Project	CODICE
Document Type	Technical Report
Protocol Number	TR9703-09
Title	Geometric Layout Analysis Techniques for Document Image Understanding: a Review
Institution	ITC-irst
Address	Via Sommarive 18, I-38050 Povo, Trento, Italy
Authors	Roldano Cattoni, Tarcisio Coianiz, Stefano Messelodi, Carla Maria Modena
Date	January 1998

Geometric Layout Analysis Techniques for Document Image Understanding: a Review

R. Cattoni, T. Coianiz,
S. Messelodi, C. M. Modena
ITC-IRST, Via Sommarive, I-38050 Povo, Trento, Italy

January 1998

Abstract

Document Image Understanding (DIU) is an interesting research area with a large variety of challenging applications. Researchers have worked from decades on this topic, as witnessed by the scientific literature. The main purpose of the present report is to describe the current status of DIU with particular attention to two subprocesses: document skew angle estimation and page decomposition. Several algorithms proposed in the literature are synthetically described. They are included in a novel classification scheme. Some methods proposed for the evaluation of page decomposition algorithms are described. Critical discussions are reported about the current status of the field and about the open problems. Some considerations about the logical layout analysis are also reported.

Keywords: document image processing, document image understanding, automatic zoning, geometric and logical layout, page decomposition, document segmentation and classification, text segmentation, skew angle detection, binarization, document model, performance evaluation.

1 Introduction

In the last decades a big research effort has been spent aiming at the development of automatic text reading systems. Although actual Optical Character Recognition (OCR) systems proved to be powerful enough to meet the requirements of many users [1], room for improvement still exists and further research efforts are required. In fact, developers of such systems mainly

addressed the character recognition task, and considered only documents with plain structure and in black and white. These limitations appear too restrictive with respect to present documents. The requirement of more and more sophisticated tools for the presentation of printed information and the concomitant availability of the needed technology promote a large diffusion of complex documents characterized by creative geometric layouts and the massive use of color.

In its general acceptation *document image understanding* is the process that transforms the informative content of a document from paper into an electronic format outlining its logical content. However, the logical comprehension of arbitrary documents is a very complex task involving high level mental processes which are currently out of the capability of an automatic system. In this context a good target is to perform automatically what is called *zoning*. Originally, the term *zoning* was introduced to define the process that identifies in a document image the text regions and their reading order. In a short time it was clear that a general document reading system would require the segmentation of text regions which play different roles in the document meaning. For example, a column of text is semantically different from a caption, and a footnote is different from a paragraph. Furthermore, a complete understanding of a document requires the interpretation of elements that are not textual, such as drawings, pictures, mathematical equations. These elements have to be isolated and processed by appropriate modules. Nowadays the zoning process is a mix of the so-called *geometric* and *logical layout analysis*. The geometric layout analysis aims at producing a description of the geometric structure of the document. This phase involves several processes: some preprocessing steps and the *page decomposition* step. This last step aims at decomposing the document image into maximal homogeneous regions whose elements belong to different data types (text, graphics, pictures,...). Among the preprocessing modules the most peculiar to the document analysis is the *skew estimation*, *i.e.* estimation of the document orientation angle with respect to the horizontal or vertical direction. The logical layout analysis aims at identifying the different logical roles of the detected regions (titles, paragraphs, captions, headings) and relationships among them.

In this paper we focus on the geometric layout analysis. Our goal is to provide an organic arrangement of the huge amount of material published on this topic. The major difficulty to this purpose is given by the fact that many of the the proposed algorithms address very different specific problems. Furthermore, authors declare performance computed on their own

databases, generally not available, making very difficult a strict comparison. Finally, a general evaluation criterion is not followed.

Although some review papers have been recently published, (see the reviews by Haralick [2], Tang *et al.* [3] and the survey of methods by Jain and Yu in [4]), along with the tutorial text by O'Gorman and Kasturi [5], we believe that an attempt to provide a reasoned systematization of the field can be of great interest. We focus on two processes: skew estimation and page decomposition, and organize the papers according to their objectives and the adopted techniques.[1]

Section 2 gives a general introduction on the geometric layout analysis. Section 3 groups several algorithms of the skew estimation according to the adopted technique. Section 4, which represents the main part of the paper, describe several page decomposition algorithms. The algorithms are organized into a two level subdivision which is reflected by the section structure. Sections 3 and 4 are closed by a summing up of the main characteristics of the different approaches. Automatic evaluation of page decomposition algorithms is an important problem which has been faced only recently. The most prominent proposed approaches are briefly explained in Section 5. On the basis of the reviewed papers, considerations about open problems and future perspectives are reported in Section 6. The logical layout analysis is a process which is complementary to the geometric one. Furthermore, the two processes are not well separated: several ambiguities rising at the geometric level can be resolved only at the logical one. For completeness we choose to report in Section 7 some considerations about standard formats for document definition and a brief survey of several approaches adopted to perform the logical layout analysis. Section 8 concludes the paper.

2 Geometric layout analysis: the issue

The geometric layout analysis aims at producing a hierarchical representation of the document, which embeds its geometric structure, *i.e.* classified *blocks*, each representing a homogeneous region of the page, and their spatial relationships. This structure allows us to describe the document layout at different levels of detail, *e.g.* a body of text can be viewed as a single coherent element, as well as, at a higher detail level, a set of lines. As above

[1] The reviewed papers deal with machine printed "structured" documents such as technical journals, newspapers, business letters. Techniques aiming at text extraction from forms, checks, postal mail pieces, engineering drawings, are not covered in this review.

mentioned in this paper we focus our attention on the skew estimation and page decomposition modules.

Skew estimation is a process which aims at detecting the deviation of the document orientation angle from the horizontal or vertical direction. First document reading systems assumed that documents were printed with a single direction of the text and that the acquisition process did not introduce a relevant skew. The advent of flat bed scanners and the need to process large amounts of documents at high rates, made the above assumption unreliable and the introduction of the skew estimation phase became mandatory. In fact, a little skewing of the page is often introduced during processes such as copying or scanning. Moreover, today documents are ever more free styled and text aligned along different directions is not an uncommon feature.

The purpose of the page decomposition module is to segment the document image into homogeneous blocks of maximum size (this step is called *page segmentation*), and to classify them into a set of predefined data types (this step is called *blocks classification*). Page segmentation takes into consideration only the geometric layout of the page, *e.g.* the spacings among different regions, while blocks classification employs specific knowledge about the data types to be discriminated, *e.g.* features can be devised to distinguish among text, pictures, or drawings.

At this point two observations are appropriate.

1. Until a few years ago, almost all DIU systems worked on binary images. This fact was probably due to limitations of the storing and computing devices, and to the dominant needs in the market for applications dealing with documents characterized by an intrinsically binary nature, such as automatic postal sorting systems and dropout color form reading. Therefore, most page decomposition algorithms have been designed specifically for binary document images. Nowaday, thanks to the development of the necessary technologies, gray level or color input can be taken into consideration, and the requests of applications dealing with not only binary documents are ever more increasing. On such documents, capturing multi spectral images gives obvious advantages to the document readability in imaging applications and is a necessity when dealing with documents containing significant gray level or color pictures which have to be accurately preserved. Although some OCR systems were developed offering gray level input as an option in order to improve their recognition accuracy [1], most OCR commercial systems work, at present, on bilevel inputs [6]. Therefore binarization re-

mains an important module to be applied at least on the textual regions detected in the block segmentation and classification phases. Several algorithms have been specifically designed for the binarization of document images [7, 8, 9, 10, 11, 12]. Other papers provide a comparison of several binarization techniques applied to document images [13, 14, 15].

2. Some sort of interleaving occurs among the processes involved in the geometric layout analysis, like noise reduction, binarization of the whole page or of interesting subregions, skew estimation, and page decomposition (in turn divided into page segmentation and block classification). These modules, although distinct, cannot be applied in a preordered sequence, but their use depends on the particular document category if not on the document itself. For instance, the page segmentation and block classification steps are often said to be separated. Although for some approaches this is true and classification follows segmentation, approaches exist where segmentation follows classification or are interleaved. Even the detection of the document skew angle, often considered a preliminary step, may be applied at different times: directly on the input image, after an eventual binarization, or limitedly to the textual regions, *i.e.* after page decomposition.

3 Skew estimation

Most of the skew estimation techniques can be divided into the following main classes according to the basic approach they adopt: analysis of projection profiles [16, 17, 18, 19, 20], Hough transform [21, 22, 23, 24, 25, 26, 27], connected components clustering [28, 29, 30, 26], and correlation between lines [31, 32, 33]. Other techniques have been proposed which are based on gradient analysis [34, 35], on the analysis of the Fourier spectrum [16], on the use of morphological transforms [36], and on subspace line detection [37].

Notice that, unless otherwise declared, all the cited methods work on binary input images.

3.1 Projection profile analysis.

The underlying assumption of these approaches is to deal with documents in which text is arranged along parallel straight lines. The basic scheme is the computation of a projection profile along each skew angle, the definition of an

objective function, often called *premium*, and the selection of the skew angle which optimizes the objective function. Because of the high computational costs, several variants of this basic method have been proposed. They aim at reducing the amount of data involved in the computation of the profile or at improving the optimum search strategy.

In the method proposed by Postl [16] only points on a coarse grid are used to compute the projection profile and the premium to be maximized corresponds to the sum of squared differences between successive bins in the projection.

In Baird's work [17] another technique is used for selecting the points to be projected: for each connected component the midpoint of the bottom side of the bounding box is projected. The objective function is computed as the sum of the squares of the profile bins. In order to speed up the search of the optimum angle, an iterative method is proposed. In the first iteration the whole angular range is searched with a coarse angular resolution; at each successive iteration the angular range is restricted to a neighborhood of the current best angle and the angular resolution is increased. The author claims [38] the approach to be fast and accurate and to work without modifications on a variety of layouts, including multiple blocks, sparse table and mixed size and typefaces.

In Ciardiello *et al.* work [18], only a selected subregion (one with high density of black pixels per row) of the document image is projected; the function to be maximized is the mean square deviation of the profile.

Ishitani [19] uses a profile which is defined in a quite different way. A sheaf of parallel lines on the image is selected and the bins of the profile store the number of black/white transitions along the lines. The variance of the profile is the objective function to be maximized by varying the slope of the lines. The method is robust to the presence of large non-text regions.

Bagdanov and Kanai [20] propose a technique for selecting points from a JBIG compressed document image. They look for black runs of pixels which have no adjacent black pixel in the lower row: the right most element of such runs is selected. These pixel arrangements give rise to the well-known *pass mode* in CCITT4 compression standard. They can be easily detected by parsing a CCITT4 or JBIG compressed bit stream. White pass codes are detected by simulating a decoding process using a simple two state automaton. The optimization function is the same proposed by Postl [16].

3.2 Techniques involving the Hough transform.

Techniques using the well-known Hough transform [39, 40] have been explored by several authors and are based on the observations that a distinguishing feature for text is the alignment of characters and that text lines of a document are usually parallel each other. Each black pixel (x, y) of the image is mapped into a curve in the parameter space (ρ, θ), the Hough space, using the transform $\rho = x\cos(\theta) + y\sin(\theta)$. Aligned pixels give rise to peaks in the Hough space. The angular resolution of the method depends on the resolution of the θ axis. The complexity is linear with respect to the number of transformed points and the required angular resolution.

Srihari and Govindaraju [21] apply this technique to binary document images, or a subregion thereof, that is known to contain only text and where the entire text block has a single orientation. Each black pixel is mapped in the Hough space and the skew is estimated as the angle in the parameter space that gives the maximum sum of squares of the gradient along the ρ component.

In order to improve the computational efficiency of the method several variants have been proposed that reduce the number of point which are mapped in the Hough space. This can be achieved by selecting a representative subset of the pixels or by restricting the analysis to a subregion of the image.

Hinds *et al.* [22] develop a skew estimation algorithm which reduces the amount of pixels to be processed by the Hough transform. The document image, acquired at 300 dots per inch (from now on *dpi*), is undersampled by a factor of 4 and transformed into a *burst* image. This image is built by replacing each vertical black run with its length placed in the bottom-most pixel of the run. The Hough transform is then applied to all the pixels in the burst image that have value less than 25, aiming at discarding contributes of non textual components. The bin with maximum value in the Hough space determines the skew angle.

Spitz [23] describes a data reduction technique which works directly on CCITT 4 compressed images. Points corresponding to pass codes are extracted with a single pass over the compressed image and are mapped into the Hough space. This technique was extended to JBIG compressed images by Bagdanov and Kanai [20].

Le *et al.* [24] describe an algorithm for the identification of page orientation (portrait or landscape) and of the document skew. Page orientation is detected by dividing the image into small squares, each of which

is classified as containing textual or non textual data according to several heuristics which take into account density and distribution of black pixels. Each textual square is then classified as portrait or landscape by analyzing its horizontal and vertical projection profiles. The classification primarily depends on the presence of peaks alternated with valleys, and secondarily on the comparison of the profiles variances. The number of black pixels in each textual square is used as a classification score. These squares constitutes the first level of a pyramid; each successive layer is constituted by larger and larger squares which are built by merging groups of nine squares of the previous layer. The top of the pyramid represents the whole page. Information about classification is propagated from the base to the top: each square is classified, portrait or landscape, by a class majority criterion among the nine underlying squares taking into account the classification scores. Skew is then estimated on the subregion of the image corresponding to the highest scored square, among the nine of the last layer of the pyramid. Skew is computed by applying the Hough transform on the black pixels of the last row of each connected component.

Another data reduction technique is proposed by Min *et al.* [25]. The document image is divided into vertical stripes whose width Δ depends on the expected distance between lines and the maximum skew angle. For each stripe a vertical signature is built by assigning value 1 to the rows which contain at least one pixel and value 0 otherwise. The central points of the black vertical runs of each signature are mapped into the Hough space.

Pal and Chaudhuri [26] propose two skew estimation techniques. The first one performs a data reduction starting from the bounding boxes of the connected components. The basic idea is to delete components that contribute to noise: punctuations, ascending and descending characters. Small components are filtered away along with components with height above the average height. Two sets of points are then collected: L_1 and L_2 contain, respectively, the leftmost pixel of the uppermost run and the rightmost pixel of the lowermost run of each component. Points in L_1 and L_2 are used in the Hough transformation.

Another skew detection algorithm based on the Hough transform is presented by Yu and Jain [27]. The first step of the algorithm aims at efficiently computing the connected components and their centroides by means of a structure called *block adjacency graph*. The Hough transform is applied to the centroides using two angular resolutions. The coarse resolution permits to approximately estimate the skew angle and to restrict the angular range where the Hough transform has to be computed at the finer resolution. The

reported results on a set of low resolution images show the fastness and accuracy of the technique.

3.3 Nearest neighbor clustering.

The methods of this class aim at exploiting the general assumptions that characters in a line are aligned and close to each other. They are characterized by a bottom up process which starts from a set of objects, connected components or points representative of them, and utilizes their mutual distances and spatial relationships to estimate the document skew.

Hashizume *et al.* [28] present a bottom up technique based on nearest neighbor clustering. For each component they compute the direction of the segment that connects it to its geometrically nearest neighbor. These directions are accumulated in a histogram whose maximum provides the dominant skew angle.

Another method for skew detection based on clustering by nearness is presented by O'Gorman [29]. The author computes a sort of spectrum of the document, called *docstrum*, and uses it as the starting point of the page layout analysis. For each connected component extracted from the document the k nearest neighbor components are computed. The set of couples formed by the component itself and, in turn, its k nearest neighbors is collected. Each couple in the collected set is transformed into the pair (d, ϕ), where d is the Euclidean distance and ϕ is the angle between the centroides of the two components. The resulting set constitutes the docstrum. The choice of k is not critical, but a good choice would require information about some geometric features of the document. The skew is estimated as the mode of the smoothed histogram of the ϕ angles.

