Explicit and Persistent Knowledge in Engineering Drawing Analysis

Thomas C. Henderson

UUCS-03-018

School of Computing
University of Utah
Salt Lake City, UT 84112 USA

October 10, 2003

Abstract

Domain knowledge permeates all aspects of the engineering drawing analysis process, including understanding the physical processes operating on the medium (i.e., paper), the image analysis techniques, and the interpretation semantics of the structural layout and contents of the drawing. Additionally, an understanding of the broader reverse engineering context, within which the drawing analysis takes place, should be exploited. Thus as part of a wider project on the reverse engineering of legacy systems, we have developed an agent-based engineering analysis system called NDAS (nonDeterministic Agent System).

In this paper, we discuss the nature of such a system and how knowledge can be made explicit (both for agents and humans) and how performance models can be de£ned, calibrated, monitored, and improved over time through the use of persistent knowledge. A framework is proposed that allows computational agents to: (1) explore the threshold space for an optimal analysis of the drawing, (2) control information gain through agent invocation, (3) incorporate and communicate knowledge, and (4) inform the software engineering and system development with deep knowledge of the relationships between modules and their parameters (at least in a statistical sense).

Explicit and Persistent Knowledge in Engineering Drawing Analysis

Thomas C. Henderson
School of Computing, University of Utah

Abstract

Domain knowledge permeates all aspects of the engineering drawing analysis process, including understanding the physical processes operating on the medium (i.e., paper), the image analysis techniques, and the interpretation semantics of the structural layout and contents of the drawing. Additionally, an understanding of the broader reverse engineering context, within which the drawing analysis takes place, should be exploited. Thus as part of a wider project on the reverse engineering of legacy systems, we have been developing an agent-based engineering analysis system called NDAS (NonDeterministic Agent System).

In this paper, we discuss the nature of such a system and how knowledge can be made explicit (both for agents and humans) and how performance models can be defined, calibrated, monitored, and improved over time through the use of persistent knowledge. A framework is proposed that allows computational agents to: (1) explore the threshold space for an optimal analysis of a drawing, (2) control information gain through agent invocation, (3) incorporate and communicate knowledge, and (4) inform the software engineering and system development with deep knowledge of the relationships between modules and their parameter (at least in a statistical sense).

1.0 Introduction

The reverse engineering of legacy systems is a difficult and complex problem, but of vital importance. This usually involves a physical instance of the system, as well as some paper drawings produced by hand or from mechanical CAD systems. The goal may range from exact replication, to changing some parameters, to a major re-design. For example, Figure 1 shows a gearbox that operated for many years as part of a shipyard crane system. Developing reverse engineering techniques from such a physical example and any available related engineering drawings is our goal.

Figure 1. Newport News Gearbox

Figure 2 shows the overall reverse engineering system we are developing; the goal is to take advantage of data about the system in all its forms: drawings, 3D scans, and CAD models as they are constructed, as well, and to allow the user

virtual access during the redesign process (see Figure 3). The wider knowledge involved includes manufacturing information and constraints, design analysis codes (e.g., stress or aerodynamics), cost/performance models, etc.

