KNOWLEDGE ORGANIZATION AND INTERPRETATION PROCESS IN ENGINEERING DRAWING INTERPRETATION

Pascal Vaxivière and **Karl Tombre**
CRIN/CNRS & INRIA Lorraine
Bâtiment LORIA, Campus Scientifique, B.P. 239
54506 Vandœuvre CEDEX, France
Karl.Tombre@loria.fr

ABSTRACT

This paper describes the way knowledge is organized and used in our CELESSTIN document interpretation system, which is a blackboard-based, multi-expert prototype system for extracting functional CAD information from scanned engineering drawings. After a general overview of the way CELESSTIN is implemented, we describe in detail several of the specialists cooperating to perform the high-level drawing interpretation: extraction of symmetric entities, disassembling, kinematics analysis.

1. Introduction

In the design of a complete document analysis system, a lot of attention is given to the individual components, especially the low-level image analysis procedures and the pattern recognition methods used in the interpretation process. But in the case of systems which aim at attaining high-level understanding of the document and semantical interpretation, the issue of knowledge organization and contextual reasoning becomes also crucial.

This is the case in our work on the conversion of mechanical engineering drawings to high-level CAD. In this context, we have developed several versions of CELESSTIN, a prototype system for conversion of mechanical engineering drawings to CAD[1]. Our aim is not to build a universal system capable of converting *any* engineering drawing to some CAD description, but rather to explore the power of knowledge-based techniques for performing high-level interpretation of documents. This explains our "depth-first" approach: we chose to focus on a narrow area in mechanical engineering, i.e. mechanisms such as speed reducers or gearboxes, and to use as much *a priori* knowledge as possible in the interpretation, taking into account structural and syntactical as well as semantical knowledge.

In CELESSTIN, we have designed a vectorization method, based on several improvements of a method proposed by Lin *et al.*[2, 3]. But as a vector in itself has no "semantics", we had to use a richer structure as the basic element for the interpretation process. We chose the *block*, i.e. the minimum closed polygon drawn in thick lines. Thin lines and isolated patterns can then be considered as mere *attributes* of a block (hatching, threading...). The structure is enriched by several pattern recognition procedures, which recognize entities such as dot-dashed lines, hatched cross sections or dimensioning sets[4]. Various reasoning methods then use this basic structure, their cooperation

being ensured by a blackboard-based multi-expert system written in ATOME[5], which progressively builds an interpretation of the whole drawing in terms of technological entities[6].

This paper describes the way knowledge is organized and used in CELESSTIN. After a general overview of the way CELESSTIN is implemented, we describe in detail several of the specialists cooperating to perform the high-level drawing interpretation.

2. Overview of the system

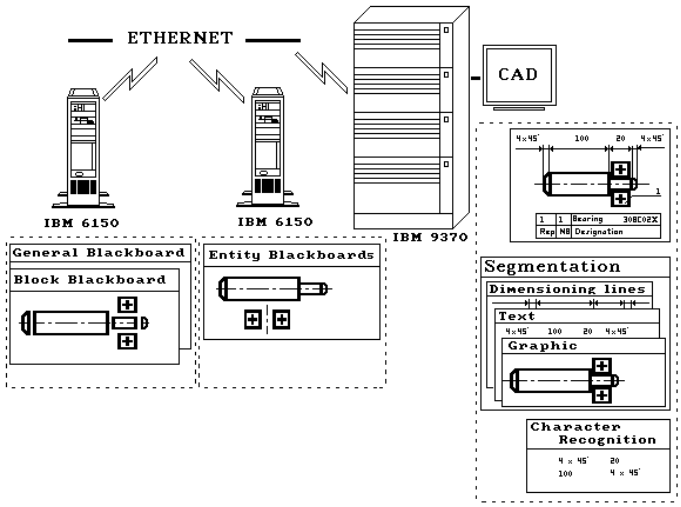

Figure 1: Overview of Celesstin's implementation.