The method described by Smith [30] is based on the clustering of the connected components into text lines. A filtering step removes small components and retains those with height between the $20 - th$ and $95 - th$ percentile of the heights distribution. Remaining components are sorted by their column coordinates. The components are grouped into lines as follows: for each component the degree of vertical overlap with existing lines, if any, is computed. It takes into account the horizontal distance between component and line, and the current estimate of the line slope (initially horizontal and updated after each assignment). The current component is assigned to a new line or to an existing one, depending on its degree of vertical overlap. For each cluster the baseline skew is estimated by means of a least median of squares fit. The global page skew is computed as the median slope.

Pal and Chaudhuri [26] present a second approach based on a clustering of two sets of points, L_1 and L_2, extracted from the image (see the first approach described in Section 3.2). The two sets are separately analyzed as follows. An initial straight line, called *ini-line*, is determined by searching three close and aligned points in the set starting from the top of the image. The points of the set are then clustered according to their distance from the *ini-line*. For each cluster, the slope of the segment joining the two furthest points, provides an estimate of the skew. The global skew is the average of these estimates computed over L_1 and L_2.

3.4 Cross correlation.

Under the assumption that deskewed textual regions present an homogeneous horizontal structure, these approaches aim at estimating the document skew by measuring vertical deviations along the image.

Akiyama and Hagita [31] describe a fast approach for skew detection: the document is divided into several vertical strips with the same width. The horizontal projection profiles of the strips are computed along with the shifts that give the best correlation of each projection with the successive one. The skew is estimated as the inverse tangent of the ratio between the average shift and the strip width.

The method described by Yan [32] has the interesting advantage that it can be applied directly to gray level or color images as well as binary images and does not require components extraction. This method is based on the computation of the accumulated correlation function R for many pairs of vertical lines, selected at a fixed distance D. It is defined as: $R(s) = \sum_{x,y} I(x+D, y+s)I(x,y)$ where the summation is over to the whole image I. The estimated skew is the inverse tangent of the ratio between the value of s that maximizes $R(s)$, and D.

Gatos *et al.* [33] propose a skew estimation technique based on a correlation measure between vertical stripes of the image preprocessed by an horizontal run smoothing. The vertical stripes, whose width is experimentally determined, are equally spaced. For each of them a vertical signature is built by assigning value 1 to the rows which contain at least one black pixel, and 0 otherwise. For each pair of signatures (L_i, L_j), a correlation matrix is built: $C_{ij}(r, \lambda) = L_i(r) \cdot L_j(r + \lambda)$, where λ represents the vertical shift. A global correlation matrix is obtained by summing the C_{ij} properly rescaled; the mode of its projection along the λ axis is used to compute the skew angle.

3.5 Other techniques for skew estimation.

Sauvola and Pietikäinen [34] propose an approach for skew detection based on gradient direction analysis, that may be applied to binary or gray-level images. The image is undersampled and convolved with two masks in order to get the gradient map (magnitude and direction). The local dominant directions for each cell of a grid is computed using the gradient information. The histogram of such directions, discarding flat sub-windows, is computed after an angle quantization. The maximum of the resulting histogram estimates the text direction.

A similar technique is used by Sun and Si [35]. The basic assumption is that in a typical document there are more points whose gradient orientations are perpendicular to the text lines. The histogram of the gradient orientation of the gray-level image is computed. The histogram is then smoothed with a median filter in order to reduce undesired effects due to quantization. The mode of the histogram gives an estimate of the skew.

The second method presented by Postl in [16] computes the Fourier transform of the document page and makes use of the power spectrum of the Fourier space to associate a score to each selected angle. Let $S(u, v)$ be the 2-D Fourier transform of the document array, and $W(u, v) = |S(u, v)|^2$ the power spectrum, then the score is computed as the line integral of $W(u, v)$ along the radius vector inclined at angle β with respect to ordinate v. The angle β which maximizes the score is selected as the skew angle.

Chen and Haralick [36] present a text skew estimation algorithm based on opening and closing morphological transforms [41]. The recursive closing transform is computed with a structuring element 2×2 or 2×3, depending on the expected range of the skew angle. The resulting image (that is a sort of anisotropic distance map) is binarized by a global threshold estimated from its histogram. This operation connects characters, words, and other components. Unfortunately, words of different lines may result connected, due to the presence of descendent and ascendent characters. The recursive opening transform is performed on the result, using the same structuring element. The thresholding of the obtained image produces a bitmap in which, ideally, text lines are represented by elongated components, whose direction is estimated. Spurious directions may be present, because of noise, graphics or picture elements. Lines whose directions are near the dominant direction are selected by means of an iterative algorithm that restricts the interesting directions to a neighborhood of the median one. The skew of the document is estimated from the selected directions.

An interesting approach is presented by Aghajan *et al.* [37]. Skew estimation is reformulated as the problem of determining the direction of arrival of planar electromagnetic waves detected by a linear sensor array. At the top of the image columns virtual sensors are placed which measure the signal generated by a set of straight lines in the image plus noise. A spectral analysis of the measurement vector takes place using a subspace-based technique (*TLS-ESPRIT* algorithm) for array processing. The algorithm potentially is capable to detect multiple skew angles although in the experiments a unique skew direction is assumed. The authors claim that the method is robust to the presence of formulas or diagrams and works well both on binary and gray-level images.

3.6 A summary.

Despite many efforts have been spent on the development of skew estimation algorithms, every year new algorithms are proposed in the literature. This is mainly due to the need of: (1) accurate and computationally efficient algorithms; (2) methods that do not make strong assumptions about the class of documents they can deal with.

Computational efficiency may be pursued by reducing the number of points to be processed, by undersampling the image, or by selecting fiducial points. A common solution which is applied to binary input images is to use the connected components as the basis for further computation: an approach which is convenient only if connected components are needed in subsequent steps. Another approach is to consider only subregions of the image under the assumption that only a single skew is present.

A general assumption which, at different extents, underlies skew estimation techniques is that text represents the most relevant part of the document image; performance often decay in the presence of other components like graphics or pictures. Furthermore, the major part of the algorithms assumes to deal with documents with a clearly dominant skew angle, and only a few methods can deal with documents containing multiple skews.

The main advantage of the projection profile methods is that the skew angle accuracy can be controlled by changing the angle step size and the accumulator resolution. Obviously, a finer accuracy requires a longer computational time. *A priori* information about the maximum expected skew angle can be straightforwardly exploited by the method. Analogously to the projection profile analysis, methods based on the Hough transform can control accuracy by changing the resolution of the θ axis. Time efficiency can

be improved by an appropriate reduction of the number of points mapped into the parametric space. Two are the major drawbacks of these methods: they typically require wide memory space and present problems to detect the right peak in the Hough space when applied to documents containing sparse text or data types different from text. Methods based on nearest neighbor clustering require the computation of the connected components which is a costly process. The accuracy depends on the number of connected components and can suffer from the presence of connected or broken characters, non textual items and noise. Assumptions about relations between intercharacter and interline spacings are typically made. Advantages of these methods are: they are the most appropriate to deal with multiple skew angles, and typically they make no restriction on the maximum admitted skew angle. Methods based on correlation typically require an *a priori* estimate of the maximum skew angle and are well suited only for documents which present an homogeneous horizontal structure. Their major advantage is the time efficiency that can be obtained by means of a drastic reduction of processed data without compromising accuracy.

Important features to be considered for a comparison of skew algorithms are: assumptions on the document image (input type, resolution, maximum amount of skew, layout structure), accuracy of the estimated angle, computational time. Unfortunately, a common database of skewed documents is not adopted, making quite difficult a rigorous comparison of features such as accuracy and computational time. Furthermore, some papers don't provide a detailed description of the experimental results. Nevertheless, we have synthesized the main characteristics of the reviewed algorithms, principally following the authors, and collected them in Tables 1 and 2.

4 Page decomposition

We propose a taxonomy of the page decomposition works that distinguishes the algorithms by objectives and techniques. The subdivision scheme is illustrated in Figure 1. Each of the following four categories of page decomposition algorithms derived from a subdivision by objectives, is in its turn divided by considering the main adopted technique.

- *Text segmentation* approaches: algorithms which analyze the document in order to extract and segment text. The textual part is divided into columns, paragraphs, lines, words,... in order to reveal the hierarchical structure of the document. These approaches

method	*reference*	*input type* *resolution*	*skew range* */ accuracy*	*characteristics* *of documents*
Projection profile	Postl [16].1	b/w, g.l. 160 *dpi*	±45° 0.6°	complex documents with a dominant text direction
	Baird [17]	b/w 300 *dpi*	±15° 0.05°	a dominant text direction, a few touching characters, text overwhelms non text
	Ciardiello *et al.* [18]	b/w 300 *dpi*	±45° 0.7°	complex documents, *e.g.* magazines
	Ishitani [19]	b/w 300 *dpi*	±30° 0.12°	complex documents with few text lines
	Bagdanov Kanai [20]	b/w, JBIG 300 *dpi*	±3°	documents with no or a few non textual parts
Hough transform	Srihari Govindaraju [21]	b/w 128 *dpi*	±90° 1°	text only documents
	Hinds *et al.* [22]	b/w 75 *dpi*	±15° 0.5°	complex documents; an estimate of max characters height is needed
	Lee *et al.* [24]	b/w 200 *dpi*	±15° 0.5°	complex documents, *e.g.* medical journals
	Min *et al.* [25]	b/w 300 *dpi*	±20° 0.5°	noisy structured documents with tables; an estimate of interline gaps is needed
	Pal Chaudhuri [26].1	b/w 160 *dpi*	±45° 0.2°	complex documents with one text direction, dominant textual part, Roman script
	Yu Jain [27]	b/w 50-75 *dpi*	±90° 0.1°	complex documents with a dominant text direction

Table 1: Some features of skew estimation algorithms as declared by their authors: input type (binary, gray-level, or color) and resolution of the images used in the experiments, skew angular range where the algorithm works or has been tested and the accuracy in that range, main characteristics of the managed documents.

method	*reference*	*input type* *resolution*	*skew range* */ accuracy*	*characteristics* *of documents*
Nearest neighbor clustering	Hashizume *et al.* [28]	b/w 54-63 *dpi*	±90° 5°	simple documents (*e.g.* envelopes) with line gaps wider than character gaps
	O'Gorman [29]	b/w 300 *dpi*	±90°	text only documents with few touching characters; multiple text directions
	Smith [30]	b/w 300 *dpi*	±15° 0.05°	one text direction
	Pal Chaudhuri [26].2	b/w 160 *dpi*	±45° 0.2°	complex documents with one text direction, dominant textual part, Roman script
Correlation	Akiyama Hagita [31]	b/w 200 *dpi*	±10°	documents with text and graphics, textual part dominant
	Yan [32]	b/w, g.l., color	±45°	one text direction, textual part dominant; an estimate of max skew is needed
	Gatos *et al.* [33]	b/w 96-300 *dpi*	±5° 0.05°	complex documents with one text direction; fast
Gradient analysis	Sauvola Pietikäinen [34]	b/w, g.l.	±20° 1°	complex documents also with few text lines, one text direction
	Sun Si [35]	g.l.	±90°	complex documents; dominant non-slanted textual part
Fourier transform	Postl [16].2	b/w, g.l. 160 *dpi*	±45°	a dominant text direction
Morphology	Chen Haralick [36]	b/w 300 *dpi*	±5° 0.5°	complex documents with a dominant text direction, line gaps wider than character gaps
SLIDE algorithm	Aghajan *et al.* [37]	b/w, g.l. 144 *dpi*	±90° ≈ 0.01°	documents with some non textual parts

Table 2: Features of skew estimation algorithms (2nd part)

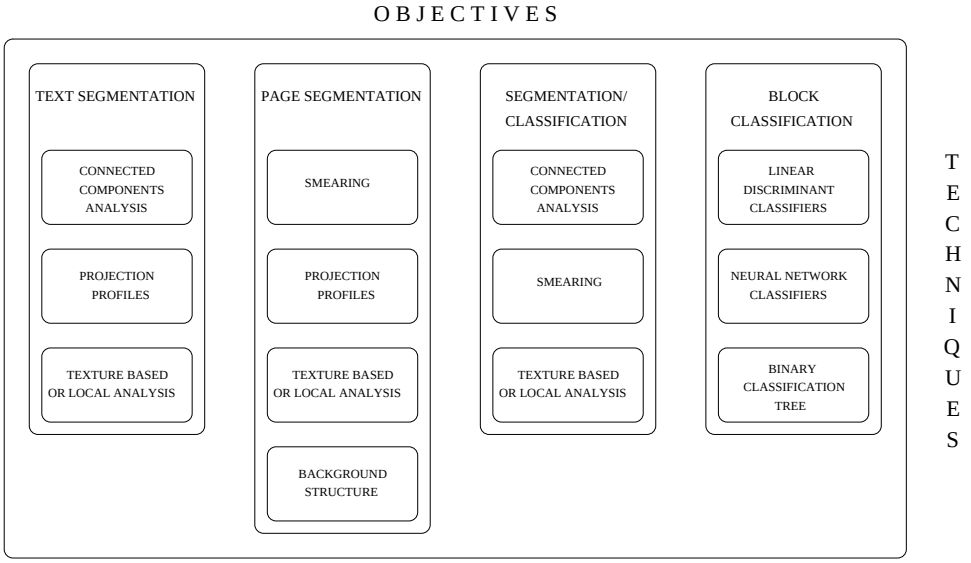

Figure 1: Page decomposition algorithms taxonomy we adopt in this review. The algorithms are clustered into four categories by objectives. Each group is in its turn divided by the adopted techniques.

are devoted to documents which contain only textual information [29, 42, 21, 43, 44, 45, 46, 47], or text mixed with some non text elements [48, 49, 50, 51]. In this case non textual parts are simply disregarded as do not matching the considered textual features.

- *Page segmentation* approaches: algorithms which aim at partitioning the document into homogeneous regions. They are grouped into the following classes related to the adopted segmentation technique: smearing technique [53], projection profile analysis [54, 55, 56, 57, 58, 59], texture based or local analysis [60, 61], and analysis of the background structure [62, 63].

- *Segmentation/Classification* mixed approaches: for some algorithms it is not possible to clearly separate the segmentation step from the classification one. The described techniques are based on connected component analysis [31, 66, 67, 68], smearing [71], and texture or local analysis [73, 74, 75, 76].

- *Block classification* approaches: algorithms which label regions previously extracted in a block segmentation phase. The major part are based on feature extraction and linear discriminant classifiers [53, 55, 59, 77], but other techniques are presented [78, 79].

The structure we propose represents a novel organization of the page decomposition field with respect to the usual classification of the algorithms into *bottom up*, *top down* or *hybrid*[2]. In our opinion this subdivision is somewhat forced and often produces an unnatural labeling of segmentation methods. This ambiguity is evidenced by the fact that, sometimes, the same algorithm is assigned to different classes by different authors.

4.1 Text segmentation approaches.

This section describes algorithms which aim at grouping the textual regions of a document into a hierarchical structure.

