ELSEVIER

April 1995

Pattern Recognition Letters 16 (1995) 377–384

Pattern Recognition Letters

Representing pattern recognition-embedded systems through object-process diagrams: the case of the machine drawing understanding system

Dov Dori [*]

Faculty of Industrial Engineering, Technion, Israel Institute of Technology, Haifa 32000, Israel

Received 15 February 1994; revised 27 September 1994

Abstract

Pattern recognition involves a host of problems, algorithms and techniques dealing with all aspects of detection and classification of objects in scenes and their conversion into meaningful interpretations. Frequently, such tasks are embedded within vision systems that are themselves embedded within more complex systems. Concentrating on minute, albeit important, details of some pattern recognition algorithm, may result in blurring of the "big picture" that puts the algorithm under consideration in the more general framework. Pattern recognition (PR)-embedded systems feature a combination of complexity on the one hand and a balance between structure and behavior on the other hand. Analysis and understanding of such systems call therefore for a methodology that represents equally well structure and behavior within a unified frame of reference and has adequate tools for complexity management. This work proposes the object-process analysis (OPA) as an approach to tackle this task. The Machine Drawing Understanding System (MDUS) is as an instance of a PR-embedded system used as a case in point. We provide a motivation for the development of the system in general and its specialized Orthogonal Zig-Zag (OZZ) vectorization algorithm in particular. To demonstrate the suitability of OPA for representing PR-embedded systems at any level of detail, we apply it to communicate a top-down introduction of MDUS and its OZZ algorithm. The result is a series of consistent, inter-related object-process diagrams that gradually expose the details of the system. Complexity is managed through visibility control, which is obtained by a host of options for scaling object process diagrams. The ease of application of object-process analysis to the case in point suggests that it can be successfully applied to analyze, understand and communicate PR-embedded systems.

Keywords: Object-oriented analysis; Pattern recognition-embedded systems; System analysis and representation; Understanding engineering drawings

1. Introduction

The domain of machine vision involves a host of problems, algorithms and techniques dealing with all aspects of capture and conversion of scenes into meaningful interpretations. Frequently, pattern recognition subsystems are embedded within computer vision systems. The latter, in turn, may be subsystems in yet more complex systems, such as industrial inspection, autonomous navigation and robotics. Such PR-embedded systems feature two

[*] Email: dori@ie.technion.ac.il.

main characteristics: structure/behavior balance and complexity.

The structure/behavior balance characteristic is expressed by the fact that these systems have two main aspects: structural (static) and procedural (dynamic). The structural aspect pertains to the objects present in the systems and the long-term relations among them, such as aggregation (whole part) and generalization (gen-spec). The procedural aspect relates to the relatively short-term, time-varying behavior of the system, which is usually guided by algorithms, procedures or routines. Even though most of the behavior of a machine vision system can be described in terms of algorithms, the traditional tools for algorithm description, such as flow diagrams or structured pseudo-code, lack proper representation of the objects involved. Since most machine vision systems feature substantial structural and behavioral aspects, their thorough understanding requires a representation that strikes a balance between structure and behavior within a unified representation scheme. Objects and processes should have adequate weight, and the procedural relations that link objects to processes should be explicitly stated.

The complexity characteristic of PR-embedded systems implies that they have a large number of objects that are linked by processes at various abstraction levels. At the lowest level, objects are as simple as pixels, variables, parameters, etc., while processes can be averaging a pixel neighborhood or convolving a pixel. Intermediate processes are of the nature of vectorization, segmentation, skeletonization and edge detection. Accordingly, objects at this level are regions, edges, skeletons and other recognized objects on the way to high-level understanding. At the highest level, the objects are stationary or moving 3-D structures, decision about a robot's trajectory change for obstacle-avoidance, acceptance or rejection of an inspected product, etc. Frequently, the objects and the processes from different levels interact because of the inherent non-linear nature of the pattern recognition process. For example, many PR-embedded systems have feedback cycles that enable hypotheses from higher levels to be tested at lower levels.

