OS Project 10b

Goal Statement

Implementation of Devanagari OCR on Cell Broadband Engine

Team Details

Name	Roll Number	Email Id	Contact No
Amulya K (Team Leader)	MT2012017	amulya.k@iiitb.org	9880345463
Astha Goyal	MT2012028	astha.goyal@iiitb.org	9740126090
Kodamasimham Pridhvi	MT2012066	pridhvi.kodamasimham@iiitb.org	8123160887
Mehak Dhiman	MT2012080	mehak.dhiman@iiitb.org	9535942619

Contents

1 Introduction

Devanagari is an age old script used for writing many languages like Sanskrit, Hindi, Marathi, Punjabi, Konkani, Thula etc. This has generated a lot of interest in developing new methods of processing documents in Devanagari. Optical Character Recognition (OCR) is an indispensible part of Document Image Analysis.

OCR is basically the process of converting printed or hand-written text into machine-encoded text by recognizing the characters in the document image and converting them to a Unicode format. The output obtained is a parsable document which a machine can interpret and perform some operations or execute certain commands based on the text interpreted. The general applications of this can include helping the blind "read" by converting the text in Unicode format to speech, developing intelligent cars which are able to interpret the signs on the road and navigate on thier own without the intervention of a human being, intelligent robotics, among many others.

The Cell Broadband Engine Architecture (CBEA) and the Cell Broadband Engine are the result of a joint venture by Sony, Toshiba, and IBM (STI). The Cell Broadband Engine was initially intended for application in game consoles and media-rich consumer-electronics devices. Implementation of an OCR on a CBE comes with the advantage of fast processing and recognition, owing to the immense parallel processing ability the CBE brings with it.

2 Problem Statement

The project is aimed at processing the image of a document to produce machine-encoded text (in Unicode format) as the output. This will be implemented on the Cell processor, for printed text and possibly extended for hand-written text. The challenges which make the problem interesting are, the clarity of the document image provided as the input, the lighting conditions under which the image has been taken, possible degrading of the paper quality due to age and skew in the scanned documents. Also, a significant difference between English and Devanagari for instance, lies in

the fact that alphabet in English language does not have anything above or below the line, which is not the case in Devanagari. This project caters to all these challenges.

3 Architecture

Figure 2 shows the architecture of for the implementation of the OCR system on CBE. This section will be in three parts:

- The first part will describe the exact steps involved and algorithms used in the OCR
- The second part will talk about the CBE architecture
- The third part will show **how the OCR system described above will be implemented on CBE**

3.1 Optical Character Recognition System (OCR)

An OCR system involves the following four main stages:

1. **Pre-processing:** Pre-processing involves the process of converting the input image into a binary image, enhancing the contrast of the image by gamma correction, histogram equalization etc. and also skew correction to account for any angular tilt in the scanned document.

   Algorithm: Binarization of an image is done by taking a threshold value and converting all intensity values above the threshold to 1 and all intensity values below the threshold to 0. This produces a 'black and white' image, which is nothing but the binarized image.

2. **Segmentation:** Segmentation is the process of splitting up the pre-processed image, first into individual lines, then into each word in a particular line, and finally into individual characters in each word.

This step is required as these characters need to be recognized in one of the following steps in order to print it in Unicode format.

Algorithm: Horizontal Projection Profile (HPP) and Vertical Projection Profile (VPP) are the two most widely used algorithms for segmentation of a document image. After an image is binarized, each row of the image is summed up to get its HPP. Once the HPP is obtained, there will be a dip in the HPP at places where the image has white spaces between two lines. This will be taken as the reference to segment lines. Each line is then taken and a VPP is taken on that line to get inter-word spaces (and hence a dip in the VPP), and hence word segmentation will be done. For individual characters of a word, a connected component analysis will be done to separate individual characters, where every connected entity (each character in this case) will be labelled as a separate object. Every part of the word below the line will be output as a separate entity and will be sent to a different classifier.

3. **Feature Extraction:** This is the process of extracting specific characteristics of an image (in this case, a character), which are refered to as it's "features". These features will be used to separate one character from the other during the recognition stage.

   Algorithm: In Devanagari, every word will have three parts to it - the top part (which is above the line), the middle part, and the bottom part (which is below the line). After segmentation, each part will be sent to a different classifier after extracting the features. Features used can be a combination of Principal Component Analysis (PCA) [7] and shape-based image invariants [6].

4. **Recognition:** Training samples are processed initially and a database of their features is created and stored. Recognition is the process where each character will be labelled. This is done by comparing the same features of each 'test' image, against the features of every 'trained' image in the database. The class of an image with the maximum match will be reported as the match class.