4.1.1 Connected components analysis.

O'Gorman describes a method [29] that aims at locating text lines and text blocks in images containing text, tables or equations. It is based on the computation of the document spectrum, or *docstrum*, already defined in Section 3, and makes use of the previously estimated skew angle. A preprocessing phase is performed to remove noise, *i.e.* small components or small holes in the components. The connected components are then clustered according to their areas. This step is suggested by the fact that the *docstrum* computes some average measures which are more reliable if the variance is low. The docstrum analysis, besides giving a rough estimation of the skew, provides an estimate of the between-lines and between-characters spacings. The between-characters spacing is estimated as the bin corresponding to the maximum value in the histogram of the distances d restricted to the couples having ϕ in the proximity of the estimated skew angle. The between-line spacing is estimated, analogously, from the histogram of the distances d whose corresponding angle ϕ is close to the direction perpendicular to the estimated skew. Text lines are obtained by merging components: a transitive closure is performed on within-line nearest neighbors pairs. For each

[2]Bottom up approaches start from grouping pixels together to detect words, lines, paragraphs,...; top down approaches start from the whole page which is segmented recursively into subregions, typically using high level knowledge about the document structure; the hybrid approaches are a combination of both the previous strategies.

group a mean square linear fit is performed on the centroids of the components. Finally, blocks of text are built by merging parallel lines which are either close in the perpendicular direction and overlapped, or collinear and close in the parallel direction. While the explained method is restricted to a single formatted page with one orientation, the author points out that it can be extended to documents containing (well separated) regions featuring different layout format and orientation. He suggests to perform a segmentation step after computing the k nearest neighbors, which group together components that appear in the same couple with distance d below a given threshold.

Hönes and Lichter [49] present a page decomposition algorithm which can deal with reverse text and multiple skew angles. The connected components of both foreground and background are extracted, in order to provide the basis for handling normal and reverse text. The connected components which are too small or too large with respect to their average size, are marked as irrelevant for line generation. For each component a list of its nearest neighbors is heuristically determined; only objects with the same color (foreground or background) can be neighbors. Information about their spatial arrangement (distance and angle) is maintained. Temporary lines are generated starting from triplets of neighbor components which are approximately aligned and have comparable size. Using the same criteria the triplets are iteratively expanded by adding new components to their boundaries. A relaxation step aims at reassigning components to lines and at building blocks of lines. This is performed by optimizing an objective function which takes into account line parallelism, components proximity and blocks homogeneity. Each connected component is labeled as text or non-text by considering some geometric features and a quality score of the line and block it belongs to. Finally, textual components are grouped into characters and words also reconsidering the components which were previously marked as irrelevant.

Déforges and Barba [50] describe a method for text extraction from complex gray level images. No assumptions are made on the shape and the slope of textual regions. The basic step is the extraction of word candidates from a multi-resolution pyramidal representation of the image [80]. The words are then simultaneously merged into lines and blocks depending on their spatial relationships and on their mutual distances. Within each blocks the text lines are clustered using a metric which takes into account their slopes and heights, in order to build more homogeneous text blocks. Isolated word candidates are independently analized: their region in the image is binarized, and some features are extracted in order to determine their textual

or non-textual nature.

The Minimum Spanning Tree (MST) method described by Dias [42], is based on a work of Ittner [81], which employs the MST to detect the text lines orientation (horizontal or vertical) of a document. Dias' algorithm makes the assumption that intercharacter spacing is smaller than interline spacing. The connected components are computed and each of them is represented by means of the bounding box. The bounding boxes are considered as the vertices of a graph whose edges are the hypothetical lines joining each pair of boxes. The cost associated to each edge is the minimum Euclidean distance between the bounding boxes it links. The minimum spanning tree of such a graph is computed. The text is segmented by breaking some branches in the MST that are selected according to a comparison between local and global information. Local information are, for instance, the branch length and the modes of run lengths of the two components linked by the branch. Global information are statistics such as a cutoff length calculated from the distribution of branch lengths, the average run length over all the components in the page, and the standard deviation of the modes of run lengths of each component. Strengths of this algorithm are the capability of working also with non rectangular blocks and the independence of text line orientation (horizontal or vertical). Furthermore, it can treat connected characters in the text; in fact the selected metric has the advantage to maintain short distances among groups of connected characters, differently from a distance computed between the centers of the components. The results of the experiments performed on 50 document images appear quite promising. The author observes that when the branch breaking process does not rely on significant statistics (sparse data) and a single incorrect branch break occurs, the algorithm produces a poor segmentation.

4.1.2 Projection profile methods.

A naive method to segment simple textual documents into lines is based on the analysis of the regularity of peaks and valleys in the projection profile of the image performed along the text skew direction. Peaks and valleys correspond, respectively, to text lines and between lines spaces. This method is used, for example, by Srihari and Govindaraju [21]. Observing that the profile of a single text line is shaped as twin peaks with a little valley in between, they can estimate the position of base and top lines in correspondence of the twin peaks.

Baird [43] presents a detailed description of a method for the analysis

of columns of text. The approach, named *global-to-local*, is characterized by the definition of a parametric model of a generic text column, which guides the analysis: the interpretation of an input image requires its partition and the inference of the model parameters. The main assumptions about isolated text columns are: text is printed along parallel ($\pm 0.05^\circ$) and roughly horizontal ($\pm 5^\circ$) lines, each with a dominant text size which varies within a broad known range; text is printed in a not connected style and detached characters are predominant. The connected components of the input binary images are extracted. Skew angle is estimated and eventually corrected (see [17]). The horizontal projection profile of the column is analyzed in order to segment it into text lines. Each text line is partitioned into characters which are preclassified: the most confident classifications and the *a priori* information about size and position of symbols with respect to the baseline, provide useful information for a robust estimate of text size and baseline position. Characters are merged into words by assuming the existence of a threshold value on the spacing between characters which can discriminate interword against intraword gaps.

Ha *et al.* [46] present a simple method for page segmentation and classification into words, text lines, paragraphs. It is based on the analysis of horizontal and vertical projections of the bounding boxes of the connected components. The method is applicable only to a restricted set of documents, as many assumptions are made: binary input image, good quality document, *i.e.* noise free and no touching characters, deskewed document, single column, and finally a hierarchy of increasing spacings between characters, between words, between text lines, between text blocks,...

Parodi and Piccioli [51] describe a method that aims at extracting text lines from unstructured documents having, possibly, a little skew. The approach relies on the projection profile analysis of narrow vertical strips, overlapping each other, extracted from the input image. For each strip the zones containing foreground pixels, called *line elements*, are localized. The line elements of successive strips are linked each other if their projections overlap and present comparable heights. For each element only one link with the elements of the next strip is allowed. The resulting lists of elements represent the text line candidates. The line skew is estimated as the slope of the best line fitting of the centers of elements. The page skew is estimated as a weighted average of the line slopes. The coordinate system is then rotated accordingly. The bounding rectangles surrounding each text line candidate are computed with respect to the rotated system. A filtering of the rectangles is then performed in order to discard non textual

elements. The number of characters for each text line candidate is estimated as the ratio width/height of the rectangle sides. Rectangles whose number of black/white and white/black transitions along the text line direction do not approximately match the estimated number of characters, are filtered away. Remaining text lines are grouped into blocks.

4.1.3 Texture based or local analysis.

Chen *et al.* [47] describe a segmentation algorithm for the detection of words in textual documents. The document image is sampled at a resolution of $150dpi$. A word block is defined as the rectangular regions which contain a word. Word blocks are detected by means of a pixel classifier which gives an estimate of the *a posteriori* probability of the pixel to be in a word block or not. These probabilities are estimated from a set of training synthetic images whose pixels has been previously labeled as belonging, or not, to a word block. A set of n *recursive closing transforms*, each characterized by a different structuring element, are applied to the training images thus producing n transformed images. Pixels of the training images are modeled by vectors of the values of the corresponding locations in the transformed images. Posterior probabilities of such vectors to be in a word block are estimated. In this way a probability map can be associated to the test images. The map is thresholded to obtain the word blocks. The threshold value is calculated from the histogram of the probability map: using a linear regression, a function is estimated between the histograms and the optimal threshold values of the images in the training set. The presence of ascender and descender characters may cause words belonging to different lines to merge into the same word block. A post processing step is performed to detect such blocks and to split them appropriately. The procedure is based on the comparison of the block height with the dominant block height in the document. Splitting is based on the detection of cut points in the projection profile of the probability map region corresponding to the block. Performance of the algorithm are evaluated only for synthetic images.

4.1.4 Analysis of the background structure.

Baird [45] describes a segmentation technique based on a previous work [44], in which the structure of the document background is analyzed in order to determine the geometric page layout. In a preprocessing phase, the components that appear to small or too large to be text are filtered

and the document is deskewed [17]. All the maximal rectangles, *i.e.* white rectangles which cannot be further expanded, covering the background are enumerated. Then a partial order is specified according to the area and the aspect ratio of the rectangles. A selection of N top ranked rectangles gives rise to a partial coverage of the background. Larger values of N produce more refined approximations of the background. The regions not covered constitute the blocks. The method does not rely on rules dependent on the characters set. It only requires a rough estimate of the range of text size.

4.1.5 Smearing based techniques.

One of the first approaches to text/non-text discrimination was presented by Johnston [48]. Several assumption are made: text is printed horizontally, the input image is clean, non-text areas are not immediately adjacent to the text, the character height and width in pixels are known. The basic idea is that text appears as a set of horizontal stripes. The algorithm works in two steps: (1) clean-up objects larger than the character size. It uses a sequence of horizontal and vertical morphological operators whose parameters depend on the character size. The result is a bitmap which is employed as a mask to lift up the text from the original image. Small isolated components and line segments will possibly remain. (2) clean-up objects smaller than the character size, using similar morphological operations. The output bitmap can be used as a mask to select only the text areas from the input image. Some problems may arise at the edges of the image where text components may disappear.

4.2 Page segmentation approaches.

This section describes algorithms aiming solely at the segmentation of the document into homogeneous regions, without classifying the obtained blocks.

4.2.1 Smearing based techniques.

The Run Length Smearing Algorithm (*RLSA*) is introduced by Wong *et al.* [53]. The input image must be a clean and deskewed bitmap. The algorithm operates on sequences of pixels, i.e the rows or the columns of the bitmap. A sequence $\mathbf{x}$ of 0's and 1's is transformed into a sequence $\mathbf{y}$ according to the following rules:

- 0's in **x** are changed to 1's in **y** if the run length of 0's is less than or equal to a predefined threshold value C;
- 1's in **x** are unchanged in **y**.

The effect is that of linking together neighboring black areas that are separated by less than C pixels. The degree of linkage depends on the value of C and on the distribution of white and black pixels in the document, with respect to scanning resolution. *RLSA* is applied by rows and by columns to a document image yielding two bitmaps. When dealing with $240dpi$ images, threshold values proposed for the horizontal and vertical smearings are different: $C_h = 300$, $C_v = 500$. The bitmaps are then combined in a logical AND operation and an additional horizontal smoothing is applied, using $C_h = 30$. The segmentation produced by *RLSA* for the textual regions is characterized by small blocks, usually corresponding to lines of text. The algorithm is fast but presents some limits: the threshold values have to be set *a priori*, it can be applied only to documents with a rectangularly structured layout. In order to get rectangular blocks a post processing step is required. *RLSA* received great interest, principally because of its easy implementation, and has been employed, with some modifications, in several systems to perform the segmentation phase [18, 82, 83, 79, 77].

4.2.2 Projection profile methods.

A popular approach to page segmentation, described in [54, 84], is the Recursive X-Y Cuts, *RXYC*, algorithm. It is applied to clean, deskewed binary input images. The *RXYC* algorithm recursively splits the document into two or more smaller rectangular blocks which represent the nodes of a tree structure. At each step of the recursion, the horizontal or vertical projection profiles are alternately computed. The segmentation of a block is performed in correspondence of the valley of the projections that are larger than a predefined threshold. Thresholds may be different for each level of recursion and may depend on the knowledge about the document class. *A priori* information is needed particularly to define the stop criterion in the recursion. This technique is suitable only for layouts that are decomposable by a sequence of horizontal and vertical subdivisions.

Several authors make use of the *RXYC* algorithm, eventually modified. Wang and Srihari [55] compare the *RLSA* and *RXYC* approaches. The *RXYC* algorithm is selected as being more suitable for the newspaper segmentation task. Nagy *et al.* [56, 57] present a top down approach that

combines structural segmentation and functional labeling. Segmentation is based on the *RXYC* procedure and is guided by a knowledge of the features of the document layout. A trainable algorithm for page segmentation based on RXYC procedure is presented by Sylwester and Seth [58].

Goal of the segmentation algorithm presented by Pavlidis and Zhou [59] is to find the largest possible *column blocks*. They are defined as subregions of the input image that contain a unique data type and are well separated each other by white space straight streams. Documents with a little skew, as well as columns with different tilt angles (a distortion caused by copying) can be handled. The algorithm is based on the analysis of the vertical projection profiles computed locally, *i.e.* over consecutive short blocks of scanlines. Wide white spaces in the vertical projections correspond to column gaps which act as separators of the so called *column intervals*. Column blocks are iteratively constructed by merging pairs of column intervals which satisfy the following rules: are very close in the vertical direction, have approximately the same width, their vertical projections are (nearly) contained one in the other. In the next step a new merging process takes place along with the estimation of the skew angle of the resulting blocks. For each column block a central axis is computed by interpolating the central points of the column intervals with a straight line. The merging of the column blocks is guided by similar rules as above along with a constraint on the alignment of the central axes. An interesting feature of the approach is that it exploits the flexibility of a bottom up method and at the same time mitigates its inherent inefficiency by using the column intervals as the basic elements of the merging process. This choice guarantees to deal with a few structured objects instead of many low level objects like pixels, runs or connected components. A limit, as reported by the authors, is that text printed in large size, such as titles, may produce fragmented column blocks, due to large gaps between words. Therefore some parameters, such as the width of the column gaps, need to be adjusted for different type of documents. Notice that this drawback is common to other approaches, although not explicitly declared by their authors.

4.2.3 Texture based or local analysis.

In the work of Jain and Bhattacharjee [60] text/non-text segmentation is viewed as a texture segmentation problem. The paper presents a multichannel filtering approach to texture segmentation. The basic assumption is that regions of text in a document image define a unique texture which

can be easily captured by a small number of Gabor filters. Filters are applied directly to the input gray level image. No *a priori* information about layout, font styles, skew angle are required. The texture segmentation algorithm operates in three main steps: (1) filtering the image through a bank of n Gabor filters; (2) computing the feature vectors as the n local energies estimate over windows around each pixel; (3) clustering the feature vectors into K clusters; the coordinates (x, y) of each pixel are used as additional features. A typical value of K used in the experiments is 3 in order to emphasize three texture types: textual regions, uniform regions (background or picture with low intensity variations), and boundaries of the uniform regions. In some applications $K = 4$ proves to be a more appropriate choice (handwritten text). To evaluate the experiments, in a supervised classification mode, some training patterns are randomly extracted from the regions belonging to the different classes to be discriminated. As long as the imaging environment remain the same, the trained classifier (nearest neighbor in the experiments) can be used for subsequent images. The selection of the n filters for optimal clustering appears to be a critical task, in fact these filters cannot guarantee the best performance on a whole class of segmentation problems.

Tang *et al.* [61] describe a page segmentation approach based on modified fractal signature. The input gray level image is considered as an approximation of a fractal surface. The surface area is used as a *fractal signature*, FS, which characterizes the local geometric structures of the different zones of the document image. The surface area is defined in term of a unit measure δ and its value increases within limit when δ decreases following the generally approximated power law: $A_\delta \approx \beta\delta^{2-D}$, where β is a constant and D stands for the fractal dimension. Taking the logarithm of both sides we can see that the fractal dimension can be viewed as a slope in a log-log space: $log(A_\delta) \approx log(\beta) + (2 - D)log(\delta)$. Therefore, the fractal dimension D can be determined by computing the surface area at only two different scales, δ_1 and δ_2. The surface area, at scale δ, is computed by counting all the points at distance less than or equal to δ from the surface and dividing the count by 2δ. The authors highlight how the fractal signature can be used to distinguish different regions such as: text areas, graphic areas and background areas. They divide the image into small non overlapping regions and classify them according to their fractal signature.