CELESSTIN was partly funded by a research grant from IBM France to ESSTIN, an engineering school in Nancy. A CAD system, CATIA (registered trademark of Dassault Systemes), was available on an IBM 9370, and Common-Lisp was available on several UNIX stations of type IBM 6150. We were aware of the fact that this hardware was quite outdated and its computing power was very limited; but a very basic communication protocol was implemented to let the different machines cooperate, thus increasing the overall capabilities of the system. Figure 1 gives a general overview of the system.

The methods use for low-level processing (image pre-processing, vectorization, block extraction...) are written in C and are running on the IBM 9370, whereas the high-level interpretation is implemented on two 6150's, using a Common-Lisp version of ATOME. The different blackboards are distributed over these two machines, the first being responsible for the general interpretation strategy and for the labeling of blocks, the second for higher-level reasoning on symmetrical entities. As illustrated by Figure 2, the general interpretation strategy activates a series of *tasks*, which themselves call different agents (called *specialists* in ATOME). Each specialist is responsible for a basic reasoning process and cooperates with the other specialists using the blackboard model for problem solving and knowledge distribution[7, 8].

More precisely, the different components of a system written in ATOME are:

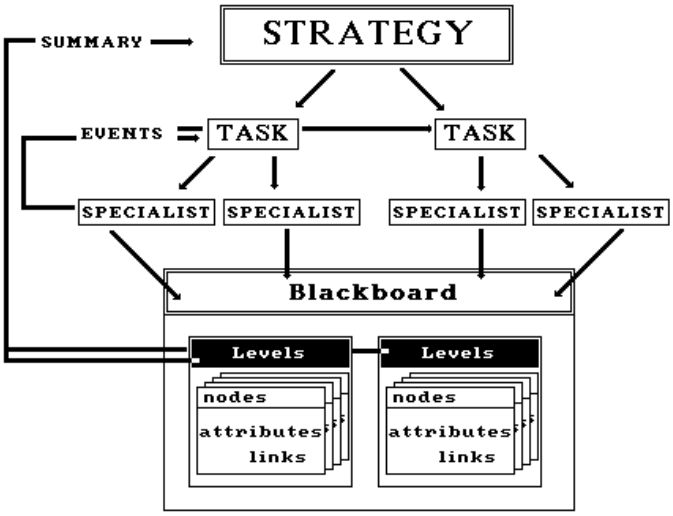

Figure 2: Strategy, tasks and specialists.

- *Blackboards*, which are organized in several *levels* corresponding to classes of objects. In a given layer, an object is represented by a node with which all the appropriate information is associated: attributes describing the object's features and links referring to its relations with other objects. A *summary* of the blackboard's main content is available to the strategy. In our case, we have one blackboard level for each phase in our analysis: lines and blocks, shafts, symmetric entities, functional setups.

- *Specialists* access the blackboard directly. A specialist has a precondition part, i.e. a set of predicates which determinate when the specialist is activated, and an action part, which can be a program, an expert system or a set of rules for updating the blackboard. In our system, some specialists are plain C programs performing feature extraction or simple recognition tasks; other specialists are written in Lisp and correspond to reasoning tasks, such as the functional reasoning described in § 3.2 and 3.3.

- *Tasks* are knowledge sources responsible for the control, whose aim is to direct and coordinate the action of a subset of specialists. They are also made of a set of rules, which guarantees maximum flexibility, are activated by the strategy and receive feedback from the specialists through events. They therefore correspond to the control level of the system, responsible for activating the right specialists, receiving the results they yield and coordinating these results in a homogeneous way, so that they can be used by the next task.

- The *strategy* receives a summary of the state of the blackboard and supervises the overall interpretation process, activating new tasks when necessary. It is the highest level of control in the system. In the present state of our system, the strategy is quite straightforward and "linear", as will be seen in this paper, but the

blackboard paradigm is powerful enough to make sophisticated strategies possible.