   Algorithm: Euclidean distance will be used for matching. Support vector machine (SVM) will act as classifier.

3.2 Cell Broadband Engine (CBE)

The IBM Cell Broadband Engine has a multi-core architecture designed for high performance computing and distributed multi-core computing. The architecture features nine microprocessors on single chip. The multi core chip capable of massive floating-point processing is optimized for computing intensive workloads, rich broadband media applications, signal and image processing.

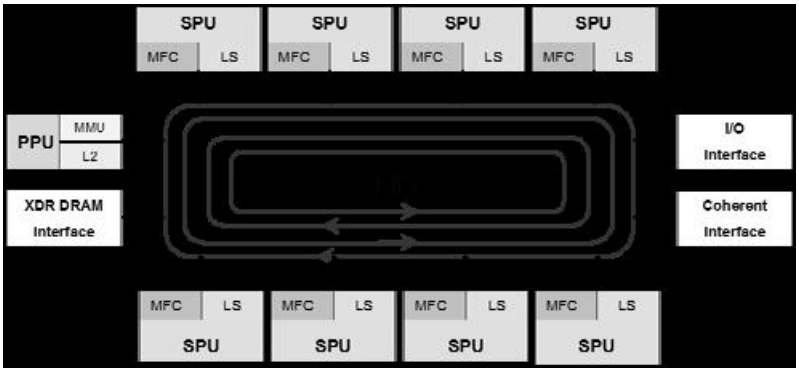

Figure 1: CBE Architecture

Figure 1 shows the architecture of the Cell Broadband Engine. The Cell processor can be split into four components:

- External input and output structures
- Power Processing Element (PPE)
- Synergistic Processing Elements (SPE)
- Element Interconnect Bus (EIB)

The Power Processing Element (PPE) which is the main processor, controls and coordinates all remaining 8 co-processing Synergetic Processing Elements (SPEs). SPEs are designed to accelerate media and

streaming workloads. Area and power efficiency are important enablers for multi-core designs that take advantage of parallelism in applications. It is on these SPEs that the OCR will run and will be controlled by the PPE.