4.2.4 Analysis of the background structure.

Normand and Viard-Gaudin [62] present a 2D smoothing algorithm for the analysis of the document background, which is basically an extension of the RLSA to two dimensions. They consider two structuring elements, the square and the octagon, and choose the octagon for its better isotropic property. Each background pixel is replaced by an index, that depends on the size of the widest structuring element that can be placed over it without intersecting foreground pixels. A hierarchical tree structure is computed by thresholding the map with different decreasing values and keeping trace of the connected components which are generated. Each node of the tree structure represents a region of the map. The root node represents the whole map, and the children of each node are the connected components obtained by thresholding the region of the map represented by the node. The leaves of the tree correspond to the connected components of the document. This structure is used to compute an effective segmentation of the image into blocks. It is obtained by selecting *relevant nodes* in the tree structure and by extracting their corresponding regions. As a critical problem remains the definition of a criterion for the automatic selection of relevant nodes.

Kise *et al.* [63] present a method which is based on the thinning of the document background. The segmentation is determined by selecting subsequences of chains which circumscribe document blocks (loops). The proposed algorithm attempts to filter away unnecessary chains and to retain such loops. Firstly, chains ending with a terminal pixel are removed. Remaining chains are analyzed in order to eliminate those lying between characters and text lines, and to preserve chains between certain regions of the same type (*e.g.* between columns) and of different data type (such as text and halftone). Two features are used: the distance of chain points from foreground pixels, and the so called *difference of average line widths*, which takes into account some characteristics of the adjacent foreground regions. The filtering process requires a critical tuning of some threshold which depend on the spacings of the input document. The presence of wide gaps between words or characters (*e.g.* in titles) may cause erroneous segmentations.

4.3 Segmentation/Classification mixed approaches.

The algorithms described in the following approach the page decomposition as a undecomposable problem and perform simultaneously the segmentation

and classification phases. They are grouped on the basis of the adopted technique.

4.3.1 Connected components analysis.

Akiyama and Hagita [31] present an approach for the page decomposition of deskewed documents (see Section 3.4) containing text and graphic zones. Some assumptions are made on the document content, *i.e.* it includes headlines, text line blocks, graphics, each contained into an isolated rectangular area, and solid or dotted lines which act as separators. Further, a set of 15 geometric properties is assumed for the document layout. The first step is the field separators extraction performed by analyzing the connected components of the image. Solid lines are characterized by very elongated components with small crossing count (*i.e.* the number of white/black transitions in horizontal and vertical directions), while dotted lines are characterized by components surrounded by long continuous white pixels. The next step is the estimation of the text lines thickness. The histogram of the heights of the remaining components is computed. The mode T of the histogram is assumed as the best estimate of the text line thickness. Components whose height is below $1.5T$ are text candidates; otherwise the component is put in the headline or graphics candidates depending on its crossing count (low or high, respectively). The text blocks are then extracted starting from the candidate text components. These candidates are clustered by means of two criteria: field separator or blank areas in the projection profile. In order to avoid over segmentation, the separation process is terminated when the crossing count of all the clusters drops below a predefined limit. Adjacent text components belonging to the same block are merged to build text lines. Isolated or irregularly spaced text lines are added to the list of headline or graphic candidates. Headlines are then detected using the same method. Graphics blocks are finally extracted by merging graphic areas and eventual captions, *i.e.* sequences of text components that are close to the graphic area.

The technique used by Zlatopolsky [66] is based on a progressive growing process that starts from the connected components. The growing process is controlled by some thresholds whose values are locally determined according to the size of the objects to be merged. A working hypothesis is that text blocks are surrounded by blank areas. A preprocessing phase aims at detecting non text components, *i.e.* small and very elongated ones. These components are examined to reveal the presence of block separators like

frames, or are merged to build graphic blocks. Within each frame, text-like components are grouped into *line segments* by a merging process which takes into account their horizontal and vertical distances. At this stage the skew angle of the page is estimated as the average orientation of the segments which are long enough, and the coordinates system is rotated accordingly. Text blocks are obtained by merging line segments which are close in both horizontal and vertical direction and similar with respect to certain line features. As declared from the author, this method cannot work in presence of many broken characters or textured images.

Wang and Yagasaki [67] present a method which is based on a hierarchical selection and classification of the connected components. First, the external boundaries, or outlines, of the connected components are searched, skipping at this stage the possible contained components. Very large components are labeled as non text and the average size of the remaining ones is computed and used to estimate a threshold for a text and non text classification. Non text components are then classified as pictures, frames, lines, tables, slanted lines, line-arts or unknown by using characteristics such as height, width, thickness, density, statistics on black runs, number and arrangement of holes, and size of adjacent components. Outlines of connected components contained in components labeled as table, frame or line-art are extracted and classified in the same way. Furthermore, two functions are provided to detect dotted lines and invisible lines, *i.e.* elongated white areas in the background. The components labeled as *unknown* are then analyzed in order to build horizontal or vertical title lines (large font). The textual components are clustered into blocks by a closeness criterion, using statistics of the spacings among components. Textual blocks are split if a line separator or an invisible line passes through them. Each text block is then classified as horizontal, vertical, or sloped by taking into account the sizes of its components and their gaps. Text lines are built by considering distances and overlapping ratios. The slope of each text line is computed by means of the last square method and the block skew is estimated as the average of the lines skew angles. Because representation of blocks as rectangles may cause undesirable overlaps a representation depending on outlines of components is adopted.

Simon *et al.* [68] present a page decomposition approach which is applied to chemical documents. The input is a binary deskewed image. The algorithm is based on the computation of the minimum spanning tree (MST) of a multi-level graph. At the first layer the nodes of the graph are the connected components and the edges are labeled by a suitable distance between

the nodes. The distance definition embodies several heuristics which vary according to the current layer of the graph and to the classification (text or graphic) assigned to the nodes. Components are iteratively merged by considering the shortest edge in the MST. In the first layer words are detected by merging components which are close and horizontally aligned. At the second layer these words become the nodes of a new graph whose MST is used to detect lines. Analogously, lines are merged into blocks at the next layer. The change of layer is controlled by a threshold on the ratio between the label of the current shortest edge and the label of the egdes involved in the last merge. Following the approach described in [71], at the end of each stage the nodes are classified using some geometric features, like the height of the group of components associated to the node, the aspect ratio of its bounding box, and the density of black pixels.

4.3.2 Smearing based techniques.

Tsujimoto and Asada [71] describe an analysis and interpretation system for different types of documents: magazines, journals, newspapers, manuals, letters, scientific papers. The segmentation phase uses a bottom up method which, in successive steps, groups small components into larger and larger ones. Adjacent connected components are aggregated into segments through an horizontal smearing of the runs separated by a small enough gap. Segments are then preclassified into one class: text line, figure, graphics, tables, frames, horizontal line, vertical line, noise, accordingly to some physical properties, such as width, height, aspect ratio, horizontal and vertical projection, number of smeared white runs. Four thresholds are used to define the separation surfaces among the different classes. In the next step the merging of adjacent text line segments takes place. Two adjacent text segments (*i.e.* words) are merged if the horizontal distance is below a threshold which is proportional to the segment height, and the left edges of the resulting segments are vertically projected to obtain an histogram. Local maxima in the histogram define the left edges of the columns.

4.3.3 Texture based or local analysis methods.

Some techniques to recognize and separate textual, graphical and pictorial regions of documents are discussed by Scherl *et al.* [73]. They are based on the analysis of local properties. The document image is divided into square, small, overlapping windows for each of which features are extracted in order

to classify the window. The first method is based on the observation that the spatial Fourier spectrum obtained for typed or machine printed text, diagrams, photograph and halftone have quite different behaviors and provides some information, such as the distance between text lines. According to the authors, this segmentation method has the drawback of being time consuming, and has not been considered in practical applications. An alternative method presented to distinguishing between text or graphics and picture is based on the analysis of statistical features extracted from the local gray level histograms. Using the knowledge that the brightest gray levels are most frequently in the background of text, the percentage of the bright levels within the window is computed. The window is classified as text or graphic if this feature value is greater than a given threshold, as picture otherwise. Another, more flexible, method is also proposed. The authors observe that skewness and curtosis capture the typical shape of the histogram of a text region. In fact, experiments performed on many text regions show that these features tend to cluster along a quadratic curve in the skewness/curtosis space while picture regions do not.

A similar method is presented by Sauvola and Pietikäinen [74]. The binary document image is divided into small square windows whose dimensions depend on the input resolution (10×10 to 20×20 in the experiments). For each window the following features are computed: black/white pixels ratio, average length of black runs, signal cross-correlation between consecutive vertical lines, and signal cross-correlation between vertical lines with five pixels distance. On the basis of the features values, each window is labeled as text, picture, or background, following a rules-based classification. Local connectivity operators are applied iteratively to the labels map in order to regularize region edges and suppress noisy classification.

Another approach where the layout segmentation is posed as a texture segmentation problem is described by Jain and Zhong [75]. The authors aim at overcoming the limitations of the approach presented in [60]. The method has the following features: (1) robust to the alphabet of the document; (2) can be applied to the input gray level document image; (3) segments the document into four distinct classes: text, halftones, graphics, background; (4) a low scanning resolution ($100 dpi$) is sufficient; (5) can be trained to perform language separation. The segmentation of the document is performed by classifying each pixel of the image into one of three classes (text or line drawing, halftone, background) and then by grouping pixels which are in the same class and close each other. The classification is based on the characterization of the textural properties of the neighborhood of each pixel. In order

to use heuristics and domain specific knowledge, special purpose filters are selected. They are optimized for the given textures by training a multilayer neural network (input layer, mask layer, hidden layer and output layer). The network is a multilayer perceptron trained with a backpropagation algorithm. Initially, the mask layer contains 20 masks (7×7), then a pruning phase takes place in order to remove the less discriminant masks. It involves the computation of connected components and thresholding on the length of the shorter side of their bounding boxes. The image obtained from the output of the neural network is smoothed through the selection of the majority class on 3×3 neighborhood of each pixel, the morphological closing with a 1×3 structuring element and opening with a 3×3 structuring element. A post processing phase removes small noisy elements, merges neighboring regions and places bounding box around the labeled regions. The output layer has three nodes corresponding to the three classes. Since text and line drawing are in the same class, they must be discriminated in a second step. This is a quite fragile step requiring some strict conditions: an appropriate binarization, no touching characters, and connected line drawings. No quantitative results of the performed experiments are reported.

Another method where document image segmentation is treated as a texture segmentation problem has been recently presented by Etemad *et al.* [76]. They propose a multi scale wavelet packets representation of the document image. Starting from a family of orthonormal basis functions they select a suitable wavelet packed tree by using a maximum class separability criterion, where the classes depend on the specific application. A mapping is defined from the signal space to the feature space by computing the second and third central moment of the wavelet packet components. Moments are computed within local windows on each subband. A fuzzy classifier is implemented by a neural network and using a conjugate gradient method. The training set is constituted by preclassified square windows randomly extracted from a sample of document images scanned at 200-300dpi. The classifier provides a soft decision vector, *i.e.* a score in $[0, 1]$ for each class. The segmentation is performed by integrating soft local decisions. This scheme guarantees both a good spatial resolution and robustness with respect to local unreliable decisions. The integration scheme combines knowledge coming from several sources: local spatial information, both within scale and across scale, and *a priori* information about the domain. In the experiments document regions are classified into four classes: background, text, picture and graphics. The background class is detected in a preliminary step. The authors report results on documents with complex layout

highlighting some advantages of their method: independence of document layout structure, capability to provide domain specific and adaptive feature extraction and possibility to deal with overlapped or mixed classes.

4.4 Block classification approaches.

Block classification techniques described in this section can be grouped into features extraction and linear discriminant classifiers [53, 77, 55, 59], binary classification tree [78], and neural networks [79].

4.4.1 Linear discriminant classifiers.

The classification algorithm described by Wong *et al.* in [53] computes some basic features from the blocks produced in the segmentation step (see Section 4.2) which makes use of a smearing technique, and discriminates between text and images by means of a linear classifier which adapts itself to varying character heights. An additional classification step resolves uncertain classifications among single text line, solid line and image. The following measurements are computed for each block: total number of black pixels (BC) after smearing; sides of the bounding box of the block ($\Delta x, \Delta y$); total number of black pixels in the corresponding region of the input document (DC); horizontal white/black transitions in the original data (TC). They are employed to compute the following features:

- height of the block ($H = \Delta y$);
- eccentricity of the bounding box ($E = \Delta x / \Delta y$);
- ratio of the number of black pixels to the area of the bounding box ($S = BC/(\Delta x \Delta y)$;
- mean length of horizontal black runs of the original data ($R = DC/TC$).

These features are used to classify the block. The underlying idea is that text blocks are characterized by a similar height H and mean length of black runs R. Firstly, a cluster in the (H, R)-space is selected which, with a high probability, contains only text blocks. This is obtained by using some *a priori* information about typical size of characters. Then, for the blocks in the cluster the mean values of the features H and R are computed: $\bar{H}$ and

$\bar{R}$. The following classification logic is then used to classify all the blocks:

$$(R < C_{21}\bar{R}) \wedge (H < C_{22}\bar{H}) \Rightarrow \text{Text}$$
$$(R > C_{21}\bar{R}) \wedge (H < C_{22}\bar{H}) \Rightarrow \text{Horizontal solid black line}$$
$$(E > 1/C_{23}) \wedge (H > C_{22}\bar{H}) \Rightarrow \text{Graphics and halftone images}$$
$$(E < 1/C_{23}) \wedge (H > C_{22}\bar{H}) \Rightarrow \text{Vertical solid black line}$$

where C_{21}, C_{22}, C_{23} are user defined parameters. Values suggested by training experiments are: $C_{21} = 3, C_{22} = 3, C_{23} = 5$. Some problems may arise in presence of linked lines of text (too close or touching) or very high lines of text (like titles), which cause the relative block to be assigned to the class *Graphics and halftone images*. For this reason a further reclassification rule is introduced for the blocks in class *Graphics and halftone images*. This rule uses information about shape factors of the components of the block (border to border distance).

Shih and Chen [77] describe a variation of the classification method designed by Wong *et al.* in [53]. They employ basically the same features, plus features relative to the vertical transitions from white to black pixels: height of the block (H), aspect ratio (R), density of black pixels (D), horizontal white/black transitions per unit width (TH_x), vertical white/black transitions per unit width (TV_x), horizontal white/black transition per unit height (TH_y). The following classification rules, which are independent of character size and scanning resolution, are used to classify blocks into text, horizontal/vertical line, graphics, or picture class:

$$c_1 H_m < H < c_2 H_m \Rightarrow \text{Text}$$
$$(H < c_1 H_m) \wedge (c_{h1} < TH_x < c_{h2}) \Rightarrow \text{Text}$$
$$(TH_x < c_{h3}) \wedge (R > c_R) \wedge (c_3 < TV_x < c_4) \Rightarrow \text{Horizontal line}$$
$$(TH_x > 1/c_{h3}) \wedge (R < 1/c_R) \wedge (c_3 < TH_y < c_4) \Rightarrow \text{Vertical line}$$
$$(H \geq c_2 H_m) \wedge (c_{h1} < TH_x < c_{h2}) \wedge (c_{v1} < TV_x < c_{v2}) \Rightarrow \text{Text}$$
$$D < c_5 \Rightarrow \text{Graphics}$$
$$\text{otherwise} \Rightarrow \text{Picture}$$

where H_m is the average height of the blocks and c's are 11 parameters that define the separation surfaces. Their values were selected after the observation of the features behavior on a large sample of documents acquired at different resolutions, containing blocks of different classes and text with different font styles and sizes.

