A Document Image Analysis System on Parallel Processors

Shamik Sural, CMC Ltd. 28 Camac Street, Calcutta 700 016, India.
P.K.Das, Dept. of CSE. Jadavpur University, Calcutta 700 032, India.

Abstract

This paper presents a document image processing system implemented on a set of parallel processors. A pre-processing stage is first used to correct skew from scanned document images. The corrected image is segmented and labelled in a two-step Minimum Containing Rectangle (MCR) detection stage. Text Block Filtering (TBF) is then done heuristically and the filtered blocks are submitted to a Multi-Layer Perceptron (MLP) for recognition of characters. Smoothing of the document image is done during MLP-based character recognition to reduce the pre-processing time. It also reduces the formation of merged characters, a main source of recognition errors in conventional approaches. The MLP identifies the bold words during recognition which are used for automatic indexing of documents. Data is partitioned exploiting the inherent parallelism in a document image data. Communication overhead is small compared to the computation time so that a high degree of parallelization is achieved, reducing the total execution time.

Keywords : Document Image Analysis, Minimum Containing Rectangle, Multilayer Perceptron, Parallel Processing, Transputer

1 : Introduction

In document processing, a paper document is first scanned and binarized to generate a two-tone image. This is followed by a pre-processing stage for noise reduction and skew correction. The image is then segmented and labelled into a number of document blocks [5]. Segmented blocks are further processed to identify semantic relationships among them [7]. Finally, the text blocks are submitted to a character recognition unit for identification of characters and words [6].

Image processing and pattern recognition techniques are used for the analysis and understanding of technical journals, business forms, engineering drawings, postal addresses, musical notations, maps and several other types of documents [3]. For many of these applications, a real time processing is necessary. It is our observation that, like many other image processing and computer vision applications, document analysis and recognition is also inherently parallel in nature. We reduce the processing time significantly by exploiting this parallelism in different stages. The document recognition system has been developed on transputers.

2 : Document processing steps

2.1 : Skew correction

A scanned image is skewed due to mis-alignment of the document with the scanner axis. Skew in an image causes improper segmentation and labelling of document blocks. Hough transform and its variations are mostly used for skew detection [2]. However, Hough transform uses global information of the document image which makes it unsuitable for parallel implementation. We use a modified profiling technique in which horizontal projection profiles (HPP) of all the rows in the document image are determined over a range of search angles. The search angle with the sharpest projection profile is considered to be the angle of skew. The image is then rotated in the opposite direction by the detected angle to correct this skew. A small step size with large search interval makes the skew detection stage highly computation intensive while a large step size over small search interval adversely affects the accuracy. To get an optimum performance, we first keep the step size as well as the search interval large and then adaptively decrease the step size over a shorter search interval for higher accuracy. Figures 1(a) - (d) explain the skew correction steps.

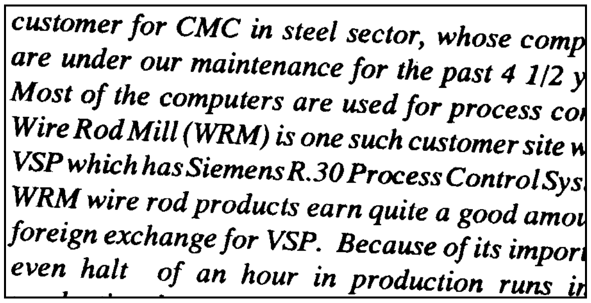

customer for CMC in steel sector, whose comp
are under our maintenance for the past 4 1/2 y
Most of the computers are used for process co
Wire Rod Mill (WRM) is one such customer site w
VSP which has Siemens R.30 Process Control Sys
WRM wire rod products earn quite a good amo
foreign exchange for VSP. Because of its impor
even halt of an hour in production runs i

Figure 1a. A skewed document image.