3.3 Overall system Architecture: Implementation of OCR on CBE

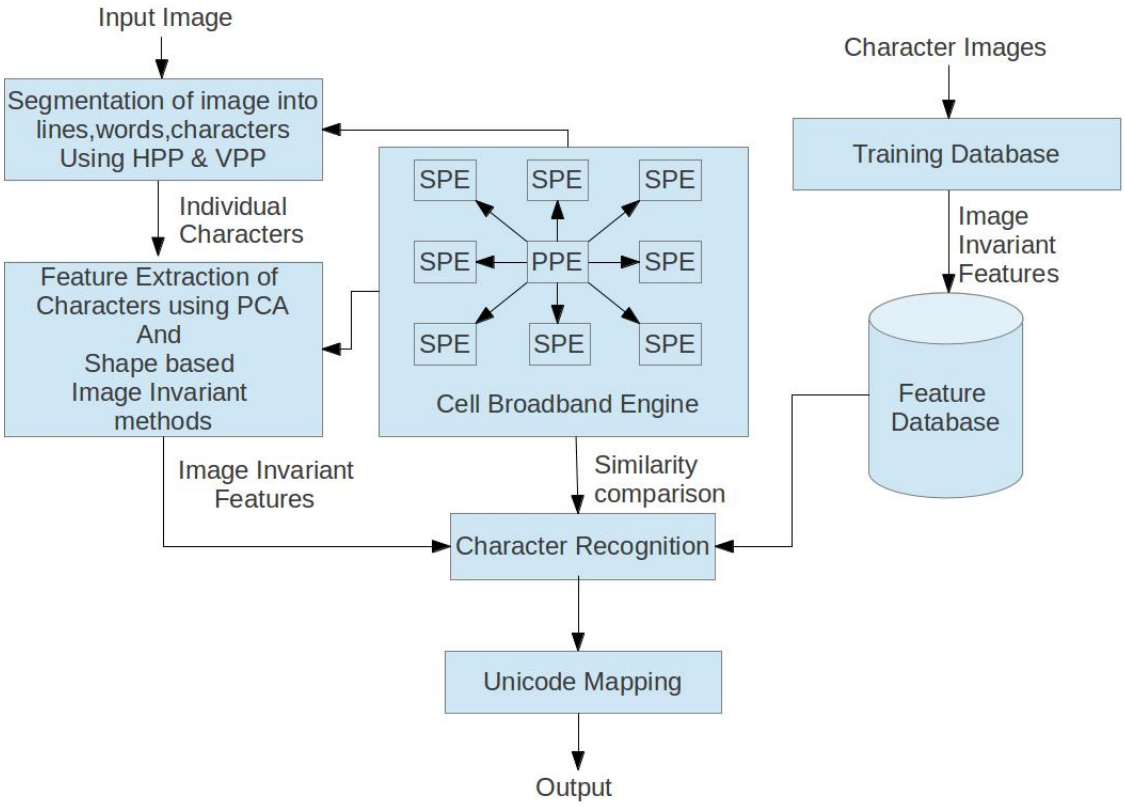

Figure 2: Architecture for implementation of OCR on CBE

Figure 2 shows the architecture for the implementation of the OCR system on CBE.

1. Training and testing are the most important steps in an OCR system. *Training* is a one time process, where character samples of the Devanagari script are taken and their features (obtained through a combination of PCA and image-invariant methods) are extracted. This step can be performed on the CBE or the on the local Ubuntu machine itself, as it is a one time process. The features so obtained are stored in a *feature database*.

2. During *testing*, an input image (document image) is given as the input to the CBE. The PPE instructs one of the SPEs to carry out the segmentation process. Since there are eight SPEs, mutiple images can be segmented during the time usually taken to segment one image. Segmentation is done by the CBE and the segmented characters are output. These are stored on the local Ubuntu machine.

3. The PPE gets the segmented characters and instructs the SPE to perform *feature extraction*. The extracted features (obtained through a combination of PCA and image-invariant methods) will be the output of this step. The PPE directs this to be stored on the local Ubuntu machine. This kind of storing is required as multiple images will be processed at a time.

4. The PPE now fetches the features of an individual image and also the features stored in the database. *Comparison* of the test features with the trained features will happen across multiple SPEs. The match values obtained by all the SPEs is compared by the PPE and the class matched with the least value (in case of Euclidean distance) is reported as the match class.

5. The *output* given by the CBE will be the matched class. This will be used by the local machine to write the corresponding character in Unicode format using the Devanagari fonts present in the system.

4 Gap Analysis

The following points are noteworthy:

- The high performance provided by the CBE can be exploited for large data applications. OCR is one such application, which has a large

dataset of characters and sometimes even words, which need to be pre-processed and stored in the database, after feature extraction. The parallelization provided by CBE will make the time taken taken for this process negligible. OCR has not been implemented on CBE before, and the fact that there is no literature about OCR on CBE is a testimony to the same.

- Devanagari is a language used widely across a large geographical area. Yet, it is yet to receive the amount of importance that really needs to be attached with it. Whatever research that has happened on Devanagari OCR has happened sparsely and it can be observed that there has been no co-ordination among the individual research groups. Hence it is difficult to clearly pin point the exact state-of-the-art in this field. Also, the number of IEEE papers published in this field in the years 2011 and 2012 put together, is only three. This clearly shows that this area needs to be addressed urgently.

- A lot of research on OCR in Devanagari has focused on the Feature Extraction techniques and very few papers have been published on Segmentation techniques. Literature survey has shown that most of the OCR implementations suffer due to errors in the segmentation stage. This usually happens because of improper segmentation due to characters from one line getting joined with the characters on the line below, and so on. This area needs to be addressed in this project.

5 Literature Survey

Literature in OCR can be classified into two - that in segmentation and that in recognition. The most widely used method for segmentation is that of horizontal and vertical projection profiles[2] of an image. Kompalli et. al. [3] have proposed a recognition driven framework along with stochastic models, where a graph representation is used to segment characters. This paper also talks about the usage of linguistics and language models. As for recognition, Neural Networks (NN) [4] [5] have been most widely used. Singh et. al. [4] have proposed an OCR system based on artificial NN, where the features include mean distance, histogram of projection based on spatial position of pixels and histogram of projection based on pixel value. [6] talks about topographic feature extraction, where the topographic features are, closed region, convexity of strokes and straight line strokes. These features

are represented as a shape based graph which acts as an invariant feature set for discriminating very similar type characters efficiently. [7] proposes a bilingual recognizer based on PCA, followed by support vector classification for a Hindi-Telugu OCR. A good survey paper can be found in [8].

6 Development Plan

Phase 1:
a) Installation of CBE SDK and Simulator
b) Generation of data for training the OCR engine. The data for the training phase will be a set of Devanagari characters. For each character, ten images will be generated by us. Seven of these are used for training and three for testing.