Wang and Srihari [55] point out some limitations of the approach proposed by Wong *et al.* [53] related to the needed information about the geometric characteristics of the text lines. In many cases (skewed input, noise, touching lines,...) the segmentation algorithm cannot correctly segment each line of text, and a block containing two or more lines may be erroneously classified into the graphics and halftone images class. They introduce a new set of features which are independent of the information about the block size. These features are based on the textural nature of the block content. The features are computed by compiling two matrices: BW (black/white run) and BWB (black/white/black run). Matrix BW aims at representing the following properties of the texture of a block of text: line segments with different widths for different font size, and line segments assembled with certain density. A black/white pair run is a set of horizontal consecutive black pixels, say n, followed by a set of consecutive white pixels, say m, and its length is the total number of pixels $n + m$. The entry $BW(i, j)$, with $i = 1, \ldots, 9$, specifies the number of times that the block contains a black/white run of length j, in the horizontal direction, with m/j close to $i/10$. Matrix BWB tries to capture the peculiar features of the line drawing blocks: there exist several large white spaces between black lines. A black/white/black combination run is a pixel sequence in which two black runs are separated by a white one. Its length is defined as the length of the white run. The length of the black runs is quantized into three categories: 1 if length is in $(1, 4)$, 2 if length is in $(5, 8)$ and 3 if length is in $(9, 12)$. The entry $BWB(i, j)$, with $i = 1, 2, 3$, specifies the number of times the block contains a black/white/black run of length j, in the horizontal direction, with the black runs belonging to same category i. Two features, $F1$ and $F2$, which respectively emphasize the short and the long runs, are derived from matrix BW, and one feature, $F3$, emphasizing long runs, from matrix BWB. Thresholds are used in the feature computation in order to weight the contribution of small matrix entries and of short run lengths. These three feature are sent to a linear classifier with five output classes: small letters (less than 14 points size), medium letters (from 14 to 32 points size), large letters (more than 32 points size), graphics and halftone. The classification criterion is determined on the basis of a sample of blocks collected from the five different categories. The separation surfaces in the space (F_1, F_2, F_3) are experimentally determined by using a fixed error correction procedure (see [85]) on a preclassified training set. Experiments were performed on a set of newspaper images digitized at $100 dpi$ and $200 dpi$. It is not clear if experiments have been conducted using a test sample completely disjoint

from the training one.

In the article of Pavlidis and Zhou [59], blocks are classified into three classes: *text*, *halftone images* (produced with dithering or error diffusion techniques) and *diagrams*. The discriminator for halftone class versus the others is based on the signal cross-correlation function computed on the binary image. It is defined between scanlines, y and $y + r$, as follows:

$$C(r, y) = \frac{1}{L} \sum_{k=0}^{L-1} [1 - 2p(y, k) \text{ XOR } p(y + r, k)]$$

where L is the length of the scanlines and $p(y, k)$ is the value of the pixel in the position (y, k). The authors observed that, at varying r, $C(r, y)$ behaves differently on text blocks or diagrams with respect to halftone images. Typically, correlation of adjacent scanlines is high and decreases rapidly for text and diagrams, while is rather flat and exhibits periodicity for the halftone blocks. Four features are extracted to describe the correlation behavior and a linear discriminant function is employed to classify a block as halftone or not. In the latter case, the number of black pixels with respect to the number of white pixels in the block, b/w, is considered as a feature able to distinguish text from diagrams. Parameters of the discrimination function and the b/w ratio threshold need to be tuned.

4.4.2 Binary classification tree.

Sivaramaakrishnan *et al.* [78] describe a statistical technique for the classification of homogeneous regions extracted from document images. The blocks are supposed to be rectangles with sides aligned to horizontal and vertical directions. The classification is based on the following set of 67 features that are extracted from each block: The number of black and white runs along the four principal directions (8 features). The mean and variance of run length for black and white runs along the four directions separately (16 features). The mean and variance of the projection profiles along the four directions are computed (8 features). Autocorrelations of the following four functions: projection profile, number of black runs for each line, spatial mean of the black pixels for each line and run length mean for each line, are computed. The spatial shift for which the autocorrelation function goes to 10% of its maximum value is computed, along with the derivative of the autocorrelation function when the spatial shift goes to zero. These two features are computed separately for the four functions and for the four

directions (32 features). The density of black pixels, the area of the block, and the ratio of the block width and the width of its column, represent the last 3 features. The classification is performed by traversing a binary classification tree, whose leaf nodes are labeled by a single class. The decision tree is built from a training set of blocks labeled with the true class (one of: small text, large text, math, table, halftone, map/drawing, ruling, logo and other). The construction process is recursive: initially the root node contains all the training samples each represented by a 67-features vector. A splitting criterion is defined to divide the node samples into two subsets aiming at separating samples belonging to different classes. It is based on the maximization of an objective function, called *purity*, which is defined as the sum of the entropy of the class distribution in the left and in the right child, respectively, after the node splitting. Samples are split by selecting a feature and a threshold value: if the value of the feature is below the threshold, the sample is assigned to the left child, otherwise to the right one. The splitting process terminates when all the sample in the node are in the same class, or when the number of samples falls below a given threshold. Each leaf node is labeled with the major represented class in the node. An unknown block is classified by traversing the decision tree until a leaf node is reached: the block is assigned to the class of the node. At each node the block follows the link to the left or to the right child according to the discriminant feature and threshold associated to the node. Experiments have been performed on a data set, from the University of Washington Database [86].

4.4.3 Neural network classifiers.

Le *et al.* [79] present a new method for the classification of blocks extracted from binary document images. The authors implement and compare four different neural approaches for the classification of blocks into two classes: text and non-text. The considered neural models are *Back propagation*, *Radial basis functions*, *Probabilistic neural networks* and *Kohonen's self organizing features map*. The basic motivation is to eliminate the manual selection of several parameters which is a critical task often required in other approaches [53, 59]. The segmentation phase is implemented by means of the *RLSA* algorithm. For each block, with sizes D_x and D_y, they compute the following quantities, the number of black pixels after smearing BC, the number of black pixels in the original data DC, and the number of white/black transitions in the original data TC. The following features are then computed

and used for blocks classification:

$$\begin{aligned} H &= D_y \\ E &= D_x/D_y \\ S &= BC/(D_xD_y) \\ R &= DC/TC \\ HR &= H \cdot R \\ ER &= E \cdot R \\ SR &= S \cdot R \end{aligned}$$

Note that the first four features are identical to the features used by Wong *et al.* in [53]. Experiments have been carried out using a database of 50 document images acquired at $200dpi$ from 12 medical journals. The data set is randomly divided into a training and a test set. The reported results, in terms of correct classification rate, are: 99.35% for *Back propagation*, 99.61% for *Radial basis functions*, 98.18% for *Probabilistic neural network* and 99.22% for *Self organizing feature maps*.

4.5 A summary.

Page decomposition algorithms have been generally designed to operate in specific domains and this fact gave rise to a variety of methods which are characterized by different objectives and different assumptions about input type, layout structure and expected data types. According to the reviewed papers, a commonly accepted list of classes in which the document is decomposed does not stand out: various authors define their own set which depends on the application. Typical classes are text, pictures and diagrams, but also line drawings, tables, line separators,... Sometimes text blocks are further classified into columns, paragraphs, lines, words; text attributes such as size are also detected. A weak aspect which is common to all the reviewed algorithms, is the lack of information about the handling of such regions which do not belong to one of the expected categories.

Ideally, page decomposition should be based solely on the geometric characteristics of the document image without requiring any *a priori* information about a specific format. Actual document analysis systems employ, often in an implicit way, a set of generic typesetting rules which are valid within large categories of documents. As an example, Nagy [87] reports a set of generic

typesetting rules for Latin text. Typically, assumptions are made on the white spaces that act as separators of the different regions in the document. In practice, they consist in a hierarchy of constraints on the relative size of spacings between different objects, *e.g.* spacings between characters are smaller than spacings between words, which, in turn, are smaller than spacings between lines and so on. The larger is the number of objects involved in the spacing protocol, the stronger is the assumption.

Many techniques rely upon *a priori* information about the document layout. A typical classification which is reported in the literature is between *Manhattan* and *non-Manhattan* layouts. Unfortunately, a clear and common definition of Manhattan layout does not stand out: Baird [38] says "Manhattan layouts [...] they can be partitioned into isolated, convex blocks (columns) of text by horizontal and vertical line segments cutting through white space", O'Gorman [29] says "Manhattan layouts, that is layouts whose blocks are separable by vertical and horizontal cuts", Haralick [2] says: "A Manhattan page layout is one where the regions of the page layout are all rectangular and the rectangles are in the same orientation. [...] Furthermore, each pair of rectangles either is mutually exclusive or one contains the other". Therefore, we avoid to use the term *Manhattan* and distinguish layouts into four main groups: layouts which are decomposable by means of horizontal and vertical cuts (xy-cut), layouts which are partitionable into rectangular blocks, layouts whose regions can be enclosed by right polygons[3] and freely shaped layouts.

Another class of assumptions is related to the characteristics of the input image. In the major part of cases the input image is requested to be noise free, binary, and deskewed, although algorithms have been proposed which deal with gray level images, skewed or noisy documents. Algorithms that work on gray level document images are able to exploit information that would be lost after a binarization phase. The major drawback is the high computational cost due to large amount of data to be handled.

As far as skew is concerned, the reviewed papers may be broadly divided into three categories: (1) For many approaches a deskewed input image is mandatory. (2) Some other algorithms deal with single-angle skewed documents, either because they are skew insensitive, or because they include a skew detection phase. (3) Others can detect and manage differently skewed regions in a document.

Although, several authors do not require explicitly noise free input doc-

[3] With *right polygon* we intend an orthogonal polygon without holes.

uments, the robustness degree of the proposed techniques with respect to the noise level is not emphasized.

A summary of the main characteristics of the reviewed algorithms is presented in Tables 3, 4 for text segmentation algorithms, in Table 5 for page segmentation algorithms, in Table 6 for segmentation/classification mixed algorithms, and, finally, in Table 7 for block classification algorithms.

5 Evaluation methods for page decomposition.

The problem of automatic evaluation of page decomposition algorithms has been faced only recently. The *text-based approach* was the first proposed in the literature [88, 89, 90]. The quality of page segmentation is evaluated by analyzing the errors in the recognized text. First, page decomposition and character recognition procedures are applied to the document page, and the result is output as an ASCII string. The quality score is based on an approximation of the cost of human editing to convert the generated string into the ideal ground truth string. The advantage of this technique is that it is purely text-based, and therefore does not require the page segmentation subsystem to specifically output any kind of zoning results. In addition, although its underlying string-matching algorithms are rather elaborate, the overall approach is straightforward. Ground truth files for use with this system are very easy to create. Therefore, the text-based zoning evaluation approach has been well accepted by the document recognition community. Nonetheless, this system suffers from the limitation that it can only deal with text regions. The involved score only reflects accuracy on text zones and the segmentation of document images containing different non textual regions cannot be evaluated.

To overcome this limitation a *region-based approach* has been more recently introduced [91, 92, 47, 93, 94, 95]. According to it, a document is defined in terms of a hierarchical representation of its structure and content, *e.g.* layout structure, logical structure, style and content. Following the region-based approach, the segmentation quality is evaluated at the different levels of the representation, in terms of correspondence of homogeneous regions by comparing the segmentation output of the system under investigation and the corresponding pre-stored ground truth.

method	*reference*	*input type*	*layout*	*text regions*	*assumptions and limitations*	*advantages, characteristics*
Connected Components Analysis	O'Gorman [29]	b/w 300*dpi*	rectangular blocks	text lines, text blocks	text only, clean docs; detached chars	estimation of parameters; multiple skews are dealt
	Hönes Lichter [49]	b/w 200-300*dpi*	rectangular blocks	chars, words, text lines, text blocks	cleaned docs; line gaps wider thar char gaps	non text, reverse text and multiple skews are dealt
	Déforges Barba [50]	g.l. 75*dpi*	freely shaped	text lines, text blocks.		noisy background and multiple skews are dealt; non text may be present
	Dias [42]	b/w 300*dpi*	freely shaped	headers, footers, text lines, text blocks	text only; no skew; line gaps wider than char gaps	independence of text orientation
Projection Profiles	Srihari Govindaraju [21]	b/w 128*dpi*	single column	text lines	text only, cleaned docs; one text direction	freely shaped regions skew is dealt
	Baird [43]	b/w 300*dpi*	single column	chars, words, text lines	text only; detached chars; one text direction	freely shaped regions; baseline computation; line spacing and text size may vary; skew up to 5° is dealt
	Ha *et al.* [46]	b/w 300*dpi*	single column	words, text lines, paragraphs	text only, cleaned docs; no skew; hierarchic spacing protocol	determination of text orientation
	Parodi Piccioli [51]	b/w 80*dpi*	freely shaped	text lines	one text direction	skew up to 6° is dealt; non text may be present

Table 3: Some features of text segmentation algorithms: input type of the documents (binary, gray-level) and working resolution, layout structure, searched text regions, assumptions and limitations, advantages.

method	*reference*	*input type*	*layout*	*text regions*	*assumptions and limitations*	*advantages, characteristics*
Texture or Local Analysis	Chen *et al.* [47]	b/w 150*dpi*	freely shaped	words	text only (synthetic) docs	skew up to 0.5° is tolerated; trainable
Background Analysis	Baird *at al.* [44, 45]	b/w	right polygons	text blocks	text only, cleaned docs; one text direction; spacing constraints	skew up to 5° is dealt; no input parameters
Smearing	Johnston [48]	b/w	rectangular blocks	text blocks	cleaned docs; no skew; known char size; interblock gaps greater than char size	non text may be present

Table 4: Part 2. Some features of text segmentation algorithms.

5.1 The text-based approach

A text-based method for measuring the performance of zoning capabilities of commercial OCR systems has been introduced by Kanai *et al.* [90]. The zoning score is based on the number of edit operations required to transform an OCR output to the correct text, using string matching algorithms [96, 97]. Three are the edit operations considered: *insertion, deletion, move*. The cost of correcting the OCR-generated text is calculated in two steps. First, the minimum number of edit operations is estimated. Next, the total cost is calculated according to the cost of each operation.