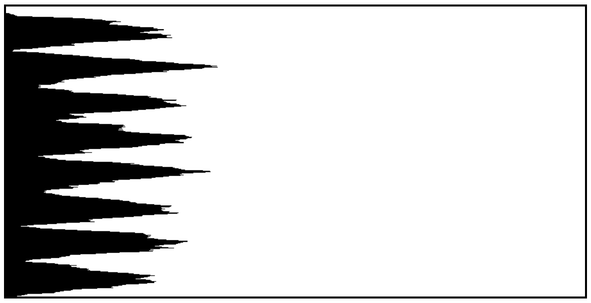


Figure 1b. HPP of the skewed document image.

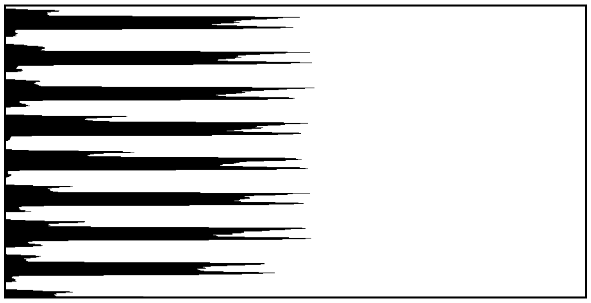


Figure 1c. HPP at the detected skew angle.

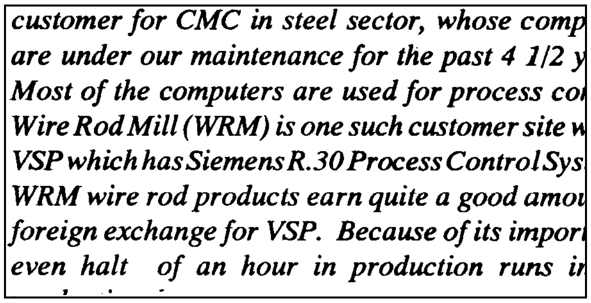

customer for CMC in steel sector, whose comp
are under our maintenance for the past 4 1/2 y
Most of the computers are used for process co
Wire Rod Mill (WRM) is one such customer site w
VSP which has Siemens R.30 Process Control Sys
WRM wire rod products earn quite a good amo
foreign exchange for VSP. Because of its impor
even halt of an hour in production runs i

Figure 1d. Document image after skew correction.

2.2 : MCR detection and text block filtering

A corrected image is segmented and labelled into individual document blocks in this stage. Each such block is first identified by its MCR. An MCR is a rectangle with minimum dimensions that completely encloses a connected region. We propose a two-step algorithm for the identification of block MCRs which can be efficiently implemented on parallel processors.

MCR detection

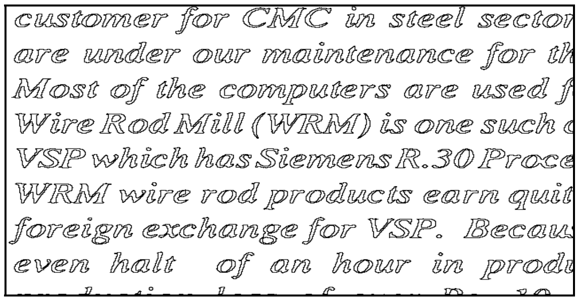
In the first step of MCR detection, the connected region boundaries are determined using the following algorithm.

```
FOR i = 1 TO no_of_rows
  FOR j = 1 TO no_of_columns
    IF pxl[i,j] = 1 AND  pxl[i,j-1] = 0 SET  pxl[i][j] = 0
      ELSE
    IF pxl[i,j] = 0 AND pxl[i][j-1] = 1 SET pxl[i][j-1] = 0
```