To fully understand PR-embedded systems, to explain them, and to communicate their analysis results, this work proposes object-process analysis (OPA) as a methodology that extends object-oriented analysis to meet these challenges. OPA combines the representation of the structure and behavior of machine vision systems within an integrated framework and enables complexity management by scaling within this framework. Section 2 provides an overview of the evolution of approaches to systems analysis. Section 3 introduces the basic principles and definitions of OPA. The Machine Drawing Understanding System (MDUS) is presented in Section 4 as a case that demonstrates the OPA methodology. The example shows that OPA adequately represents the structure and behavior of a machine vision system, and that its scaling tools provide for a rational, top-down presentation of the system to any desired level of detail. To show the power of OPA in managing complexity, we scale up the lexical phase of MDUS in Section 5. Further up-scaling in Section 6 focuses on the Orthogonal Zig-Zag (OZZ) algorithm, which is the vectorization machine within the lexical phase.

2. System analysis paradigms: from processes to objects

Biological systems in general, and human beings in particular, respond to changes in structure, such as motion, change in color, texture, illumination or temperature, much more readily than to the structure itself. This is the case because such changes convey potential information about possible changes in the environment that might affect their existence or well-being. Because of their varying nature, processes are externally more visible than the objects that make them happen. Hence, they attract the attention of an observer, trying to understand and explain the surrounding reality. The attention of humans drawn by changes in things is a plausible explanation to the fact that first attempts in system modeling concentrated on the dynamics of the system. A notable example of these early approaches is the data flow diagram (DFD) method (De Marco, 1978), emphasizing processes as the major theme of the analysis. Only later approaches, e.g., entity-relationship diagrams (Chen, 1976) and object-orientation (OO) (Coad and Yourdon, 1991; Embley et al., 1992; Shlaer and Mellor, 1992; Rumbaugh et al., 1991; Nerson, 1993) put objects at the center of the analysis.

OO is the currently accepted paradigm for systems analysis, design and programming. It is built on the premise that every thing in any domain can be represented as an object. Processes (referred to as "methods" or "services") are encapsulated within objects and are activated via messages passed among objects. This "objectification" of the universe is adequate for describing the structure of a system, but it lacks appropriate tools to explicitly model the dynamic, procedural aspect of systems. To model the system's behavior, OO methodologies use flow charts (Coad and Yourdon, 1991), DFD or some variation thereof, such as Action DFD (Shlaer and Mellor, 1992), state charts (Rumbaugh et al., 1991), or a combination of these methods. These methods are generally not directly related to the object model. The lack of integration between structure and behavior is particularly significant in PR-embedded systems, because of their image processing component.

3. Object-process analysis: principles and definitions

Object-process analysis (OPA) (Dori et al., 1994; Dori, 1995) is a superset of OOA that has been designed to respond to the two main challenges of systems analysis discussed above: (1) integration between structure and behavior, and (2) provision of tools for complexity management. OPA is based on the description of how objects interact with each other through processes. This is the basis for the structure/behavior integration. Scaling serves as a complexity management tool, and the resulting object-process diagrams are amenable to a variety of scaling options. Scaling enables controlling the visibility and level of detail of objects and processes. Following are basic principles and definitions of the object-process analysis paradigm.

Objects and processes are the two fundamental things in the universe. The *universe of interest* is a subset of the universe that is relevant to the system under consideration. Its modeling is guided by two complimentary aspects: structural and procedural. The structural aspect explains what are the objects in the universe of interest that play a meaningful role and how they relate to each other in the long run. The procedural aspect explains how the objects in the universe of interest interact with each other through processes.

The universe of interest consists of a collection of *things*. Things are objects and processes, and may be simple or compound. Simple (atomic) things cannot be decomposed into things nor are they characterized by other things. Compound things consist of other things or are characterized by other things.

An *object* is a persistent thing in the universe of interest. A *process* is a transient thing in the universe of interest. It requires at least one object to enable it and it affects at least one object (possibly the enabling object).

A *feature* is a thing that characterizes a higher-level thing. A feature is related to the class which it characterizes through a *characterization relation.*

A *class* is the set of all the things that are characterized by the same set of features.

An *object (process) class* is the set of all the objects (processes) that are characterized by the same set of features.

An object (process) is a typical member of the object (process) class to which it belongs.

An *instance* is a member of a class.

Features are attributes and services. An *attribute* (*service*) is an object (process) class that characterizes a higher-level class.

Being an object class, an attribute has a number of instances (legal values). At any given point in time, the *state* of an object is the set of attribute values that are valid at that time.