The edit operations are counted with the following procedure. Let S_c be a string of characters corresponding to the correct text of a page and S_o be the OCR output using the automatic zoning. S_c and S_o are compared, and matches are identified. The longest substring common to S_c and S_o is found and constitutes the first match. The second match is determined by finding the longest substring common to an unmatched substring of S_c and S_o. The process continues until no common substring can be found. The number of unmatched characters D in S_o is the number of deletion operations needed, while the number of unmatched characters I in S_c is the number of insertion operations needed. The number of moves M required

method	*reference*	*input type*	*layout*	*assumptions and limitations*	*advantages, characteristics*
Smearing	Wong *et al.* [53]	b/w 240*dpi*	rectangular blocks	clean docs; no skew; known chars size	segmentation of graphics or picture blocks and text lines
Projection Profiles	Nagy *et al.* [54, 84]	b/w 300 *dpi*	xy-cut	cleaned docs; no skew;	segmentation into columns, blocks, text lines
	Pavlidis Zhou [59]	b/w 300*dpi*	right polygons	blocks surrounded by straight white streams; fixed parameters	skew up to 15° is dealt; segmentation into rectangular blocks
Texture or Local Analysis	Jain Bhattacharjee [60]	g.l. 75-150*dpi*	freely shaped	user provided number of texture classes; selection of Gabor filters	multiple skews are admitted
	Tan *et. al* [61]	g.l. 300*dpi*	freely shaped		segmentation of text, graphics, and background areas
Background Analysis	Normand Viard-Gaudin [62]	b/w 150*dpi*	freely shaped	no automatic way for *relevant nodes* selection is provided	multiple skews are dealt; segmentation of text, large text, graphics areas
	Kise *et al.* [63]	b/w 90*dpi*	freely shaped	interblock gaps wider that interline gaps; fixed parameters; no wide word gaps	multiple skews are dealt; segmentation of text blocks, figures, separators

Table 5: Features of the page segmentation algorithms.

method	*reference*	*input type*	*layout*	*classes*	*assumptions and limitations*	*advantages, charactersitics*
Connected Component Analysis	Akiyama Hagita [31]	b/w 200*dpi*	rectangular blocks	headline text lines graphics separators	text majority; one text direction; hierarchic spacing protocol	skew up to 10° is dealt; English and Japanese documents are dealt
	Zlatopolsky [66]	b/w 300*dpi*	rectangular blocks	text lines, text blocks, separators, graphics blocks	interblock gaps wider that interline gaps; no textured pictures	multiple skews are dealt
	Wang Yagasaki [67]	b/w 50-75*dpi*	freely shaped	text blocks, text lines, pictures, tables, separators	text size in (6,72) points range	dynamic parameters estimation; multiple skews are dealt
	Simon *et al.* [68]	b/w	freely shaped	words, text lines, text and graphic blocks	cleaned docs; one text direction	skew up to 5° is tolerated; chemical docs
Smearing	Tsujimoto Asada [71]	b/w 300*dpi*	rectangular blocks	text lines, figures, graphics, tables, separators, noise	no skew	frequent touching chars are admitted
Texture or Local Analysis	Scherl *et al.* [73]	g.l.	freely shaped	text, pictures (graphics with the Fourier method)	spacing constraints	multiple skews may be present
	Sauvola Pietikäinen [74]	b/w	rectangular blocks	text, pictures, background	cleaned docs; no skew	
	Jain Zhong [75]	g.l. 100*dpi*	freely shaped	text and graphics, halftone, background	availability of a representative training set	skew may be present; robust to the alphabet
	Etemad *et al.* [76]	g.l. 200-300*dpi*	freely shaped	text, images, graphics	availability of a representative training set	skew may be present; text in images

Table 6: Features of the segmentation/classification mixed algorithms.

method	*reference*	*input type*	*classes*	*assumptions and limitations*	*advantages, charactersitics*
Linear Discriminant Classifiers	Wong *et al.* [53]	b/w 240*dpi*	text lines, graphics and halftones, horizontal lines, vertical lines	no skew; known chars size; problems with close or high text lines	
	Shih Chen [77]	b/w 300*dpi*	text lines, horizontal lines, vertical lines, graphics, pictures	no skew	independence of char sizes and resolution
	Wang Srihari [55]	b/w 100-200*dpi*	small, medium, large text, graphics, halftones	no skew	trainable; independence of blocks size
	Pavlidis Zhou [59]	b/w 300*dpi*	text blocks, diagrams, halftones	fixed parameters	
Binary Classification Tree	Sivaramaakrishnan *et al.* [78]	b/w 300*dpi*	normal, large text, math, tables, halftones, drawings, ruling, logo, others	rectangular zones organized into columns	trainable
Neural Networks	Le *et al.* [79]	b/w 200*dpi*	text, non text blocks	no skew; only text/non-text block classification	trainable

Table 7: Features of the algorithms for block classification.

to rearrange the matches of S_o in the proper order is calculated as follows. The N matched substrings in S_c are labeled in increasing order with integers from 1 to N. As a consequence the labels of the substrings in the generated text S_o, are a permutation of the integers from 1 to N. The purpose is to find a sequence of moves that transforms this permutation into the identity. After each move, any pair of adjacent substrings with consecutive indices are merged. This requires a relabeling of the substrings and reduces the permutation size. At each step the algorithm attempts to find the move that provides the greatest reduction in the size of the permutation. Whenever two or more moves yield the same reduction, the one moving the smallest number of characters is selected.

Two cost functions are defined to evaluate the output of the page decomposition module. The first function, called *Cost*, calculates the cost of correcting all types of errors generated from automatically segmented pages. A move operation can be performed either by *cut and paste* of a block of text (string) or by a sequence of *delete and re-type*. The choice depends on the length of the string. It is assumed that the cost of a move operation is independent of the moving distance, and of the string length n when it is greater than a threshold T. The *Cost* function is defined as follows:

$$\begin{aligned} &Cost(S_o, S_c, W_i, W_d, W_m) \\ &\quad = W_i I + W_d D + W_m M. \end{aligned}$$

where W_i, W_d and W_m are the costs associated with an insertion, deletion and move, respectively. The threshols T is set equal to $W_m/W_i + W_d$. If $n < T$, delete and re-type is preferred, otherwise cut and paste. The second function, called *Calibrated Cost*, calculates only the cost of correcting the zoning errors. It is introduced to eliminate the effects of recognition errors. It is assumed that OCR systems make the same recognition errors when a page is automatically or manually zoned. Let S_m be the result of the manual page segmentation, then the *Calibrated Cost* function is defined as follows:

$$\begin{aligned} &Calibrated\_Cost(S_o, S_m, S_c, W_i, W_d, W_m) = \\ &\quad Cost(S_o, S_c, W_i, W_d, W_m) - \\ &\quad Cost(S_m, S_c, W_i, W_d, W_m) \end{aligned}$$

5.2 The region-based approach

Region-based page segmentation benchmarking environments are proposed by Yanikoglu and Vincent [92, 93], and Haralick *et al.* [94, 95]. The quality of

the page decomposition is assessed by comparing the segmentation output, described as a set of regions, to the corresponding ground truth. The major difficulty for these approaches is the definition of a distance measure between two sets of regions: it has to encompass and to balance several elements such as correspondence between regions, overlap degree between regions and presence of unmatched regions in the two sets.

In the evaluation environment proposed by Yanikoglu and Vincent, named Pink Panther, a region is represented as a polygon together with various attributes. By analyzing matching between ground truth and segmented regions, errors like missing, false detection, merging, and splitting are detected and scored. The detailed shape of regions is not taken into account in the matching, so that small irrelevant differences between ground truth and segmented polygons are ignored. The overall page decomposition quality is computed as the normalized weighted sum of the individual scores.

Haralick *et al.* propose a quality measure which is based on the overlapping between rectangular regions coming, respectively, from the ground truth and the page decomposition procedure. Given two sets of rectangular regions, $\mathcal{G} = \{G_1, G_2, \ldots, G_M\}$ for ground truthed regions and $\mathcal{D} = \{D_1, D_2, \ldots, D_N\}$ for detected regions, two matrices are computed as follows:

$$\sigma_{ij} = \frac{Area(G_i \cap D_j)}{Area(G_i)} \quad and \quad \tau_{ij} = \frac{Area(G_i \cap D_j)}{Area(D_j)}$$

where $1 \leq i \leq M$, $1 \leq j \leq N$. Correct matchings and errors, *i.e.* missing, false detection, merging and splitting, are detected by analyzing the matrices $\Sigma = (\sigma_{ij})$ and $T = (\tau_{ij})$. The overall goodness function is defined as a weighted sum of these quantities.

6 A critical discussion

Current systems for document transformation from paper to electronic format present several limitations. They are tailored to suit specific applications and it is difficult to adjust a system for an application different from the original one. Documents often contain, along with relevant pictorial information, extraneous elements, *e.g.* noise or background pattern, that, if not detected and removed, could disturb the text features extraction and compromise the whole document understanding process. Non uniform backgrounds, which are present in documents like newspaper and modern magazine pages, has been almost ignored in the reviewed papers. Another

typical weakness is the fact that the analysis of the behavior of the proposed techniques applied to degraded documents have not been reported, although they are often present in practical applications or may represent the exclusive input when translating old paper archives into an electronic format. Furthermore, we observed that some types of regions are disregarded in the page decomposition phase: mathematical equations, chemical formulae, tables (distinguishing tables from multi-columned text is a critical task).

We noticed that several authors report experimental results which are hardly comparable. Experiments are characterized by different significance levels, depending on the size of the adopted sample, on its diffusion in the research community, and its representativeness of the documents involved in the specific application. Benchmarking of layout analysis algorithms is an important topic to which only recently efforts have been devoted, *e.g.*, for page decomposition evaluation and for databases creation.

Finally, the employment in real applications of many among the proposed layout analysis methods, is made unfeasible by their high computational cost.

Creative layouts, non uniform background, and text with different directions, are becoming frequent in the documents world for a variety of reasons: to obtain an appealing visual look (newspapers, magazine and book covers), to convey information at a glance (advertisements) or to avoid falsification (checks). Future document image analysis systems have to cope with these new features. Among the requirements they have to satisfy we mention:

- flexibility with respect to various applications;
- computational efficiency;
- automatic management of document with:
  - non uniform background;
  - presence of gray level or color information;
  - page structured with complex layouts (non rectangular or even freely shaped blocks);
  - text with very different sizes, styles and fonts on the same page;
  - text regions with different orientations in the same page;
  - text and inverse text in the same page or, more generally, text and background with different colors in the same page;

  - text embedded in graphics or figures;
  - text lines with curved baseline;

- robustness with respect to degraded input documents;
- validated performance on large, significant, commonly accepted databases in which are included binary, gray-level, color images of documents with various type of layout structure

Therefore, in our opinion, future document analysis research, will be able to improve the current products if the efforts will be focused on the above mentioned topics.

7 Logical layout analysis

Purpose of logical layout analysis is to assign a meaningful label to the homogeneous regions (blocks) of a document image and to determine their logical relationships, typically with respect to an a priori description of the document, *i.e.* a *model*.

The objects related to the logical layout encode document regions that humans perceive as meaningful with respect to the content. For example: title, abstract, paragraph, section, table, figure and footnote are possible logical objects for technical papers, while: sender, receiver, date, body and signature emerge in letters. This highlights how much application dependent is the definition of logical objects. Relationships among different objects are also possible. A typical organization is a hierarchy of objects depending on the specific context. Other examples of relations are the cross reference of a caption to a figure or the reading order of some parts of the document. Another interesting class of relations are those existing between the logical objects and the geometric blocks: they can be exploited to help the whole understanding process.

Recognizing the logical structure of a document can be performed only on the basis of some kind of a priori information (the *knowledge*) which can be represented in very different forms. Classical AI approaches (like blackboard based control systems) as well as probabilistic techniques (extension of the Hidden Markov Models paradigm) have been used. In any case a common problem is the level of detail the knowledge has to reach: a generic knowledge about the typical spatial and geometric features of the various elements in a document may be sufficient to separate homogeneous regions. On the

other hand assigning roles to the different textual regions often requires a more specific knowledge on the class of documents. Although specific knowledge favours precise recognition, it restricts the domain on which it can be applied. Moreover, *ad hoc* solutions like heuristics cannot be exported in different contexts. As a matter of fact, the most effective systems on the market work in restricted domains where the available knowledge is precise, for example mail address reading systems or form processing for banking applications. A general purpose document analysis system has to rely upon rather generic knowledge about typesetting rules and the way logical information can be deduced from them. As an alternative, systems can be endowed with the capability to distinguish among different document categories, each one described by its own model with detailed and specific knowledge.

Related to the previous problems is the item of flexibility. In many of the proposed systems the knowledge is provided by a human expert of the application domain. Therefore the analysis will succeed only for those documents having a previously defined structure and will fail when a new kind of document has to be recognized. Alternative approaches aim at automating the process of supplying knowledge, by means of appropriate learning algorithms, so that no a priori information on the specific format of a document should be provided by the user.

In the past, logical layout analysis was often considered a stand-alone process, well separated from the geometric layout analysis. This view is changing. For example the constraint that the geometric layout analysis must precede the logical one during the understanding process has been relaxed: Krishnamoorthy *et al.* [57] have proposed an approach in which segmentation and classification are claimed to be performed simultaneously. Moreover, the use of knowledge is not limited to logical layout analysis; as highlighted in Section 4, the a priori information can be utilized to guide or optimize the geometric layout analysis. The problem is that often it takes the form of implicit assumptions and heuristics, specific for particular documents and difficult to export in different contexts. The same remarks previously discussed about the use of the knowledge in logical layout analysis hold for geometric layout analysis.

The rest of the section is divided in two parts: section 7.1 presents the standard formats used for the definition and the interchange of documents in the perspective of the DIU. Section 7.2 briefly reports a review of the most interesting techniques used for the logical layout analysis, tentatively grouped into homogeneous topics.

7.1 Standards for document format

Standards to encode the format of documents have been introduced in order to allow an easy interchange of electronic documents. They have been developed without taking into account the problems concerning the processing and understanding of document images. Nevertheless, their relevance for the DIU community is deep not only for compatibility issues but mainly for the structured representations and the notion of *abstract document architecture* that some of them have defined. In this respect having some knowledge about the standards for document format is useful to understand recent works of the DIU that have been influenced and inspired by them, in particular for the representation and the organization of the geometric and logical objects.

The problem of exchanging electronic documents between different hardware and software platforms arose concurrently with the quick and wide diffusion of computers, giving strength to the process of the definition of common formats. Several formats were defined by private companies, for example the Microsoft's RTF (Rich Text Format), some of which has become very popular and *de facto* standard, like Adobe's PostScript [98]. In order to overcome the limitations of the private formats, the international community has defined official standards such as SGML (Standard Generalized Markup Language, ISO 8879:1986, [99]) and ODA (Open Document Architecture, ISO 8613:1989, [100, 101]).

SGML comes from the USA community and was developed for highly complex documents from the publishing environment and many products use it. ODA comes from Europe and was initially developed for simple documents from the office environment; its diffusion in the products on the marketplace is very low. SGML is a simpler standard than ODA since it is a language for describing only the logical structure of a document with respect to a predefined class; SGML documents need external information about the formatting issues and the document class of reference. The generality and complexity of ODA come from the fact that it provides the means to define and manage the logical layout as well as the geometric layout and the presentation issues; ODA documents are therefore self-describing. SGML is markup-oriented and can be used with the current editors, while ODA is object-oriented and requires a new generation of editors. As far as the interchange of documents is concerned, an important difference between SGML and ODA is that the latter allows blind communication: the receiver has to know nothing more than it receives an ODA document; with SGML

the sender and receiver have to agree on the formatting specifications and on the class of documents that is used.

In some detail, SGML is a language which uses *markups* (additional annotations to the content) to define the logical elements of the documents. The set of available markups changes depending on the specific class of documents, which is encoded in a DTD (Document Type Definition). The DTD defines a grammar that indicates the type of allowed markups and how markups are distinguished from the text content. A SGML document consists of three parts: (1) the *DTD*, (2) the *SGML declaration* defining the characters used in the DTD and in the text, (3) the *Document instance* containing the text of the document, including a reference to the DTD. It is worth noting that the three parts correspond physically to three different files.