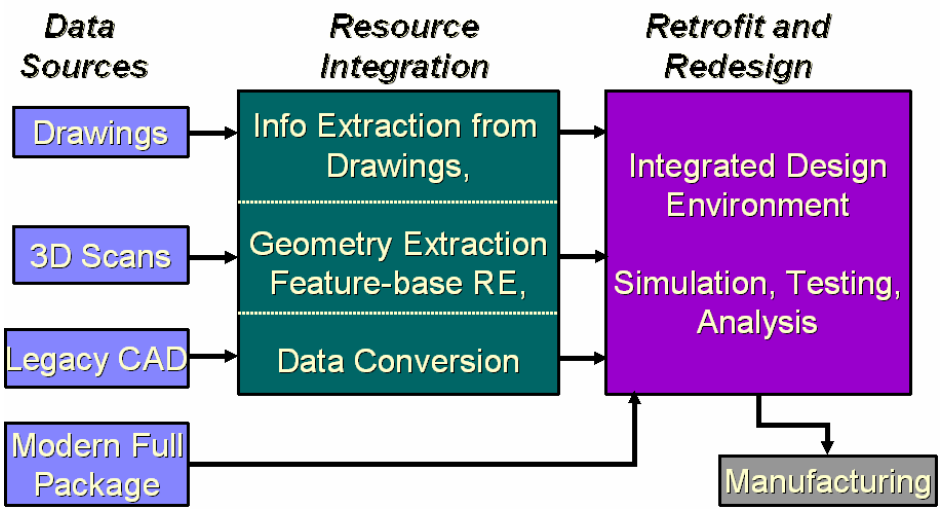

Figure 2. Reverse Engineering System

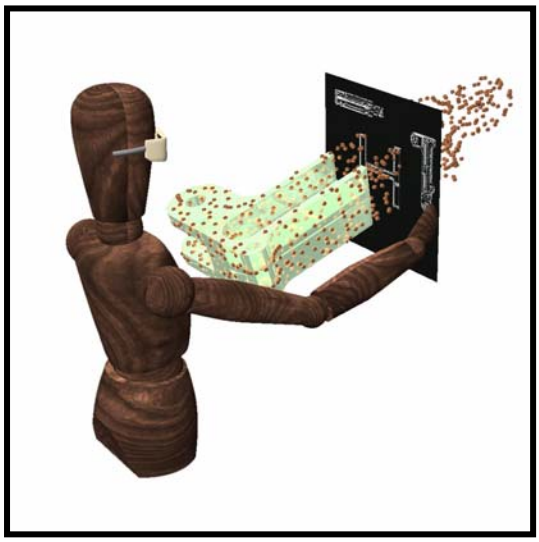

Figure 3. Envisioned Virtual Interface to model surface, point cloud and drawing data.

2.0 The Vision

Before giving details on the systems we have been building, we would like to give our vision of how to construct a system so that domain knowledge can be exploited in a powerful way. We now give a high-level summary of our proposed theoretical framework and enumerate some advantages that may result from this approach.

Figure 4 shows a set of agents, Ai, each of which produces various outputs using a set of parameters and thresholds, Ti, and each having an associated model (or set of models), Pi(Xi|Ti), describing the agent's variance from the ideal in terms of some appropriate measure. Knowledge of three sorts (physical, image analysis, and structural interpretation) is available and informs the agents' actions and understanding of each others results. Higher-level control processes may exploit this in several ways:

1. **Explore** the threshold space for global optima (see feedback loop in Figure 4).
2. **Control** acquisition of new data (e.g., view token generation as state estimation and select agent action that optimizes information gain).
3. **Incorporate** knowledge in abstract form and communicate abstractions between agents and users.
4. **Inform** the software engineering and system development with deep knowledge of the relationships between modules and their parameters (at least in a statistical sense).