As noted, complexity is managed through scaling. Each compound thing, which is called *seed* in the context of scaling, can be scaled up to yield an embedded object-process diagram, called *plant*, in which the inner structure and behavior of the seed is exposed. Conversely, a collection of things – the plant – can be scaled down to a seed, thereby obtaining a higher-level, more compact view of the system. Hence, the instances of the attribute *direction* of scaling are "up" and "down". The second attribute of scaling is *seed preservation.* The three instances of seed preservation are: (1) no seed preservation, with explosion for up-scaling and implosion for down-scaling, (2) background seed preservation, with blow-up and shrinking, for up-scaling and down-scaling, respectively, and (3) root seed preservation, with unfolding and folding for up-scaling and down-scaling, respectively.

4. The machine drawing understanding system

The case in point selected to demonstrate the application and effectiveness of object-process analysis in analyzing and representing PR-embedded systems is the Machine Drawing Understanding System (MDUS). The motivation for the development of MDUS has been that going back from paper to CAD is extremely limited by current commercial systems and requires a great deal of trained human labor. Several works have coped with the problem of converting mechanical engineering drawings into a CAD format (King, 1988; Joseph and Pridmore, 1989; Dori, 1989; Antoine et al., 1990; Collin and Colnet, 1990; Tombre and Vaxiviere, 1991; Collin and Vaxiviere, 1991; Dori and Tombre, 1994). MDUS is an experimental system designed to automate the process of converting mechanical engineering drawings into CAD format (Dori et al., 1993).

Within MDUS, we concentrate on the Lexical Analysis Phase. Probing yet deeper, the Orthogonal Zig-Zag (OZZ) algorithm is presented. This is a vectorization method that is particularly suitable for extracting bars (straight line segments) from engineering drawings. Following the motivation for the development of the system, a series of object-process diagrams represent the system structure and behavior at each one of the three levels.

Fig. 1 is an object-process diagram (OPD) of MDUS and its environment. As the legend explains, objects and processes are denoted by boxes and ellipses, respectively. Structural relations (aggregation and generalization) and procedural relations (links between objects and processes) are explicitly shown. Engineering Drawing is a complete representation of the product or part of the product to be manufactured. The representation can be done in three ways: on paper, as a raster image or in CAD format. The object class Engineering Drawing specializes respectively into three sub-classes: Paper Drawing, Raster Drawing and CAD Drawing. This is expressed by the generalization relation, denoted by the blank triangle in Fig. 1. Applying the process

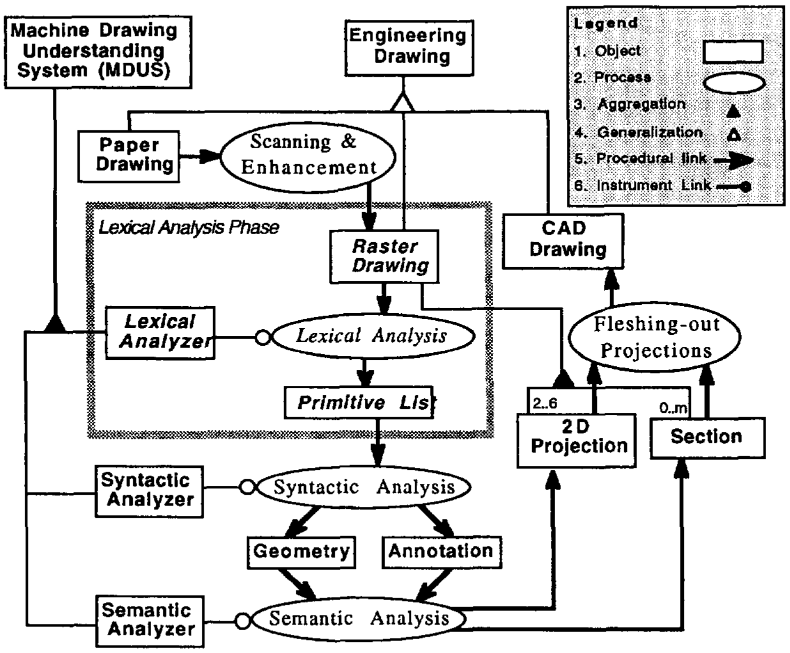

Fig. 1. An object-process diagram of MDUS and its environment.