ODA has been defined to be as open as possible, resulting in a huge amount of features supported. An ODA document, physically a single file, consists of six parts: (1) the *Logical View* encoding the specific logical structure of the document, (2) the *Layout View* encoding the specific geometric structure, (3) the *Logical and Layout Generic Structures* being sets of rules that define the class of the document, (4) the *Content Information* containing text, geometric graphics and raster graphics, (5) the *Styles* subdivided in layout style and presentation style, (6) the *Document Profile* including attributes like author's name, date of creation, keywords, etc. The ODA format of a document contains a huge amount of information and attributes. However a specific application does not need all the available features; for this reason ODA provides a mechanism to restrict the features needed in a particular application to a specific subset called Document Application Profile (DAP). In this way documents are stored in less space. Two ODA applications conforming to the same DAP are guaranteed to work correctly. At the moment three DAPs have been proposed: DAP level 1 provides functionality for simple text-only documents, DAP level 2 for simple text and graphics documents, DAP level 3 for advanced text and graphics documents such those used in electronic publishing.

Although rather different, SGML and ODA share important features: first, the goals are the same, that is the interchange of compound documents between open systems through electronic data communication and the automated processing of documents based on their logical structures and not only on their contents. Second, both SGML and ODA are based on the concept of *document architecture*, that is an abstract model to represent the information contained in a document. This is an important feature since it

differentiates SGML and ODA from the formats that belong to the *pdl* (page description language) class, like Adobe's PostScript and the recent Adobe's PDF (Portable Document Format) [102]: these formats address the geometric issues concerning the presentation of a document, losing completely the logical information. No document architecture is defined by PDF which is useful in situations where documents are delivered in their final form.

As far as the relation between SGML/ODA and the DIU is concerned, their main influence comes from the concept of abstract document architecture and the structured representations that they have introduced. In this respect some authors have proposed to extend (or better to adapt) such standards in order to obtain additional functionalities specifically designed for the document understanding, for example the management of the uncertainty and alternative interpretations. This is the case of ΠODA (Paper interface to ODA) [103], a system that receives in input the image of an office document and attempts to reconstruct an electronic document in ODA format. Another interesting example is DAFS (Document Attribute Format Specification) [104] that is a file format specification for documents, explicitly designed for the DIU. DAFS is implemented as a special SGML application with specific additionals in order to manage the formatting features, the image storage and the portability between different DTDs. DAFS appears to be very promising for two reasons: first it is based on SGML, more popular and agile than ODA; second the solutions to overcome the SGML limitations have been guided by the DIU requirements.

7.2 Techniques for logical layout analysis

Tree transformation. Tree transformation assumes to work in a context similar to that proposed by the ODA standard, in which the geometric structure as well as the logical structure take the form of a tree. Logical layout analysis is therefore the process to build up the logical tree starting from the information gathered into the geometric tree. In Tsujimoto and Asada's system [71], tree transformation is performed with a set of deterministic rules, possibly repeated, which label the blocks and define their reading order. In experiments conducted on documents of different categories (magazine, journals, newspapers, books, manuals letter and scientific papers) the logical structure was correctly determined for 94 of documents out of 106. Dengel *et al.* [103] proposed the ΠODA system (see section 7.1) for office documents. Tree transformation is performed with a set of weighted rules. Backtracking on different hypotheses is employed during and after tree

transformation, then the most probable logical tree is validated by matching the text contents with the knowledge stored in vocabularies. A sort of tree transformation, based on the DAFS format (see section 7.1), is used in the system proposed by Haralick's group [105]. The logical layout analysis is performed after the text reading phase (via OCR).

Description language. In this approach the structure of a document and the rules to build it up are described by means of an *ad hoc* language, called *description language*. Its syntax may heavily vary from system to system and it may encompass both geometric and logical information. Schürmann *et al.* [106] have developed the language FRESCO (Frame Representation languagE for StruCtured dOcument) in an object oriented style, with classes of objects and relations among them; it can be seen as an extension of ODA (see section 7.1). The logical layout analysis is performed with a control strategy that allows different hypotheses and backtracking to be accomplished in a search space by means of the best first search algorithm A^*. Higashino *et al.* [107] have defined the FDL (Form Definition Language), a lisp-like language based on rectangular regions. A predicate in FDL encodes a rule for the layout in terms of identification of regions and relations among them. Evaluating a FDL program means to find the best matches between the predicates and the blocks extracted by means of a depth first search with backtracking. As reported in Haralick's review [2], Derrien-Peden [108] defines a *frame* based system to determine the logical structure of scientific and technical documents. Frames are structures based on properties and relations, well studied in the past by the Knowledge Representation discipline.

Blackboard system. A typical approach to the control of an AI system is the *blackboard* approach, very popular in the 80's. Briefly, the basic idea of a blackboard based system is to divide a complex problem into loosely coupled subtasks, each managed by a specialized procedure which encodes a particular piece of knowledge and updates a common data area, the blackboard. During the computation, each procedure is associated to a dynamic score that encodes its degree of applicability to the current context. A scheduler encodes the strategy to divide the problem into subtasks and to select the best procedure at each step, by using fixed or weighted IF-THEN rules. Advantages and drawbacks of the blackboard approach, widely discussed in the AI community, are outside the focus of this paper. Srihari *et al.* [109, 110]

developed a sophisticated blackboard system organized as a three level hierarchy in order to recognize address blocks on mail pieces. A mix of frames and rules is used to model the knowledge for selecting the procedure to be applied and for computing the plausibility degree of a particular labeling. A restricted version of a blackboard system is proposed by Yeah *et al.* [111] for the same task (address location on envelopes). The logical layout analysis is based on sequential rules with heuristics applied to the segmented blocks: finding no acceptable labelings determines a new page segmentation with different thresholds. In this way logical and geometric layout analysis affect each other.

Syntactic approach. In this approach the knowledge required to segment the page in blocks and to label them is represented by means of formal (typically context free) grammars; geometric and logical layout analysis are therefore performed with programs for the syntactic analysis (*e.g.* a parser), obtained from the formal grammars. Nagy *et al.* [57, 56] have developed a syntactic based system working with technical journals. They define a set of appropriate context free grammars, each defining rules to aggregate pixels into more and more structured entities, till up to the logical objects. From the grammars, programs for the syntactic analysis (parsers) are automatically obtained: they are then used to perform segmentation and labeling in the same phase (simultaneously). A set of alternative grammars is used to allow different document structures (hypotheses) to be extracted and checked. A branch-and-bound algorithm searches for the best hypothesis. The criterion to optimize and prune the search is based on the cumulative area of the labelled blocks: the higher the area, the better the labeling hypothesis.

Hidden Markov Models. Probabilities have been hardly mentioned in the above approaches which tend to use deterministic heuristics over search spaces. A completely different technique based on an extension of HMMs (Hidden Markov Models) has been proposed by Kopec and Chou [112] for telephone yellow page text extraction. Extending HMMs to manage two-dimensional image regions is not straightforward and poses theoretical as well as computational problems. In the Kopec and Chou's approach the basic feature on which the (extended) HMMs works is the *glyph*, a portion of image representing a single character. This solution allows a very high recognition rate but the required time is huge. In a second paper [113] Kam

and Kopec present *ad hoc* heuristics to cut down the computational time, obtaining significant speedups.

Learning. Systems in which the knowledge is fixed once and for all, can fail with documents that do not belong to the domain for which the knowledge has been provided. Learning techniques aiming at an automation of the process of supplying knowledge appear to be a good solution towards greater generality and flexibility. A particularly interesting system has been developed by Esposito *et al.* [114]. A big relevance is given to the problem of the automatic *document classification* intended as the process of identifying the category of an input image document. They approach the classification task by means of machine learning techniques in order to directly acquire classification rules from a set of training documents. Even the logical layout analysis is approached as (another) supervised learning problem: the system has to learn the mapping between the features extracted from the document image and the logical structure of the document. When processing an input document, the geometric layout analysis extracts the blocks and some relevant features which allow the classification subsystem to identify the membership category of the document; this information greatly simplifies the logical layout analysis which uses the learned rules specific for the particular category to reconstruct the logical structure.

Interactive system. Some researchers have suggested to approach the DIU by allowing the humans to interact with the system in order to exploit their ability to solve ambiguous situations and to make decisions. For example the CIDRE (Cooperative and Interactive Document Reverse Engineering) project at the Fribourg University (Switzerland) is aimed at developing an interactive document understanding system running in three different and complex scenarios [115]: (1) construction of a structured bibliography (2) building an on-line help facility and (3) mail distribution in office automation. As far as the logical layout analysis is concerned the basic idea is to present the results or ambiguous situations to the user who selects the best operation by means of appropriate graphical tools.

8 Conclusions

Document layout analysis is a process that aims at detecting and classifying zones on a document image in various data types and at representing them

in a logical structure. Techniques for the layout analysis devoted to the understanding of structured pages of machine-printed documents (such as technical journals, magazines, newspapers, handbooks, business letters,...) have been reviewed in this paper. We reported popular and/or promising algorithms proposed in the DIU field, grouped by objective and adopted technique. Summing up tables permit to compare easily the characteristics of the various algorithms devoted to skew detection and page decomposition.

Our bibliographical analysis points out (Section 6) that todays DIU systems succeed only in limited domains. Improvements of current systems can be achieved by working on the management of complex documents and on the flexibility of the systems. An important tool that can be bettered, is the development of environments for the comparison of the document understanding systems, particularly in the case of complex documents.

Acknowledgements. The authors wish to thank Francesco Fignoni and Fausto Giunchiglia for the fruitful discussions and comments about the content and the structure of this review.

References

[1] S.V. Rice, F.R. Jenkins, and T.A. Nartker. The Fifth Annual Test of OCR Accuracy. Technical Report TR-96-01, Information Science Research Institute, University of Nevada, Las Vegas, April 1996.

[2] R.M. Haralick. Document Image Understanding: Geometric and Logical Layout. In *IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR)*, pages 385–390, Seattle, Washington, 1994.

[3] Y.Y. Tang, S.W. Lee, and C.Y. Suen. Automatic Document Processing: a Survey. *Pattern Recognition*, 29(12):1931–1952, 1996.

[4] A.K. Jain and B. Yu. Document Representation and Its Application to Page Decomposition. Technical Report MSUCPS:TR96-63, Michigan State University, East Lansing, MI, December 1996.

[5] L. O'Gorman and R. Kasturi. *Document Image Analysis*. IEEE Computer Society Press, Los Alamitos, CA, 1995.

[6] F. Fignoni, S. Messelodi, and C.M. Modena. Review of the State of the Art in Optical Character Recognition. Part 1: Machine Printed Documents. Technical Report #9607-03, IRST, Trento, Italy, June 1996.

[7] J.M. White and G.D. Rohrer. Image Thresholding for Optical Character Recognition and Other Apllications Requiring Character Image Extraction. *IBM Journal of Reserch and Development*, 27(4):400–411, July 1983.

[8] T. Taxt, P.J. Flynn, and A.K. Jain. Segmentation of Document Images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 11(12):1322–1329, December 1989.

[9] L. O'Gorman. Binarization and Multithresholding of Document Images Using Connectivity. *CVGIP: Graphical Models and Image Processing*, 56(6):494–506, 1994.

[10] H.-S. Don. A Noise Attribute Thresholding Method for Document Image Binarization. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 231–234, Montreal, Canada, August 1995.

[11] Y. Liu and S.N. Srihari. Document Image Binarization Based on Texture Features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):540–544, May 1997.

[12] J. Sauvola, T. Seppänen, S. Haapakoski, and M. Pietikäinen. Adaptive Document Binarization. In *Proc. of the 4th International Conference on Document Analysis and Recognition*, pages 147–152, Ulm, Germany, August 1997.

[13] P.W. Palumbo, P. Swaminathan, and S.N. Srihari. Document image binarization: Evaluation of algorithms. In *Proc. of SPIE Symposium. Applications of Digital Image Processing IX*, volume 697, pages 278–285, San Diego, California, August 1986.

[14] O.D. Trier and T. Taxt. Evaluation of Binarization Methods for Document Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(3):312–315, March 1995.

[15] A.T. Abak, U. Baris, and B. Sankur. The Performance Evaluation of Thresholding Algorithms for Optical Character Recognition. In *Proc. of the 4th International Conference on Document Analysis and Recognition*, pages 697–700, Ulm, Germany, August 1997.

[16] W. Postl. Detection of linear oblique structures and skew scan in digitized documents. In *Proc. of the 8th International Conference on Pattern Recognition*, pages 687–689, Paris, France, 1986.

[17] H.S. Baird. The skew angle of printed documents. In *Proc. of the Conference Society of Photographic Scientists and Engineers*, volume 40, pages 21–24, Rochester, NY, May, 20-21 1987.

[18] G. Ciardiello, G. Scafuro, M.T. Degrandi, M.R. Spada, and M.P. Roccotelli. An experimental system for office document handling and text recognition. In *Proc. of the 9th International Conference on Pattern Recognition*, volume 2, pages 739–743, Roma, Italy, November, 14-17 1988.

[19] Y. Ishitani. Document Skew Detection Based on Local Region Complexity. In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pages 49–52, Tsukuba, Japan, October 1993. IEEE Computer Society.

[20] A. Bagdanov and J. Kanai. Projection Profile Based Skew Estimation Algorithm for JBIG Compressed Images. In *Proc. of the 4th International Conference on Document Analysis and Recognition*, pages 401–405, Ulm, Germany, August 1997.

[21] S.N. Srihari and V. Govindaraju. Analysis of Textual Images Using the Hough Transform. *Machine Vision and Applications*, 2(3):141–153, 1989.

[22] S. Hinds, J. Fisher, and D. D'Amato. A document skew detection method using run-length encoding and the Hough transform. In *Proc. of the 10th International Conference on Pattern Recognition*, pages 464–468, Atlantic City, NJ, June, 17-21 1990.

[23] A.L. Spitz. Skew Determination in CCITT Group 4 Compressed Document Images. In *Proc. of the Symposium on Document Analysis and Information Retrieval*, pages 11–25, Las Vegas, 1992.

[24] D.S. Le, G.R. Thoma, and H. Wechsler. Automated Page Orientation and Skew Angle Detection for Binary Document Images. *Pattern Recognition*, 27(10):1325–1344, 1994.

[25] Y. Min, S.-B. Cho, and Y. Lee. A Data Reduction Method for Efficient Document Skew Estimation Based on Hough Transformation. In *Proc. of the 13th International Conference on Pattern Recognition*, pages 732–736, Vienna, Austria, August 1996. IEEE Press.

[26] U. Pal and B.B. Chaudhuri. An improved document skew angle estimation technique. *Pattern Recognition Letters*, 17(8):899–904, July 1996.

[27] B. Yu and A.K. Jain. A Robust and Fast Skew Detection Algorithm for Generic Documents. *Pattern Recognition*, 29(10):1599–1629, 1996.

[28] A. Hashizume, P.S. Yeh, and A. Rosenfeld. A method of detecting the orientation of aligned components. *Pattern Recognition Letters*, 4:125–132, 1986.

[29] L. O'Gorman. The Document Spectrum for Page Layout Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, 1993.

[30] R. Smith. A Simple and Efficient Skew Detection Algorithm via Text Row Accumulation. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 1145–1148, Montreal, Canada, August 1995.

[31] T. Akiyama and N. Hagita. Automated Entry System for Printed Documents. *Pattern Recognition*, 23(11):1141–1154, 1990.

[32] H. Yan. Skew Correction of Document Images Using Interline Cross-Correlation. *CVGIP: Graphical Models and Image Processing*, 55(6):538–543, November 1993.

[33] B. Gatos, N. Papamarkos, and C. Chamzas. Skew Detection and Text Line Position Determination in Digitized Documents. *Pattern Recognition*, 30(9):1505–1519, 1997.