Black pixels on each white-black horizontal transition of the image are replaced by white pixels at this stage and a new bitmap is generated. The above algorithm is then repeated for the new bitmap, moving in the vertical direction along each column. Black pixels on each white-black transition, both horizontal and vertical, of the original image are replaced by white pixels and a transformed bitmap is generated. The pixel values of the original and the transformed bitmaps are then logically EXCLUSIVE-ORed to generate a boundary bitmap. The number of pixels in the document image is thus reduced considerably. MCR of each connected region boundary is then determined using a recursive algorithm as follows.

```
FOR i = 1 TO no_of_rows
  FOR j = 1 TO no_of_columns
   IF pxl[i,j] = 1
     SET  MAX_X = MIN_X = i
     SET  MAX_Y = MIN_Y = j
     CALL PROC MCR(i,j)

PROC MCR (cur_x, cur_y)
FOR 8 ngbr_pxl OF  pxl[cur_x, cur_y]
 IF ngbr_pxl = 1
   IF ngbr_pxl_x > MAX_X SET MAX_X = ngbr_pxl_x

   IF ngbr_pxl_x < MIN_X SET MIN_X = ngbr_pxl_x

   IF ngbr_pxl_y > MAX_Y SET MAX_Y = ngbr_pxl_y

   IF ngbr_pxl_y < MIN_Y SET MIN_Y = ngbr_pxl_y

   SET ngbr_pxl = 0
   CALL PROC MCR(ngbr_pxl_x, ngbr_pxl_y)
```

Figures 2(a) and 2(b) show the boundary bitmap and the MCRs of a document image, respectively.



customer for CMC in steel secto
are under our maintenance for th
Most of the computers are used f
Wire Rod Mill (WRM) is one such c
VSP which has Siemens R.30 Proce
WRM wire rod products earn quit
foreign exchange for VSP. Becau
even halt of an hour in produ

Figure 2a. Boundary bitmap of the document

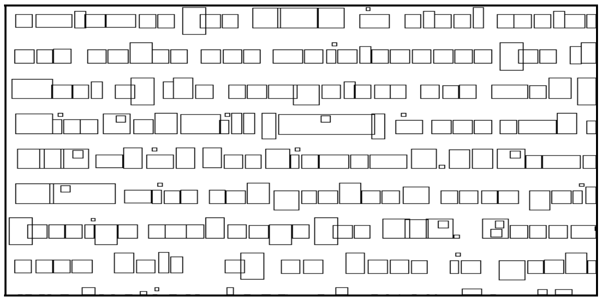


Figure 2b. MCRs detected from the boundary bitmap

Text block filtering

At this stage, the MCRs of all the connected regions in the document image are determined. In the next step, the MCRs are filtered using a number of heuristics to identify the text blocks. Non-text blocks are filtered out in the following cases :

1. Block Height > MAXHTFCTR**.**Average Height
2. Block Width > MAXWDFCTR**.**Average Width
3. Block Height < MINHTFCTR**.**Average Height
4. Block Width < MINWDFCTR**.**Average Width
5. Pixel Density > MAXPXLDEN

Here MAXHTFCTR, MAXWDFCTR, MINHTFCTR, MINWDFCTR and MAXPXLDEN are processing constants determined through a large number of trial runs. Since character recognition is done by a Multi-Layer Perceptron, the sample pattern dimensions are used to determine these values. This step filters out the figures and diagrams which are usually larger than text blocks in a document image. Small spurious document blocks are also eliminated.

2.3 : MLP-based character recognition

The blocks marked as text are next submitted to a Multi-Layer Perceptron for character recognition. MLP and other neural network architecture are discussed in [4]. Since noise affects almost all document images, noise elimination is an essential stage in document image analysis. Smoothing is usually done to remove noise as a pre-processing step along with skew detection. However, any such noise elimination process modifies the character pattern and hence the MLP cannot classify the test input as one of the learnt patterns. We, therefore, train the neural network using the smooth patterns during learning by back propagation. This makes the system more robust to distortion by noise. While classifying, the pattern within each MCR is submitted to the neural network. The same smoothing algorithm is run on the set of pixels in the MCR to form the MLP input. This novel technique of smoothing during recognition enhances the performance of the system in two ways. First, the smoothing algorithm is applied only to the text blocks intended for recognition and not to the entire range of bitmap pixels. Secondly, the usual sequence of smoothing in the pre-processing stage increases the number of black pixels in the document image. The time and resource requirement in the MCR detection stage thereby increases. Formation of merged characters, a main source of recognition error, is also found to be more in this case. The suggested technique of smoothing during recognition reduces these errors significantly.