Scanning & Enhancement to the object Paper Drawing yields Raster Drawing. MDUS is shown to consist of three parts: Lexical Analyzer (Dori et al., 1993), Syntactic Analyzer (Tombre and Vaxiviere, 1991; Dori, 1992), and Semantic Analyzer. This is denoted by the aggregation relation (black triangle). Each one of the three analyzers is an instrument to the corresponding analysis phase, as expressed by the instrument link (blank circle). Thus, for example, the Raster Drawing is processed by the Lexical Analyzer in the Lexical Analysis process, which, in turn, results in the Primitive List.

The Primitive List contains bars – straight line segments, arcs, arrowheads and textboxes (Chai and Dori, 1992). These primitives are processed by the Syntactic Analyzer in the Syntactic Analysis process, in which Geometry is separated from Annotation. Geometry is the set of primitives that delineate the contour of the designed object for each orthogonal view in the drawing, while Annotation is the set of primitives and higher-order objects that define precisely the dimensions, tolerances, and other requirements of the product or part to be manufactured. These objects include dimension-sets, hidden (dashed) lines, axis (dash-dotted) lines, hatching, bearing, threading, welding, etc. The Syntactic Analyzer separates the Geometry primitives from the Annotation ones and uses them to construct higher-level objects, such as dimension-sets.

In the Semantic Analysis process, these objects

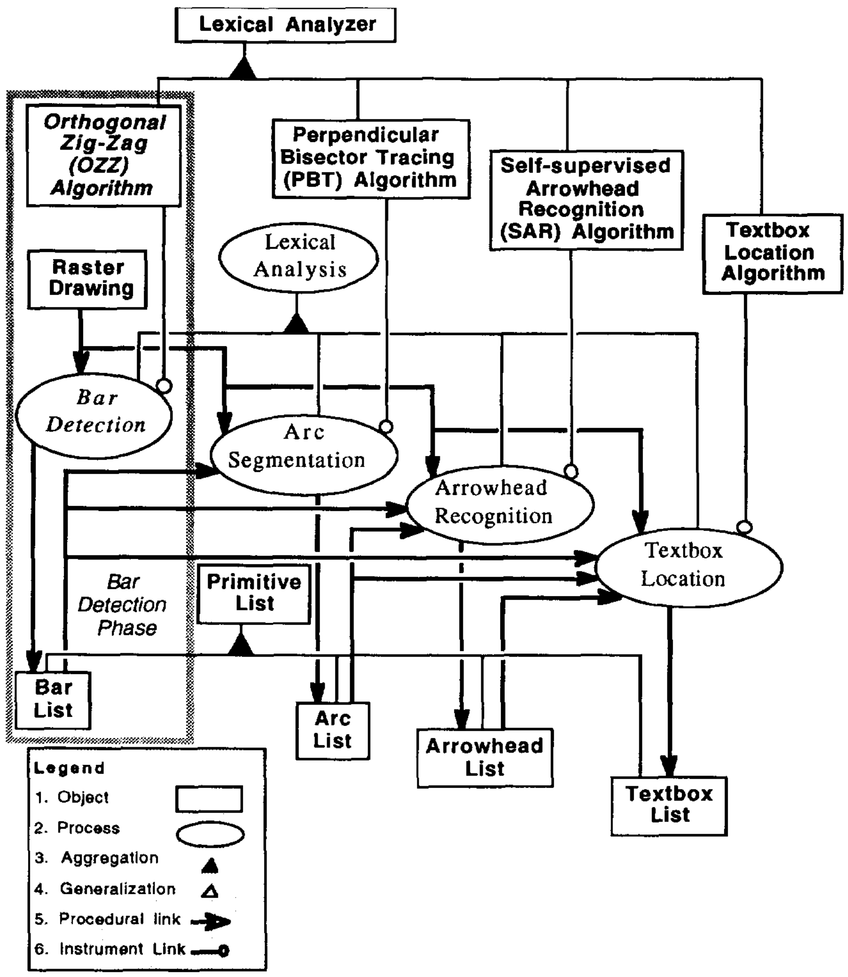

Fig. 2. The Lexical Analysis Phase OPD resulting from scaling up the blow-up frame of the OPD in Fig. 1.

are used to carry out 2-D understanding, i.e., understanding the drawing at the projection level. Finally, the set of 2-D projections (between 2 to 6, as noted by "2..6" above the 2-D Projection box in Fig. 1) and optional $(0..m)$ sections resulting from Semantic Analysis, is processed by the Fleshing-out Projections (or 3-D Reconstruction) process (Wesley and Markowski, 1981; Preiss, 1984). This process produces the CAD Drawing, i.e., a complete 3-D representation of the part described originally in the Paper Drawing. In a CAD Drawing, each entity has a precise geometric definition in the 3-D space, including its tolerance. Hence, it is amenable to further manipulations by a CAD system.