3. ATOME Implementation of CELESSTIN

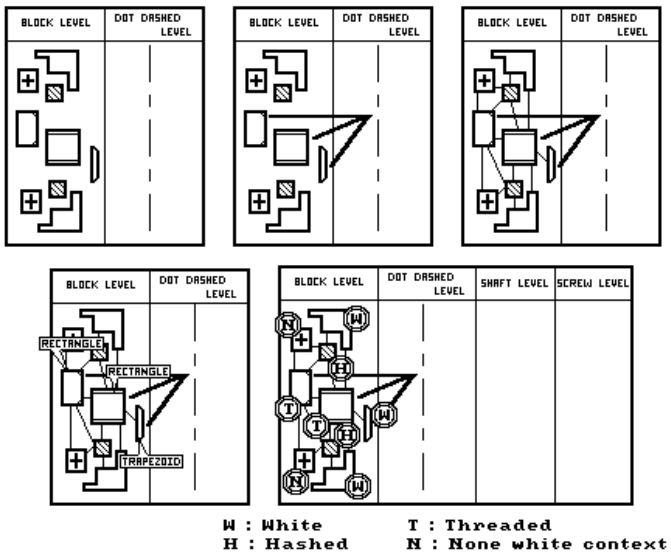

Figure 3: Creation of the shaft level.

The interpretation of an engineering drawing is decomposed into several steps: blackboard initialization, matter/empty space separation[1], recognition of screws and shafts along the dot-dashed lines, interpretation of the drawing in terms of technologically meaningful elements.

The blackboard is initialized by creating two levels: the dot-dashed level and the block level. A set of rules activated by the strategy leads to the creation of the shaft level, as illustrated by Figure 3. The blocks and the dot-dashed lines are basic elements yielded by the low-level process, written in C. They are systematically extracted from the low-level features yielded by the vectorization. A first set of specialists compute relations such as “is-centered-on” between a block and a dot-dashed line, or “neighbor-of” between blocks. Then the type of each block (white, hatched, threaded, non-white context) is determined from its thin-line attributes.

A shaft recognition task is then activated, which groups blocks along the axis line and propagates matter[9]; this leads to the recognition of shafts and screws and to the creation of the corresponding new levels in the blackboard.

3.1. Symmetric entities

In order to get a higher level of interpretation, the next step is to leave the world of blocks and use another set of rules, applied to a new structure, defined as the combination of several blocks, and called the *entity*[6]. When a person looks at a drawing of a

mechanical device, the first entity which is identified is probably the casing (Figure 4). Whatever shape it has, it appears as being a unique object, and actually all its parts

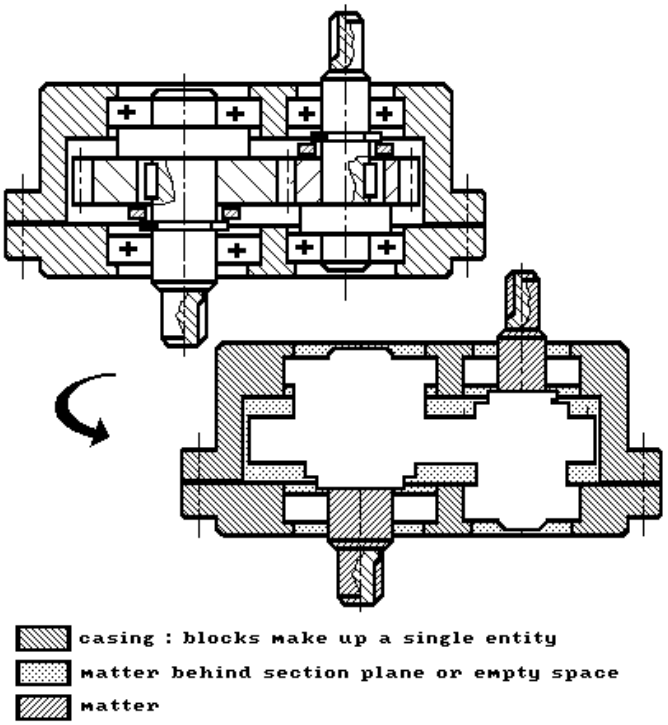

Figure 4: Recognition of casing.

have the same hatching pattern.