A document image pattern submitted to the MLP is one of the possible characters in English alphabet. Most of the reported works use as many outputs as the number of patterns [1]. As a result, the number of hidden layer to output layer connections increases, thus increasing the MLP computation time and resource requirements. We, however, use eight units in the output layer to represent the recognized pattern in ASCII coded format and use one extra unit to represent whether the test pattern is in bold or not. Both the bold as well as the regular patterns are used to train the MLP. During recognition, based on the value of the extra output unit (i.e. the bold character indicator), the document is automatically indexed by the bold characters in the image. This step eliminates any human intervention needed for indexing.

During the process of recognition, if an error occurs frequently for a particular image pattern, user interactively identifies the character to the MLP. This is considered an extra pattern and the MLP is re-trained in an off-line mode. While re-learning, the MLP can optionally start from initial random weights or from the existing set of weights. The process of learning during recognition enables the system to identify non-standard fonts also.

3 : Data partitioning and parallelization

In conventional document processing systems, the entire set of bitmap pixels is processed sequentially in different stages. We use a number of processors for parallel processing of document image data. For skew correction in the pre-processing stage, the data set is partitioned and submitted to the parallel processors. Data sets are overlapped to reduce inter-processor communication during processing of data. Initially, the interval and the search step are transmitted along with the data over inter-processor communication channels. Each processor sequentially processes its data set without further parallelization. At the end of computation, the sharpest horizontal projection profile values from all the processors are compared. A first estimate of the skew angle is made based on this data and a shortened interval with a smaller step size is communicated back to the processors. This process is repeated till there is no further improvement in projection profile. The search angle with the sharpest profile is considered to be the skew angle. Each processor then rotates the data set allotted to it in the reverse direction to correct the skew.

Let P_v and P_h be the number of scan lines and the number of pixels along each scan line of the document image, respectively, and N be the number of processors. The bitmap is represented by a two-dimensional array, the number of rows being equal to the number of scan lines P_v. Since most of the binary bitmap files store eight pixel values along a scan line in one byte of data, the number of columns is $P_h/8$. The number of pixels allotted to the nth processor is

$$\begin{aligned} P(n) &= P_h(P_v/N + L) && \text{for } n = 1 \text{ and } N \\ &= P_h(P_v/N + 2L) && \text{for } n = 2, 3, \dots N\text{-}1 \end{aligned}$$

Here $L = P_h \sin\theta$ is the number of overlapping rows required for boundary pixel processing at the edge of each data set; θ being the maximum search angle and the search is made for both positive and negative θ values. The total computation time is reduced by the number of processors and is given by :

$$T_{comp}(N) = (2\theta/\theta_{step})P_hP_v\, t_p/N \qquad (1)$$

where t_p is the average profile calculation time for one pixel and θ_{step} is the search step. The total communication time for data distribution is :

$$T_{dd}(N) = N[t_s + t_b(P_h/8)(P_v/N + L)] + (N\text{-}2)[t_b\,(P_h/8)L] \qquad (2)$$

Here t_s is the set up time and t_b is the byte transfer rate over inter-processor communication channels. The communication time for collecting data as well as the time for comparing results from the N processors is negligible. From eqs. (1) and (2), the total skew detection time is :

$$T_{SD}(N) = (2\theta/\theta_{step})P_h\, P_v\, t_p/N + N(t_s + 2t_bLP_h/8) + t_bP_h(P_v - 2L)/8 \qquad (3)$$

The number of processors for which $T_{SD}(N)$ is minimum, is given by :

$$\begin{aligned} N_{min} &= \sqrt{\frac{(2\theta/\theta_{step})P_hP_v\, t_p}{t_s + 2t_bLP_h/8}} \\ &= \sqrt{\frac{(2\theta/\theta_{step})P_hP_v\, t_p}{t_s + 2t_bP_h^2\sin\theta/8}} \qquad (4) \end{aligned}$$

Eq. (4) shows that a high degree of parallelization is achieved for a wide search interval. Also, for portrait style documents where $P_v > P_h$, a higher number of processors can be efficiently used for processing larger documents. On the other hand, if the set-up time or the byte transfer time is high, parallelization causes communication overhead to increase.