5. Scaling up the lexical analysis phase of MDUS

The Lexical Phase is denoted in Fig. 1 as the collection of four things, surrounded by a gray, thick blow-up frame. The frame encloses three objects – Raster Drawing, Lexical Analyzer, and Primitive List, and one process – Lexical Analysis. To focus our attention on the Lexical Phase, it undergoes up-scaling. As explained in Section 3, scaling up involves increasing the level of detail in a subsequent OPD (object-process diagram). This is done in the OPD of Fig. 2. The "seed" here is Lexical Analyzer, and it is preserved as the root of the scaled up OPD. As Fig. 2 shows, Lexical Analyzer consists of four component objects, each object being an algorithm.

The four algorithms are: Orthogonal Zig-Zag (OZZ), Perpendicular Bisector Tracing (PBT), Self-supervised Arrowhead Recognition (SAR), and Textbox Location. The process Lexical Analysis is unfolded like the object Lexical Analyzer, exposing the four constituent processes of Lexical Analysis: Bar Detection, Arc Segmentation, Arrowhead Recognition, and Textbox Location. Each one of the four algorithms is an instrument that does part of the Lexical Analysis process. This is denoted by an instrument link (a line ending with a blank circle; see legend). OZZ is the instrument for the Bar Detection (vectorization) process, PBT is the instrument for Arc Segmentation process, SAR – for the Arrowhead Recognition process, and the Textbox Location Algorithm – for the Textbox Location process. As the object-process diagram shows, each one of these four processes uses as input the Raster Drawing, and the list of primitives that has already been obtained. Thus, the Bar Detection process uses only the Raster Drawing as input to OZZ. Arc Segmentation uses the Bar List obtained from the Bar Detection along with the Raster Drawing as input to PBT. The next process, Arrowhead Recognition, uses the Raster Drawing and the Bar List plus the Arc List, as inputs to the SAR algorithm. Finally, Textbox Location uses the Arrowhead List in addition to the Raster Drawing and the three previously obtained lists.

6. Bar detection: the Orthogonal Zig-Zag (OZZ) algorithm

Bars are the most prominent constituent in most engineering drawings. They appear in the raster image as elongated rectangles of black pixels, usually with noisy edges. Vectorization, or recognition of bars (straight line segments) is the main problem in low-level processing (Arcelli and Sanniti di Baja, 1984). This is a heavy undertaking if each pixel is to be addressed at least once, as required by Hough Transform or thinning-based methods. Most vectorization algorithms get a skeleton as an input, and find a subset of the skeleton's pixels. Following is a brief overview of various dense-pixel vectorization algorithms that motivated the development of the Orthogonal Zig-Zag (OZZ) algorithm, which is sparse-pixel.

To further focus our attention on bar detection, we apply yet another up-scaling to the Bar Detection Phase, surrounded by the gray blow-up frame on the left hand side of the Lexical Analysis Phase OPD depicted in Fig. 2. In particular, we concentrate on the italicized object Orthogonal Zig-Zag (OZZ) Algorithm and the italicized process Bar Detection.

The underlying idea of OZZ is inspired by a light beam conducted by an optic fiber (Dori and Chai, 1992): a one-pixel-wide "ray" travels through a rectangular black pixel area designating a bar, as if it were a conducting pipe. The ray's trajectory is always parallel to one of the drawing axes. The course of the ray zig-zags orthogonally, changing direction by 90° each time a white area is encountered. Accumulated statistics about the two sets of black run-