Phase 2:
a) Implementation of Segmentation based on HPP and VPP, on Matlab
b) Implementation of Segmentation based on HPP and VPP, on CBE, in C

Phase 3:
a) Training and implementation of Feature Extraction based on a combination of PCA and shape-based image invariant methods, on Matlab
b) Training and implementation of Feature Extraction based on a combination of PCA and shape-based image invariant methods, on CBE, in C

Phase 4:
a) Recognition of characters using Euclidean distance or a classifier like SVM
b) Conversion of the recognized characters to Unicode

Phase 5:
a) Testing of character recognition accuracy
b) Testing of the entire system. To test the entire system, images from the pages of a Sanskrit book will be used. The expected accuracy of the system is about 90%

7 Timeline

Table 1: Timeline

Milestone	Estimated Completion	Date
Phase 1	Initial Project Goal documentation draft and choosing team leader	18-Jan-2013
Phase 2	Final draft of goal statements, CVS/SVN/Git version control repositories set up for project	28-Jan-2013
Phase 3	Installation of SDK and Simulator for CBE and creation of character database	29-Jan-2013
Phase 4	Implementation and Verification of the Segmentation Algorithm based on HPP and VPP, in Matlab	31-Jan-2013
Phase 5	First brief presentations of project architecture and plan by team	5-Feb-2013
Phase 6	Implementation of the Segmentation Algorithm based on HPP and VPP on CBE, in C	12-Feb-2013
Phase 7	Verification of the Segmentation Algorithm based on HPP and VPP on CBE	15-Feb-2013
Phase 8	Mid-term review of project and alpha version release, with presentations by each team member of individual contributions	04-March-2013
Phase 9	Implementation of Feature Extraction on segmented characters, based on a combination of PCA and shape-based image invariant methods on both Matlab and CBE	15-March-2013
Phase 10	Implementation of Recognition of characters on both Matlab and CBE, and conversion of the characters to Unicode format	30-March-2013
Phase 11	Testing and Review, Release of Beta version and first drafts of final docu- ments due	02-Apr-2013
Phase 12	Final Release, Submission of final documents and reports.	18-Apr-2013

8 References

[1] Michael Kistler, Michael Perrone, Fabrizio Petrini, "CELL Multiprocessor Communication Network: Built for Speed", *IEEE Micro*, May-June 2006.

[2] Manoj Kumar Shukla, Dr. Haider Banka, "An Efficient Segmentation Scheme for the Recognition of Printed Devanagari Script", *IJCST*, Vol. 2, Issue 4, Oct.- Dec. 2011.

[3] Suryaprakash Kompalli, Srirangaraj Setlur, Venu Govindaraju, "Devanagari OCR using a recognition driven segmentation framework and stochastic language models", *IJDAR (2009)*, Vol. 12, pp. 123-138, DOI 10.1007/s10032-009-0086-8.

[4] Raghuraj Singh1, C. S. Yadav, Prabhat Verma, Vibhash Yadav, "Optical Character Recognition (OCR) for Printed Devnagari Script Using Artificial Neural Network", *International Journal of Computer Science Communication*, Vol. 1, No. 1, January-June 2010, pp. 91-95.

[5] Anilkumar N. Holambe, Sushilkumar N. Holambe, Dr.Ravinder.C.Thool, "Comparative Study of Devanagari Handwritten and printed Character Numerals Recognition using Nearest-Neighbor Classifiers", *IEEE 2010*, 978-1-4244-5540-9.

[6] Soumen Bag, Gaurav Harit, "Topographic Feature Extraction For Bengali and Hindi Character Images", *Signal Image Processing : An International Journal (SIPIJ)*, Vol.2, No.2, June 2011

[7] C. V. Jawahar, M. N. S. S. K. Pavan Kumar, S. S. Ravi Kiran, "A Bilingual OCR for Hindi-Telugu Documents and its Applications", International Institute of Information Technology, Hyderabad, 2006.

[8] R. Jayadevan, Satish R. Kolhe, Pradeep M. Patil, and Umapada Pal, "Offline Recognition of Devanagari Script: A Survey", *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, Vol. 41, No. 6, November 2011.

[9] U. Pal, T. Wakabayashi, and F. Kimura, "Comparative study of Devanagari handwritten character recognition using different features and classifiers", in *Proc. 10th Conf. Document Anal. Recognit.*, 2009, pp. 1111-1115.

[10] S. Arora, D. Bhattacharjee, M. Nasipuri, D. K. Basu, M. Kundu, and L. Malik, "Study of different features on handwritten Devnagari character", in *Proc. 2nd Emerging Trends Eng. Technol.*, 2009, pp. 929-933.