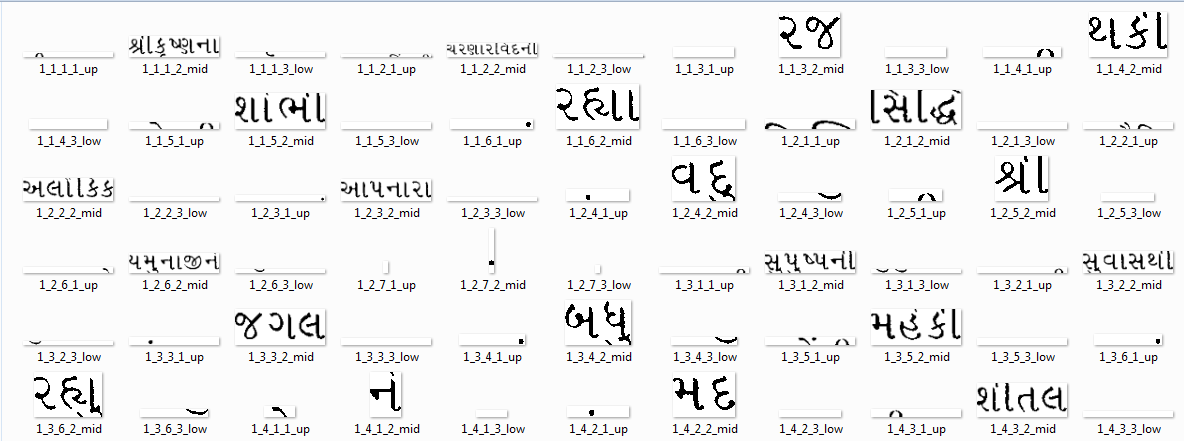


Figure : Output in 'Zones' folder

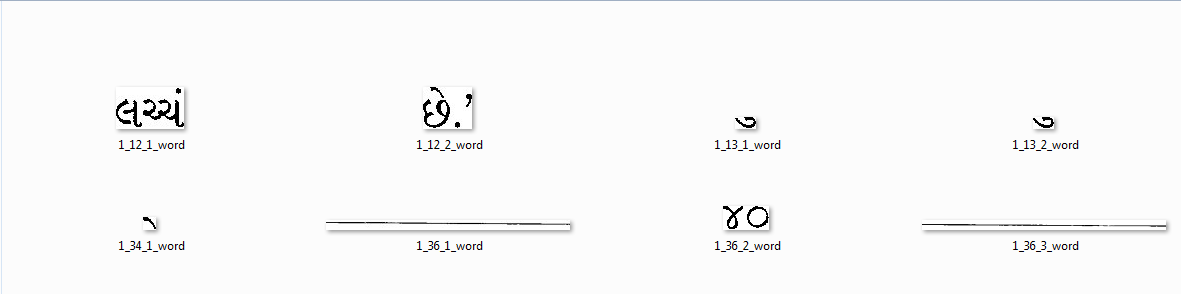


Figure : Output in 'SkipWords' folder

7: Symbols collection and Alignment of symbols is performed on zone level so Symbols related functions will be called by ‘ZoneSegmentation()’.

**Commands:**

**>>Alignment1(Alig\_Zone\_path,upp\_zone\_img,mid\_zone\_img,low\_zone\_img,page\_count,line\_count,word\_count)**

**>>Alignment2(Alig\_Zone\_path,upp\_zone\_img,mid\_zone\_img,page\_count,line\_count,word\_count)**

**>>Alignment3(Alig\_Zone\_path,mid\_zone\_img,low\_zone\_img,page\_count,line\_count,word\_count)**

**>>Alignment4(Alig\_Zone\_path,mid\_zone\_img,page\_count,line\_count,word\_count)**

**Outputs:**



Figure : Output in 'Alignment' folder

## 8.2 INDEXING

Each newly generated Line, Word, Zone and Symbol will be stored as separate image on given location and named in a sequence like- 2\_1\_3\_8\_1\_upp, 2\_1\_3\_8\_2\_mid and so on. Here 2 are indicating image number, 1 are indicating line number. 3 are indicating word number, 8 are indicating character number, and 1 are indicating zone number. During this process, the number of lines, words, zones, symbols in a document are stored in one single folder with indexing.

## 8.4 HOW TO CUSTOMIZE CODE:

**Steps:**

**1:** In our tool we created sub functions like,

**Main task function:** Binarization

**Sub function:** My\_Binarization

**Main task function:** LineSegmentation

**Sub function:** My\_LineSegmentation

2: For customizing our original code we only changed this Sub Function. See Fig. 33.