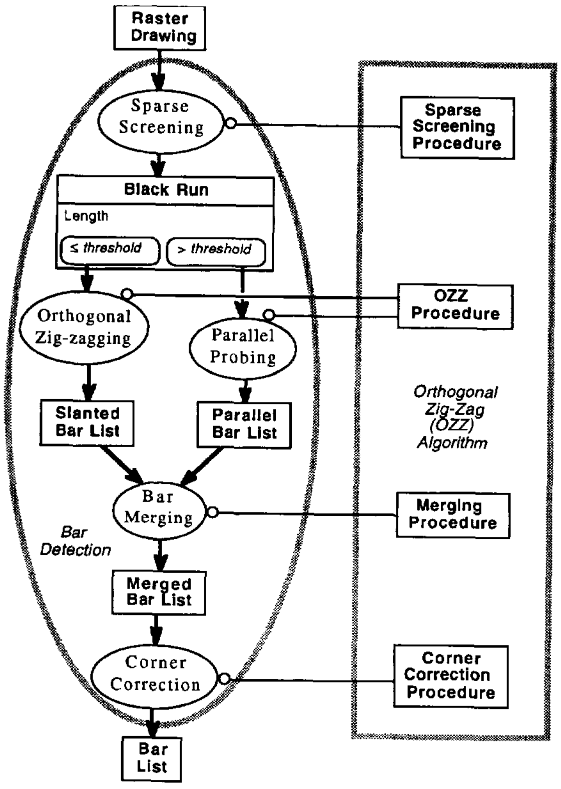

Fig. 3. The Bar Detection Phase OPD resulting from scaling up the blow-up frame of the OPD in Fig. 2. The Bar Detection Process and Orthogonal Zig-Zag Algorithm object of Fig. 2 are blown up.

lengths gathered along the way provide data for deciding about the presence of a bar, its endpoints and its width, and enable skipping junctions. Fig. 3 is an OPD describing the Bar Detection Phase, resulting from scaling up the blow-up frame of the OPD in Fig. 2. The Bar Detection process and the OZZ Algorithm object of Fig. 2 are both blown up, and each is inscribed within a corresponding (gray ellipse and box) blow-up frame. The up-scaling of Bar Detection and OZZ Algorithm makes explicit not only their constituents, but also the instrument relations between each one of the OZZ Algorithm's constituents and those of the Bar Detection process. Thus, the Sparse Screening Procedure is the instrument of the Sparse Screening process, the OZZ Procedure is the instrument of both the Orthogonal Zig-zagging and Parallel Probing, the Merging Procedure is the instrument of the Bar Merging process, and the Corner Correction Procedure is the instrument of the Corner Correction process. The result of the last process, Corner Correction, is the result of the entire Bar Detection process, i.e., the object Bar List. During the Sparse Screening process, the Raster Drawing is screened sparsely, i.e., only each nth line of the raster file is examined. When the screening encounters a black pixel, a black-pixel run count starts. As expressed in Fig. 3, the object Black Run has Length as an attribute. If Length is below a certain threshold, the potential bar is diagonal, otherwise it is parallel to either one of the drawing axes. Accordingly, the OZZ Procedure serves as an instrument to either the Orthogonal Zig-zagging or Parallel Probing process. The former process yields the object Slanted Bar List, while the latter yields the Parallel Bar List. Both lists undergo a Bar Merging Process, whose instrument is the merging Procedure. Corner Correction is applied to the resulting Merged Bar List, which finally outputs the Bar List.

8. Conclusion

Machine vision systems feature complexity and a substantial behavioral aspect beside the structural one. The work suggests that a holistic view of machine vision systems, which is sometimes set aside, should be adopted. An object-process analysis (OPA) approach, which is an extension of object-oriented analysis, has been suggested as a methodology for analysis, representation and communication of machine vision systems. The main benefit of OPA is presentation of the structural and behavioral aspects of a system within a single, coherent frame of reference at any level of detail without loss of consistency and links among the various abstraction levels. The Machine Drawing Understanding System (MDUS) was analyzed as a case in point to demonstrate how OPA can be used both to strike the required balance between the systems' structure and behavior and to manage the inherent complexity of such systems. The analysis was done in a top-down fashion, showing concurrently objects in the system and how they interact through processes. The top-down presentation was made possible by the use of scaling. Scaling is a powerful tool for complexity management, as it provides for controlling the visibility and level of detail of objects and processes of interest.

References

Antoine, D., S. Collin and C. Tombre (1990). Analysis of technical documents: the REDRAW system. In: *Pre-Proceedings IAPR Workshop on Syntactic & Structural Pattern Recognition*, Murray Hill, NJ, 1–20.