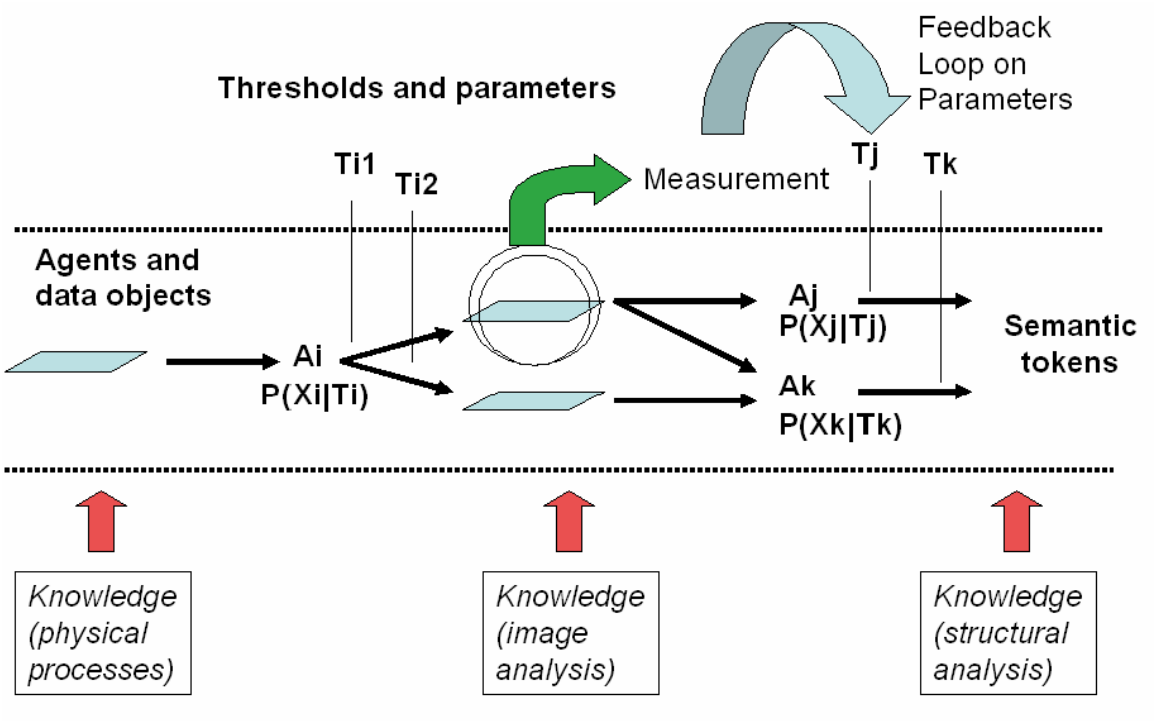

Figure 14. Smart Agents Network System

The current status of the project (called the Smart Agent Network System or *SANS*) is that the core image and structural analysis components have been developed and applied

to engineering drawing analysis to gain experience and insight into crucial agents, their parameters and interactions. We are now exploring the representation of this domain knowledge in specific nomenclatures. We are also investigating state estimation frameworks to provide a more incremental analysis based on observations provided by the system, and the associated information measures (see [Catlin 1989] for an introduction to the area). Notice that each program execution can be viewed itself as a measurement on the image, and the set of measurements will be used by a control process to achieve the best interpretation of the drawing.

Note that state estimation is a reasonably mature tool in many engineering applications. More recently, such methods have been incorporated in multisensor systems to try and achieve optimal control and sensing [Durrant-Whyte 2003]. More broadly, this approach is starting to see proponents in scientific computing as well [Emery 2001, Emery 2002]. We believe that it can be applied to general large software systems; however, in this paper, we discuss how it might be used in engineering drawing analysis.

2.0 Engineering Drawing Analysis with NDAS

We have shown that a structural model may be realized through a set of software agents acting independently and in parallel to ultimately achieve a coherent analysis of CAD drawings [Henderson2003a,Henderson2003b,Henderson2003c,Swaminathan2002]. The high-level goals of the analysis are to:

- Understand legacy drawings.
- Acquire context of field and engineering data.
- Respond to external analysis, user input.
- Integrate drawing analysis in redesign.

NDAS allows multiple agents to produce the same type of data, for example, line segments or text. Other agents which use these entities as inputs may choose from any or all of the available sets of data to produce their own data. Moreover, even a single agent can produce its output using multiple thresholds, or can be asked by another agent to produce output with a given set of control parameters. This allows people or more sophisticated agents to explore the entire parameter space of all the agents involved in the analysis.

The mechanism to handle the combinatorial explosion of data is tied to the structural definition of the engineering drawing, and uses syntactic analysis to eliminate redundant comparisons. This symbolic redundancy calculation uses both the syntax of structural re-write rules, as well as parsing constraints on the tokens generated from the image analysis to achieve orders of magnitude reductions in the possible combinations of tokens. However, NDAS to date has done little else to incorporate or exploit the wealth of other knowledge involved in understanding engineering drawings.

3.0 Knowledge about Engineering Drawing Analysis

Figure 4 shows the sequence of paper drawing creation and exploitation with which we are concerned. We consider knowledge about physical processes, image analysis and document interpretation.