This is however only a first initialization. It is much more interesting to identify all the *symmetric entities* present in the drawing. Therefore, the "entity" blackboard is initialized with the result of the block extraction, and a set of symmetry finding specialists cooperate to find all entities symmetric with respect to a given axis (dot-dashed line).

- The *mathematical symmetry* specialist identifies all pairs of blocks which are nearly symmetric with respect to a dot-dashed line. Figure 5 illustrates the rules which identify all such symmetric entities: for each block, the symmetric of its center of gravity with respect to the axis line is computed and all blocks in the neighborhood of the supposed symmetric are compared with the candidate block. The block in this neighborhood having the same shape is paired with the candidate block to form a symmetric entity. A tolerance threshold is used to take into account drafting, scanning and vectorization errors.

- The previous symmetry is not sufficient, as some blocks may be technically symmetric although they have different shapes. This is the case with partial sections or other specific features which are added to the drawing to show some mechanical details. The *mechanical symmetry* specialist therefore applies another set of

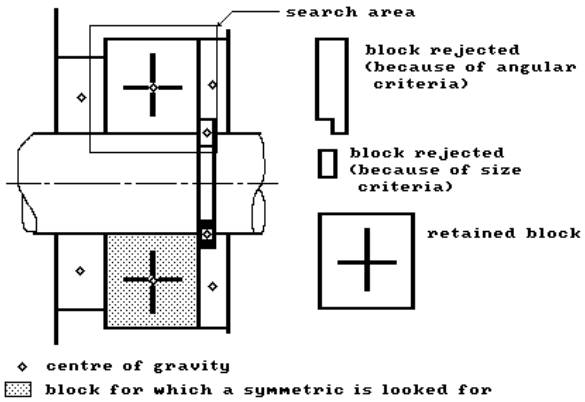

Figure 5: Mathematical symmetry.

rules which recognize symmetric entities even without exact shape matching (Figure 6). The strategy followed by this specialists goes like this:

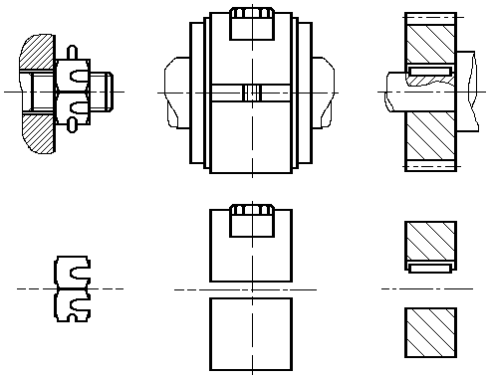

Figure 6: Mechanical symmetry.

- when the enclosing rectangles of two blocks match, the blocks are said to be symmetric;
- if the hatching pattern of the two blocks is the same, they are said to be symmetric, even if their shapes differ;
- all blocks contained in the symmetric projection of the largest enclosing rectangle are added to the symmetric entity.