Arcelli, C. and G. Sanniti di Baja (1984). Quenching points in distance labeled pictures. In: *Proc. 7th Internat. Conf. on Pattern Recognition*, Montreal, 344–346.

Chai, I. and D. Dori (1992). Extraction of text boxes from engineering drawings. In: *Proc. SPIE / IS\& T Symposium on Electronic Imaging Science and Technology, Conference on Character Recognition and Digitizer Technologies*, San Jose, CA, February 9–14.

Chen, P.P., (1976). The entity relationship model: toward a unifying view of data. *ACM Trans. on Data Base Systems* 1 (1), 9–36.

Coad, P. and E. Yourdon (1991). *Object Oriented Analysis* (2nd Ed.), Prentice-Hall, Englewood Cliffs, NJ.

Collin, S. and D. Colnet (1990). Analysis of dimensions in mechanical engineering drawings. In: *Proc. Machine Vision Applications*, 105–108.

Collin, S. and P. Vaxiviere (1991). Recognition and use of dimensioning in digitized industrial drawings. In: *Proc. First Internat. Conf. on Document Analysis and Recognition*, IEEE Computer Society, Saint-Malo, France.

De Marco, T. (1978). *Structured Analysis and System Specification*, Yourdon Press, New York.

Dori, D. (1989). A syntactic/geometric approach to recognition of dimensions in engineering machine drawings, *Computer Vision, Graphics and Image Processing* 47, 1–21.

Dori, D. (1991). Self-structural syntax-directed pattern recognition of dimensioning components in engineering drawings. In: H.S. Baird, H. Bunke, and K. Yamamoto, eds., *Structured Document Image Analysis*, Springer, Heidelberg.

Dori, D. (1992). Dimensioning analysis: a step towards automatic high level understanding of engineering drawings, *Comm. ACM* 35 (10), 92–103.

Dori, D. (1995). Object-process analysis: maintaining the balance between system structure and behaviour. *J. Logic Comput.* 5 (2) (to appear).

Dori, D. and I. Chai (1992). Orthogonal zig-zag: an efficient method for extracting bars in engineering drawings. In: C. Arcelli, L.P. Cordella and G. Sanniti di Baja, eds., *Visual Form*, Plenum, New York, 127–136.

Dori, D., Y. Liang, J. Dowell and I. Chai (1993). Sparse pixel recognition of primitives in engineering drawings. *Machine Vision and Applications* 6, 69–82.

Dori, D. and K. Tombre (1994). From engineering drawings to 3-D CAD models – are we ready now? *Computer-Aided Design* (to appear).

Dori, D., I. Phillips and R.M. Haralick (1994). Incorporating documentation and inspection into computer integrated manufacturing: an object-process approach. In: S. Adiga, ed., *Applications of Object-Oriented Technology in Manufacturing*, Chapman & Hall, London.

Embley, D.W., B.D. Kurtz and S.N. Woodfield (1992). *Object Oriented Systems Analysis*, Prentice Hall, Englewood Cliffs, NJ.

Joseph, S.H. and T.P. Pridmore (1989). Knowledge directed interpretation of mechanical engineering drawings. *IEEE Trans. Pattern Anal. Mach. Intell.*, August.

King, A.K. (1988). An expert system facilitates understanding the paper engineering drawings. In: *Proceedings, IASTED International Symposium Expert Systems Theory and Their Applications*, Los Angeles, CA, ACTA Press, Anaheim, 169–172.

Nerson, J.M. (1993). Applying object oriented analysis and design. *Comm. ACM* 35 (9), 63–74.

Preiss, K. (1984). Constructing the solid representation from engineering projections. *Computers and Graphics* 8 (4), 381–389.

Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy and W. Lorensen (1991). *Object-Oriented Modeling and Design*. Prentice-Hall, Englewood Cliffs, NJ.

Shlaer, S. and S.J. Mellor (1992). *Object Lifecycles: Modeling the World in States*, Yourdon Press, PTR Prentice Hall, Englewood Cliffs, NJ.

Tombre, K. and P. Vaxiviere (1991). Structure, syntax and semantics in technical document recognition. In: *Proc. First Internat. Conf. on Document Analysis and Recognition*, IEEE Computer Society, Saint-Malo, France.

Wesley, M.A. and G. Markowski (1981). Fleshing out projections, *IBM J. Res. Development* 26, 934–953.