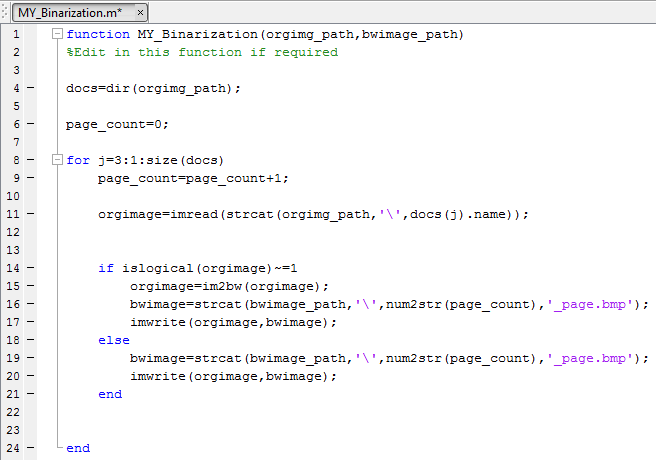


Figure : Sub function of Main task Function

3: All Function arguments are renaming same. We only changed the code of sub function.

4: All outputs are stored using these arguments. Like- bwimage\_path, lines\_path, words\_path, zone\_path and Alig\_Zone\_path.

# 9.0 LIMITATION AND FUTURE ENHANCEMENT

## 9.1 LIMITATION

* We didn't get perfect outputs if input images are noisy.
* The tool is only for the Machine printed Gujarati Text Documents.
* It’s doesn't work on classifier, complex newspaper document.

## 9.2 FUTURE ENHANCEMENT

OCR includes three major steps namely pre-processing, recognition and post processing. This work was for pre-processing, which includes Binarization, Line segmentation, Word segmentation, Zone identification, Zone segmentation, Symbols collection, Alignment of symbols.

For better pre-processing certain operations like- thinning, opening and closing can be used in future. These operations will reduce the effect if shadow characters present in words and it will improve the accuracy of the Alignment. After completion of pre-processing we will move for the next steps of OCR that is Recognition and Post processing.

# 10.0 CONCLUSION AND DISCUSSION

By performing a study of modules we come to a conclusion that preprocessing state of OCR is interesting and very hard to implement. By performing on different modules of preprocessing state we realize that some good programming and problem solving skill is necessary.

Here we worked with Machine printed Guajarati script for Preprocessing state of OCR. We studied various methods for Zone Segmentation for Western and Asian languages like English, Korean, Japanese, Hindi, Tamil, Bangla etc., during this research work. From this study, two methods are implemented. For zone segmentation and alignment of symbols for Gujarati Text, namely VB based method and LDC based method. All these HP, VP, CC, VB, LDC methods are implemented and tested for Module 1, Module 2, Module 3. And all Data set for Segmentation and alignment is also collected. The performance of HP, VP, VB, and LDC based method 98.30%, 99.50%, 95.54%, 89.90% respectively.

# References

1. J. Dholakiya, ―: Mathematical Techniques for Gujarati Document Analysis and Character Recognition, ‖ M. S. University of Baroda (2010).
2. D. D. Huanfeng Ma, ―Adaptive Hindi OCR using Generalized Hausdorff Image Comparison, ‖ 2003.
3. J. H. Bae, K. C. Jung, J. W. Kim, and H. J. Kim, ―Segmentation of touching characters using an MLP,‖ *Pattern Recognition Letters*, vol. 19, no. 8, pp. 701– 709, 1998.
4. S. A. Sattar, ―A Technique for The Design and Implementation of An OCR for Printed Nastalique Text,‖ N.E.D. University of Engineering & Technology (2009).
5. ―http://en.wikipedia.org/wiki/Magnetic\_Ink\_Character\_Recognition.‖ .
6. ―http://en.wikipedia.org/wiki/Optical\_character\_recognition.‖
7. M. H. N. Sinha R. M. K., ―Machine recognition of Devanagari script,‖ *IEEE* *Trans Syst., Man, Cybern.*, vol. vol. SMC-9, pp. 435–441, 1979.
8. U. Pal and B. B. Chaudhuri, ―Indian script character recognition : a survey,‖ vol.

37, pp. 1887–1899, 2004.

1. V. Bansal and R. M. K. Sinha, ―A complete OCR for printed Hindi text in Devanagari script,‖ *Proceedings of Sixth International Conference on Document* *Analysis and Recognition*, no. d, pp. 800–804, 2001.
2. S. Kompalli, S. Nayak, S. Setlur, and V. Govindaraju, ―Challenges in OCR of Dev anagari Documents,‖ pp. 1–5, 2005.
3. B. B. Chaudhuri and U. Pal, ―A Complete Printed Bangla OCR system,‖ vol. 31, no. 5, 1998.
4. K. G. Aparna and A. G. Ramakrishnan, ―A Complete Tamil Optical Character Recognition System,‖ pp. 53–57, 2002.
5. B. Negi, A., Bhagvati, C., and Krishna, ―An OCR System for Telugu,‖ *Proc. 4th* *ICDAR*, pp. 1110–1114, 1997.

[14] S. Antani, L., Agnihotri, ―Gujarati Character Recognition,‖ *Proc. 6th ICDAR*, vol. 418–421.