- In some cases, two blocks crossed by a dot-dashed line are symmetric with respect to another dot-dashed line. The *double symmetries* specialist recognizes such configurations. This double symmetry may be purely incidental and have no effect on the interpretation of the drawing. This is the case in the first example of Figure 7, where the blocks on both sides of the split ball bearing are crossed by the axis line of the shaft, and they are also symmetric with respect to the screw which adjusts

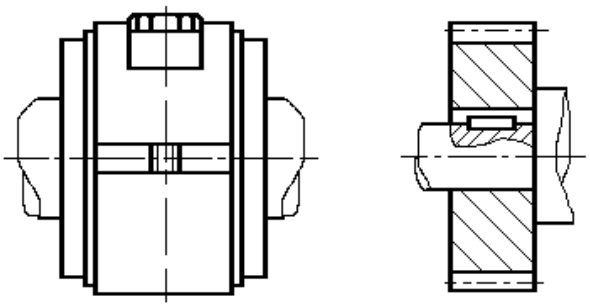

Figure 7: Double symmetries.

the strap. But in some cases, especially when the two symmetry lines are parallel, this double symmetry is extremely useful for later interpretation stages. For instance, in the second example, the teeth of the gear are represented by empty blocks crossed by a dot-dashed line; these blocks are symmetric with respect to the axis line of the shaft and this double symmetry is a typical "signature" for a gear. This information is stored in the corresponding blackboard for later use by higher-level tasks.

At the end of this task, these three specialists have put on the corresponding blackboard a list of recognized entities, to which a technical meaning can be assigned. Further reasoning is then performed on these entities.

3.2. Disassembling

A new blackboard level is activated for the disassembling analysis, which is based on a set of rules which try to disassemble the represented object. The idea is that it *must* be possible to disassemble a mechanical setup and decompose it into functional parts[10]. This reasoning should ideally be performed on a 3D model[11], but in the present state of our system, we suppose in addition that it is possible to analyze the disassembling sequence by looking at only one view of the object. The basic strategy in this task can then be based on a straightforward computation of the degree of "movability" of each entity with respect to the others, as illustrated by Figure 8. The specialist removes one

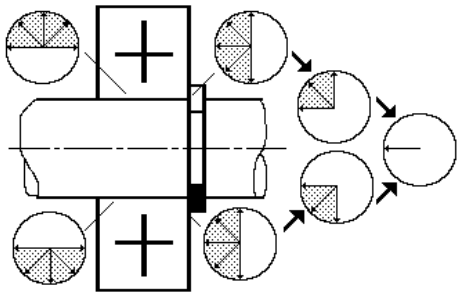

Figure 8: Degree of movability.

entity after the other and updates consequently the degree of movability for its neighbors. For this, the following rules are repeated in a loop:

- Remove the casing if there is one.
- If an entity *without attributes* and located at the current end of a shaft hinders all motion along the axis line, it is considered as representing empty space and is disassembled from the setup.
- Locking devices hold the parts of a shaft in place; hence they have no degree of movability according to our rules, as their rôle is to hinder all motion. Therefore, when an entity *with attributes*, located at the current end of a shaft, has no degree of movability, it is labeled as a locking device and split into two parts in order to be "forcibly" removed from the setup.
- When an entity at the current end of a shaft has a degree of movability, remove it in the direction where it can move.
- In all cases, update degree of movability for all neighbors of removed entity.

Figure 9 illustrates how this specialist works by progressive removal of symmetrical entities.

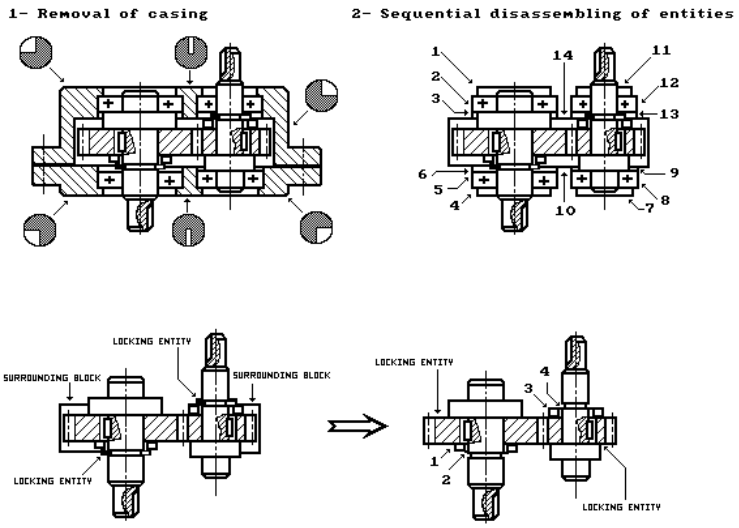

Figure 9: Disassembling.