Once skew is eliminated, a document image is processed for MCR detection, Text Block Filtering and MLP-based character recognition. For these steps, data partitioning is done along “clean” rows. A clean row is a scan line with no black pixels. The entire bitmap is first partitioned into data sets with equal number of pixels. Each boundary is checked to see if it is clean. If a boundary is not clean, scan lines are searched both above and below the nominal boundary. If, after a certain maximum number of tries, a clean row is not found for any boundary, the data is added to the data set of the neighboring processor. Otherwise, the data set boundary is shifted to the clean row. All the processors use the same set of weights for the MLPs. There is no further parallelization of the operations of the units in different layers of the MLPs. The worst case computation time for determining clean rows for all the data sets is :

$$T_c(N) = 2Nm_t t_c \qquad (5)$$

where m_t is the maximum number of tries for “clean row” detection and t_c is the average time for checking if a row is “clean”. The computation time for the processor with maximum number of pixels is:

$$T_P(N) = (P/N + 2m_t)\, t_p \qquad (6)$$

Here $t_p = t_{MCR} + t_{TBF} + t_{MLP}$

t_{MCR}, t_{TBF} and t_{MLP} are the average processing time for one pixel for MCR detection, Text Block Filtering and MLP based character recognition, respectively. The total computation time is, therefore,

$$\begin{aligned} T_{comp}(N) &= T_c(N) + T_P(N) \\ &= 2Nm_t t_c + (P/N + 2m_t)t_P \qquad (7) \end{aligned}$$

The communication time required for data distribution is given by :

$$T_{dd}(N) = Nt_s + t_b P/8 \qquad (8)$$

The total processing time is obtained from eqs. (7) and (8) as :

$$T_{DP}(N) = 2Nm_t t_c + (P/N + 2m_t)t_P + Nt_s + t_b P/8 \qquad (9)$$

The number of processors for which $T_{DP}(N)$ is minimum, is given by :

$$\begin{aligned} N_{min} &= \sqrt{\frac{Pt_p}{t_s + 2m_t t_c}} \\ &= \sqrt{\frac{P(t_{MCR} + t_{TBF} + t_{MLP})}{t_s + 2m_t t_c}} \qquad (10) \end{aligned}$$

Eq. (10) shows that a high degree of parallelization is achieved for processing of large documents. It is also seen that the time for detecting "clean" rows for each data set boundary is a processing overhead. Decreasing the maximum number of tries reduces the search time but causes improper load balancing. This value is, therefore, set after a number of trial runs.

4 : Implementation results

We have implemented the parallel document analysis system on T800 transputers using OCCAM. Each character input is represented by a 20X20 matrix which feeds a 400-input MLP with one hidden layer. The number of units in the output layer is 9. Single font recognition of 26 characters requires 16 units in the hidden layer. The speedup variation of the skew detection algorithm with number of processors for different search intervals is shown in figure 3. It is seen that speedup improvement is almost linear. The skew detection step is highly computation intensive and hence there is a significant performance gain through parallelization. Since a larger search interval implies more computation time for the same communication overhead, the speedup gain is also higher. The total speedup variation of the MCR detection, TBF and character recognition steps is shown in the figure 4 for different values of m_t. These results also show a considerable speedup improvement with increase in the number of processors. It is observed that for a lower value of m_t, speedup is higher for the same number of processors. However, for certain values of N, the speedup is less due to improper load balancing as explained in the last section.

5 : Conclusions

We have described a document analysis and recognition system in this paper. A number of modifications have been suggested over the conventional approaches to improve the response time of the system. The algorithms are suitable for implementation on a parallel architecture. Results for a transputer-based implementation of the system have been presented. It is seen that a considerable speedup improvement is achieved through parallel processing due to less communication overhead compared to computation time.