[34] J. Sauvola and M. Pietikäinen. Skew Angle Detection Using Texture Direction Analysis. In *Proc. of the 9th Scandinavian Conference on Image Analysis*, pages 1099–1106, Uppsala, Sweden, June 1995.

[35] C. Sun and D. Si. Skew and Slant Correction for Document Images Using Gradient Direction. In *Proc. of the 4th International Conference on Document Analysis and Recognition*, pages 142–146, Ulm, Germany, August 1997.

[36] S. Chen and R.M. Haralick. An Automatic Algorithm for Text Skew Estimation in Document Images Using Recursive Morphological Transforms. In *Proc. of the first IEEE International Conference on Image Processing*, pages 139–143, Austin, Texas, 1994.

[37] H. K. Aghajan, B. H. Khalaj, and T. Kailath. Estimation of skew angle in text-image analysis by *SLIDE*: subspace-based line detection. *Machine Vision and Applications*, 7:267–276, 1994.

[38] H.S. Baird. Anatomy of a Versatile Page Reader. *Proc. of the IEEE*, 80(7):1059–1065, 1992.

[39] P.V.C. Hough. Methods and means for recognizing complex patterns. US Patent #3,069,654, December 18, 1962.

[40] E.R. Davies. *Machine Vision: Theory, Algorithms, Practicalities*. Academic Press, 1992.

[41] S. Chen and R.M. Haralick. Recursive Erosion, Dilation, Opening and Closing Transforms. *IEEE Transaction on Image Processing*, 4(3):335–345, March 1995.

[42] A.P. Dias. Minimum Spanning Trees for Text Segmentation. In *Proc. of Fifth Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, 1996.

[43] H.S. Baird. Global-to-Local Layout Analysis. In *Proc. of the IAPR Workshop on Syntactic and Structural Pattern Recognition*, pages 136–147, Pont-a-Mousson, France, September 1988.

[44] H.S. Baird, S.E. Jones, and S.J. Fortune. Image Segmentation using Shape-Directed Covers. In *Proc. of the 10th International Conference on Pattern Recognition*, Atlantic City, NJ, June, 17-21 1990.

[45] H.S. Baird. Background Structure in Document Images. In *Advances in Structural and Syntactic Pattern Recognition*, pages 253–269. World Scientific, Singapore, 1992.

[46] J. Ha, R.M. Haralick, and I.T. Phillips. Document Page Decomposition by the Bounding-Box Projection Technique. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 1119–1122, Montreal, Canada, August 1995.

[47] S. Chen, R.M. Haralick, and I.T. Phillips. Extraction of Text Layout Structures on Document Images based on Statistical Characterization. In *IS&T/SPIE Symposium on Electronic Imaging Science and Technology, Document Recognition II*, pages 128–139, San Jose', USA, 1995.

[48] E.G. Johnston. SHORT NOTE: Printed Text Discrimination. *Computer Graphics and Image Processing*, 3:83–89, 1974.

[49] F. Hönes and J. Lichter. Layout extraction of mixed mode documents. *Machine Vision and Applications*, 7:237–246, 1994.

[50] O. Déforges and D. Barba. Segmentation of Complex Documents Multilevel Images: a Robust and Fast Text Bodies-Headers Detection and Extraction Scheme. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 770–773, Montreal, Canada, August 1995.

[51] P. Parodi and G. Piccioli. An Efficient Pre-Processing of Mixed-Content Document Images for OCR Systems. In *Proc. of the 13th International Conference on Pattern Recognition*, pages 778–782, Vienna, Austria, August 1996. IEEE Press.

[52] M.B.H. Ali, F. Fein, F. Hönes, T. Jäger, and A. Weigel. Document Analysis at DFKI. Part 1: Image Anlysis and Text Recognition. Technical Report RR-95-02, German Research Center for Artificial Intelligence (DKFI), Kaiserslautern, Germany, March 1995.

[53] K.J. Wong, R.G. Casey, and F.M. Wahl. Document Analysis System. *IBM Journal of Reserch and Development*, 26(6):647–656, 1982.

[54] G. Nagy and S.C. Seth. Hierarchical Representation of Optically Scanned Documents. In *Proc. of the 7th International Conference on Pattern Recognition*, pages 347–349, Montreal, Canada, 1984.

[55] D. Wang and S.N. Srihari. Classification of Newspaper Image Blocks Using Texture Analysis. *Computer Vision, Graphics and Image Processing*, 47:327–352, 1989.

[56] G. Nagy, S. Seth, and M. Viswanathan. A Prototype Document Image Analysis System for Technical Journals. *Computer*, 25(7):10–22, 1992.

[57] M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan. Syntactic Segmentation and Labeling of Digitized Pages from Technical Journals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(7):737–747, 1993.

[58] D. Sylwester and S. Seth. A Trainable, Single-Pass Algorithm for Column Segmentation. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 615–618, Montreal, Canada, August 1995.

[59] T. Pavlidis and J. Zhou. Page Segmentation and Classification. *CVGIP: Graphical Models and Image Processing*, 54(6):484–496, 1992.

[60] A.K. Jain and S. Bhattacharjee. Text Segmentation using Gabor filters for automatic document processing. *Machine Vision and Applications*, 5(3):169–184, 1992.

[61] Y.Y. Tang, H. Ma, X. Mao, D. Liu, and C.Y. Suen. A New Approach to Document Analysis Based on Modified Fractal Signature. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 567–570, Montreal, Canada, August 1995.

[62] N. Normand and C. Viard-Gaudin. A Background Based Adaptive Page Segmentation Algorithm. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 138–141, Montreal, Canada, August 1995.

[63] K. Kise, O. Yanagida, and S. Takamatsu. Page Segmentation Based on Thinning of Background. In *Proc. of the 13th International Conference on Pattern Recognition*, pages 788–792, Vienna, Austria, August 1996. IEEE Press.

[64] O.T. Akindele and A. Belaid. Page Segmentation by Segment Tracing. In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pages 341–344, Tsukuba, Japan, October 1993. IEEE Computer Society.

[65] L.A. Fletcher and R. Kasturi. A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):910–918, 1988.

[66] A.A. Zlatopolsky. Automated document segmentation. *Pattern Recognition Letters*, 15(7):699–704, July 1994.

[67] S.-Y. Wang and T. Yagasaki. Block Selection: A Method for Segmenting Page Image of Various Editing Styles. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 128–133, Montreal, Canada, August 1995.

[68] A. Simon, J.-C. Pret, and A.P. Johnson. A Fast Algorithm for Bottom-Up Document Layout Analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):273–277, 1997.

[69] T. Saitoh and T. Pavlidis. Page Segmentation without Rectangle Assumption. In *Proc. of the 11th International Conference on Pattern Recognition*, pages 277–280, The Hague, 1992.

[70] Y. Hirayama. A Block Segmentation Method For Document Images with Complicated Column Structures. In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pages 91–94, Tsukuba, Japan, October 1993. IEEE Computer Society.

[71] S. Tsujimoto and H. Asada. Major components of a Complete Text Reading System. *Proceedings of the IEEE*, 80(7):1133–1149, 1992.

[72] F. Lebourgeois, Z. Bublinski, and H. Emptoz. A Fast and Efficient Method For Extracting Text Paragraphs and Graphics from Unconstrained Documents. In *Proc. of the 11th International Conference on Pattern Recognition*, pages 272–276, The Hague, 1992.

[73] W. Scherl, F. Wahl, and H. Fuchsberger. Automatic Separation of Text, Graphic and Picture Segments in Printed Material. In E.S. Gelsema and L.N. Kanal, editors, *"Pattern Recognition in Practice"*, pages 213–221. North-Holland, Amsterdam, 1980.

[74] J. Sauvola and M. Pietikäinen. Page Segmentation and Classification using fast Feature Extraction and Connectivity Analysis. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 1127–1131, Montreal, Canada, August 1995.

[75] A.K. Jain and Y. Zhong. Page Layout Segmentation based on Texture Analysis. *Pattern Recognition*, 29(5):743–770, 1996.

[76] K. Etemad, D.S. Doermann, and R. Chellappa. Multiscale Segmentation of Unstructured Document Pages Using Soft Decision Integration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(1):92–96, January 1997.

[77] F.Y. Shih and S.S. Chen. Adaptive Document Block Segmentation and Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(5):797–802, 1996.

[78] R. Sivaramaakrishnan, I.T. Phillips, J. Ha, S. Subramanium, and R.M. Haralick. Zone Classification in a Document using the Method of Feature Vector Generation. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 541–544, Montreal, Canada, 1995.

[79] D.X. Le, G.R. Thoma, and H. Wechsler. Classification of Binary Document Images into Textual or Nontextual Data Blocks using Neural Network Models. *Machine Vision and Applications*, 8:289–304, 1995.

[80] O. Deforges and D. Barba. A Fast Multiresolution Text-line and Non Text-line Structure Extraction and Discrimination Scheme for Document Image Analysis. In *Proc. of the International Conference on Image Processing*, pages 134–138, Austin, Texas, 1994.

[81] D. Ittner. Automatic Inference of Textline Orientation. In *Proc. of the Second Annual Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, 1993.

[82] J. Fisher, S. Hinds, and D. D'Amato. A Rule-Based System For Document Image Segmentation. In *Proc. of the 10th International Conference on Pattern Recognition*, pages 567–572, Atlantic City, NJ, June, 17-21 1990.

[83] N. Amamoto, S. Torigoe, and Y. Hirogaki. Block segmentation and Text Area Extraction of Vertically/Horizontally Written Document. In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pages 341–344, Tsukuba, October 1993. IEEE Computer Society.

[84] G. Nagy, S.C. Seth, and S.D. Stoddard. Document analysis with an expert system. In *Proceedings Pattern Recognition in Practice II*, Amsterdam, The Neaderlands, June, 19-21 1985.

[85] K.S. Fu. *Applications of Pattern Recognition*. CRC Press, Boca Raton, FL, 1982.

[86] I.T. Phillips, S. Chen, and R.M. Haralick. English Document Database Standard. In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pages 478–483, Tsukuba, Japan, October 1993. IEEE Computer Society.

[87] G. Nagy. At the Frontiers of OCR. *Proceedings of the IEEE*, 80(7):1093–1100, 1992.

[88] J. Kanai, S.V. Rice, and T.A. Nartker. A Preliminary Evaluation of Automatic Zoning. Technical Report TR-93-02, Information Science Research Institute, University of Nevada, Las Vegas, April 1993.

[89] J. Kanai, T.A. Nartker, S.V. Rice, and G. Nagy. Performance Metrics for Document Understanding Systems. In *Proc. of the 2nd International Conference on Document Analysis and Recognition*, pages 424–427, Tsukuba, Japan, October 1993. IEEE Computer Society.

[90] J. Kanai, S.V. Rice, T. Nartker, and G. Nagy. Automated Evaluation of OCR Zoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(1):86–90, January 1995.

[91] S. Randriamasy, L. Vincent, and B. Wittner. An Automatic Benchmarking Scheme for Page Segmentation. In L. Vincent and T. Pavlidis, editors, *SPIE/SPSE Document Recognition*, volume 2181, San Jose', CA, February 1994. SPIE.

[92] B.A. Yanikoglu and L. Vincent. Ground-truthing and Benchmarking Document Page Segmentation. In *Proc. of the 3th International Conference on Document Analysis and Recognition*, pages 601–604, Montreal, Canada, August 1995.

[93] B.A. Yanikoglu and L. Vincent. Pink Panther: A Complete Environment for Ground-truthing and Benchmarking Document Page Segmentation. Technical report, Xerox Desktop Document Systems, February 1996.

[94] J. Ha, I.T. Phillips, and R.M. Haralick. Document Page Decomposition Using Bounding Boxes of Connected Components of Black Pixels. In *IS&T/SPIE Symposium on Electronic Imaging Science and*

Technology, Document Recognition II, pages 140–151, San Jose', USA, 1995.

[95] J. Liang, R. Rogers, R.M. Haralick, and I.T. Phillips. UW-ISL Document Image Analysis Toolbox: An Experimental Environment. In *Proc. of the 4th International Conference on Document Analysis and Recognition*, pages 984–988, Ulm, Germany, August 1997.

[96] E. Ukkonen. Algorithms for approximate string matching. *Information and Control*, 64:100–118, 1985.

[97] S. Latifi. How can permutations be used in the evaluation of zoning algorithms? *International Journal of Pattern Recognition and Artificial Intelligence*, 10(3):223–237, 1996.

[98] Adobe Systems Inc. *PostScript Language Reference Manual, 2nd edition*, December 1990.

[99] J.M. Smith and R.S. Stutely. *SGML: The Users' Guide to ISO 8879*. Chichester/New York: Ellis Horwood/Halsted, 1988.

[100] W. Horak. Office document architecture and office document interchange formats: current status of international standardization. *IEEE Computer*, 18(10):50–60, October 1985.

[101] P. Mancino. Can the Open Document Architecture (ODA) Standard Change the World of Information Technology? A study of the Documentation Standard Open Document Architecture (ODA, ISO 8613) for Information Technology. Master's thesis, Politecnico di Torino, Rijen, The Netherlands/Stockholm, Sweden: Ericsson Telecom, September 1994.

[102] Adobe Systems Inc. *Portable Document Format Reference Manual*, November 1996.

[103] A. Dengel, R. Bleisinger, R. Hoch, F. Fein, and F. Hoenes. From Paper to Office Document Standard Representation. *Computer*, 25(7):63–67, July 1992.

[104] RAF Technology Inc. *DAFS: Document Attribute Format Specification*, 1995.

[105] J. Liang, R. Rogers, B. Chanda, I.T. Phillips, and R.M. Haralick. From Image to Desktop Publishing: a Complete Document Understanding System. Working paper, 1995.

[106] J. Schürmann, N. Bartneck, T. Bayer, J. Franke, E. Mandler, and M. Oberländer. Document Analysis - From Pixels to Contents. *Proceedings of the IEEE*, 80(7):1101–1119, 1992.

[107] J. Higashino, H. Fujisawa, Y. Nakano, and M. Ejiri. A knowledge-based segmentation method for document understanding. In *Proc. of the 8th International Conference on Pattern Recognition*, pages 745–748, Paris, France, 1986.

[108] D. Derrien-Peden. Frame-based System for Macro-typographical Structure Analysis in Scientific Papers. In *Proc. of the 1st International Conference on Document Analysis and Recognition*, pages 311–319, France, September 1991.

[109] S.N. Srihari, C. Wang, P. Palumbo, and J. Hull. Recognizing Address Blocks on Mail Pieces: Specialized Tools and Problem-Solving Architecture. *AI Magazine*, 8(4):25–40, Winter 1987.

[110] C. Wang and S.N. Srihari. A Framework for Object Recognition in a Visual Complex Environment and its Application to Locating Address Blocks on Mail Pieces. *International Journal of Computer Vision*, 2:125–151, 1988.

[111] P.S. Yeh, S. Antoy, A. Litcher, and A. Rosenfeld. Address Location on Envelopes. *Pattern Recognition*, 20(2):213–227, 1987.

[112] G.E. Kopec and P.A. Chou. Document Image Decoding Using Markov Source Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):602–617, June 1994.

[113] A.C. Kam and G.E. Kopec. Document Image Decoding by Heuristic Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):945–950, September 1996.

[114] F. Esposito, D. Malerba, and G. Semeraro. Multistrategy Learning for Document Recognition. *Applied Artificial Intelligence*, 8(1):33–84, 1994.

[115] F. Bapst, R. Brugger, and R. Ingold. Towards an Interactive Document Structure Recognition System. Internal working paper, University of Fribourg (Switzerland), April 1995.