We hence see that this analysis already extracts some functional information, such as the presence of locking devices, and helps in determining where the empty space is located with respect to the mechanical device itself. This information is further complemented by the kinematics analysis.

3.3. Kinematics

Another specialist creates an additional blackboard level, devoted to the kinematics of the whole setup. The idea is to rotate the shafts around the axes and to propagate this rotation to the neighboring entities. The behavior of these entities gives clues about their functionalities (Figure 10). The following rules are applied:

- If an entity is symmetric with respect to an axis line, it may *a priori* rotate around this axis.
- An entity *without attributes* which "rotates" with a shaft and touches a fixed (non-moving) entity or an entity rotating with another shaft represents empty space.
- An entity *with attributes* touching both a rotating shaft and a fixed part is a bearing entity: ring, ball bearing, etc.
- An entity *with attributes* connecting two shafts is a transmission entity: gear, pulley, etc.

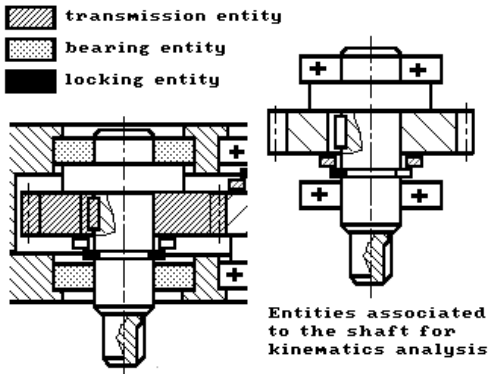

Figure 10: Analysis of kinematics.

Thus, at the end of these two analyses, we have recognized locking devices, bearing entities, transmission entities and empty space, and the spatial and structural relations between them.

3.4. Result

Figure 11 is a screen dump of the result of our interpretation system on a mechanical engineering drawing. As the recognized entities can be replaced by the corresponding entity in the CAD library, they can be displayed using usual shading techniques (upper left corner); as the functionalities themselves have been identified, it is also possible to extract from the results of the interpretation the functional setup of the represented gearbox, as illustrated in the lower left corner.

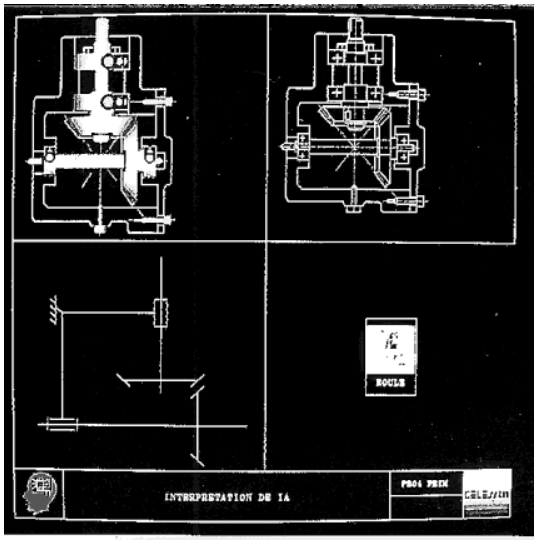

Figure 11: Result of interpretation.