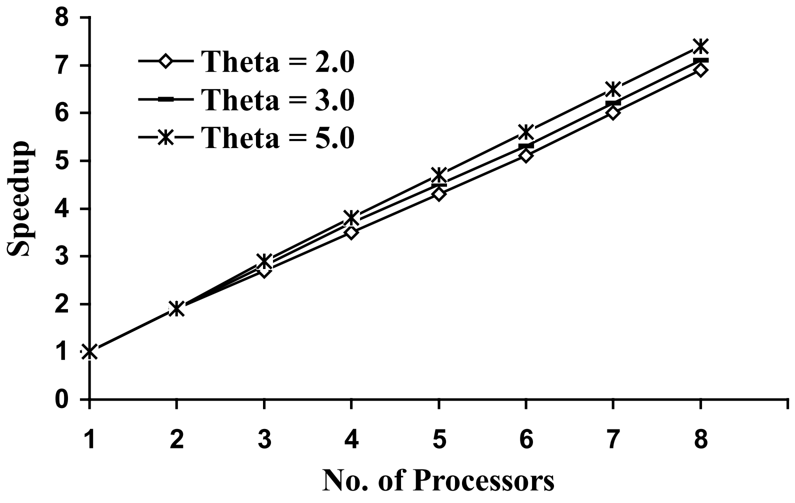


Figure 3. Speedup variation of skew detection algorithm

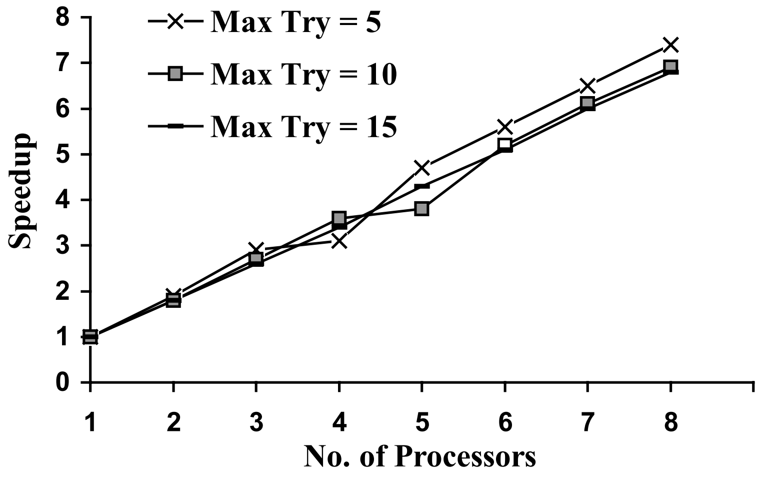


Figure 4. Speedup variation of complete document processing algorithms

References

[1] I. Hadar et al, "High accuracy OCR using neural network with centroid dithering,", *IEEE Transactions on PAMI*, 17 (2), 218-222, 1995.

[2] S. C. Hinds et al, "A document skew detection method using run-length encoding and the Hough transform", *Proc. 10th Int. Conf. Patt. Recogn.*, 464-468, 1990.

[3] R. Kasturi and L O'Gorman, "Document image analysis: A bibliography", *Machine Vision and Applications*, 5, 231-243, 1992.

[4] R.P.Lippmann, "An introduction to computing with neural nets", IEEE ASSP Mgazine, 3-22, 1987.

[5] Y. Lu, "Machine printed character segmentation - an overview", *Patt. Recogn.*, 28, 67-80, 1995.

[6] L. O'Gorman, "Image and document processing techniques for the RightPages electronic library system", *Proc. 11th Int. Conf. Patt. Recogn.*, 260-262, 1992.

[7] Y. Y. Tang and C. Y. Suen, “Document structures : A survey”, *International J. Patt. Recogn. and Artificial Intelligence*, 8, 1081-1111, 1994.