4. Problems and perspectives

We are aware of the fact that even in the area of mechanical engineering, our prototype is only covering a very limited set of the possible functional interpretations. Of course, on one hand, the multi-agent blackboard-based model of reasoning we have used is flexible and powerful enough to add much more expertise, if only the latter can be formalized and written down as a set of tasks, specialists and corresponding knowledge rules, recognition processes and reasoning mechanisms. But on the other hand, even the simple and limited knowledge base we use becomes rapidly difficult to manage, in terms of consistency rules, clarity, as well as in terms of writing down an efficient general strategy. Part of the solution lies in the further development of our methodology for performing technical document interpretation. But we also believe that it is not sufficient to organize knowledge as we have done until now, i.e. just by writing the specialists as C programs or lines of Lisp. The knowledge must be better organized, following a taxonomy and probably using some kind of hierarchical knowledge representation tool, such as a frame language[12]. This would enhance the power of the multi-agent we have used, and we hope it will allow us to implement much more ambitious reasoning mechanisms, where multiple specialists have to cooperate intensively, as is the case with our project for real 3D reconstruction from multiple views[13].

5. References

[1] P. Vaxivière and K. Tombre. Celesstin: CAD Conversion of Mechanical Drawings. *IEEE COMPUTER Magazine*, 25(7):46–54, July 1992.

[2] X. Lin, S. Shimotsuji, M. Minoh, and T. Sakai. Efficient Diagram Understanding with Characteristic Pattern Detection. *Computer Vision, Graphics and Image*

Processing, 30:84–106, 1985.

[3] P. Vaxivière and K. Tombre. Subsampling: A Structural Approach to Technical Document Vectorization. In *IAPR Workshop on Syntactic and Structural Pattern Recognition, Nahariya (Israel)*, October 1994. To appear in "Shape, Structure and Pattern Recognition", D. Dori and A. Bruckstein, editors, 1995.

[4] S. Collin, K. Tombre, and P. Vaxivière. Don't Tell Mom I'm Doing Document Analysis; She Believes I'm in the Computer Vision Field. In *Proceedings of 2nd International Conference on Document Analysis and Recognition, Tsukuba (Japan)*, pages 619–622, October 1993.

[5] H. Lâasri and B. Maître. Flexibility and Efficiency in Blackboard Systems : Studies and Achievements in ATOME. In V. Jagannathan, R. Dodhiawala, and L. Baum, editors, *Blackboard Architectures and Applications*, chapter 14, pages 309–322. Academic Press, Boston, 1989.

[6] P. Vaxivière and K. Tombre. CELESSTIN IV : Knowledge-Based Analysis of Mechanical Engineering Drawings. In *Proceedings of IEEE International Conference on Systems Engineering, Kobe (Japan)*, pages 242–245, September 1992.

[7] H.P. Nii. Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures – Part 1. *AI Magazine*, 7(2):38–53, 1986.

[8] H.P. Nii. Blackboard Systems: Blackboard Application Systems, Blackboard Systems from a Knowledge Engineering Perspective – Part 2. *AI Magazine*, 7(3):82–106, 1986.

[9] P. Vaxivière and K. Tombre. Interpretation of Mechanical Engineering Drawings for Paper–CAD Conversion. In *Proceedings of IAPR Workshop on Machine Vision Applications, Tokyo (Japan)*, pages 203–206, 1990.

[10] I. Lee, S. Dan, T. Kitahashi, and N. Abe. A Study on a Method of Dividing Machine-parts into Functional Groups for Technical Illustrations. In *Proceedings of 2nd International Conference on Document Analysis and Recognition, Tsukuba (Japan)*, pages 886–889, 1993.

[11] R.H. Wilson and A. Schweikard. Assembling Polyhedra with Single Translations. In *Proceedings of the IEEE International Conference on Robotics and Automation, Nice (France)*, pages 2392–2397, 1992.

[12] S.H. Joseph and T.P. Pridmore. Knowledge-Directed Interpretation of Mechanical Engineering Drawings. *IEEE Transactions on PAMI*, 14(9):928–940, September 1992.

[13] D. Dori and K. Tombre. From Engineering Drawings to 3-D CAD Models: Are We Ready Now? *Computer-Aided Design*, 29(4):243–254, April 1995.