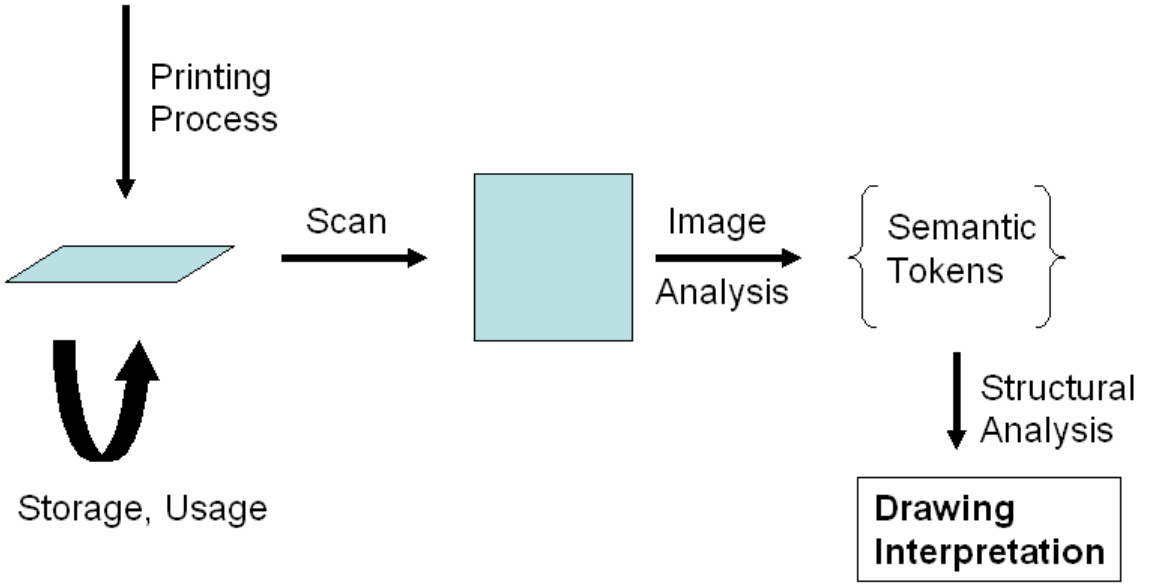

Figure 4. Engineering Drawing Analysis Process

3.1 Physical Processes

It is important to capture knowledge about all aspects of the physical processes involved. For example, printing gives rise to certain errors that can influence the image analysis and subsequent interpretation. During storage and usage, it is possible to introduce lines by folding or creasing, or to obscure lines and text by stains, writing or damage to the paper. Scanning is itself a physical process subject to motion blur, lighting, scale and other perturbations. Good understanding is necessary for robust and correct analysis, and a good synthesis model will allow the controlled creation of test data with defects.

3.2 Image Analysis

Discrete geometry plays a large role in the analysis of engineering drawings, and involves abstract notions, including:

- 0-dimensional objects: isolated points, corners, branch points, end points, etc.
  and relations: distance, near, same kind, etc.
- 1-dimensional objects: line segments, straight segments, circles, boxes, etc.
  and relations: collinear, parallel, perpendicular, neighbor, closed, etc.
- 2-dimensional objects: blobs (e.g., arrowheads)
  and relations: above, left of, touches, occludes, etc.

Moreover, these notions cannot be implemented perfectly, and it is important to know how the realizations differ from the ideal (e.g., what's the threshold for parallel?). Even

more important is the relation of these notions and their recovered approximations to the semantic tokens which form the basis for the structural analysis.

3.3 Structural Analysis

The structure of the drawing is given by a set of tokens (e.g., line segments, text, pointers, graphics, manufacturing symbols, etc.) and the relations that hold between them. Thus, the production of the tokens is crucial, and interpretation problems arise when tokens are missing, broken into parts, or falsely reported. The relations between the tokens need to be clearly defined, as well as the amount of divergence from the ideal. Context of various sorts is also extremely important, and ranges from geometric frame (which way is up?) to drawing type (detail drawing, assembly description, manufacturing constraint requirements, etc.).

These various sets of knowledge are usually not made explicit, either during the development of the system or for exploitation during an analysis. We are interested in answering the following kinds of questions:

1. How can this knowledge be made explicit?
2. How can the differences between the ideal and the implementations be given?
3. Can some of the knowledge (ideal or performance) be learned by the agents?
4. How can people interact with this knowledge to understand why the system does something or to change how the system does it?
5. How can the knowledge be exploited during the analysis of one image; over a set of related images; over various projects, i.e., in order to gain and record more insight on engineering drawing analysis in the log term.

It is essential to answer these questions so that the system can improve over time, and be more effectively understood and exploited by its human operators.

4.0 Proposed Method

We propose the following approach to address this problem:

1. Give a specification for the ideal.
2. Give ways that implementation can differ from ideal.
3. Give a measure of the difference.
4. For every analysis, keep a record of the ideal referent, actual produced, difference measure and analysis parameters.

For example, parallel segments should ideally have 0 degrees difference in angle. A difference measure would be the actual difference in angle, or some monotonically increasing function (square, exponential, etc.). Various implementations would carry different information; e.g., if parallel is computed from the two segment angles, then an angle difference threshold would be kept; if parallel is determined by whether the points defining the one segment are all the same distance from the other segment, then the

maximum and minimum distances would be kept. It is possible to have agents for both parallel operators, and the system can decide (based on training or operator feedback) which is better. This goes with our notion to develop a system which allows many different analysis methods in parallel, and from this wealth of data, chooses between them to construct the best interpretation possible.

This approach also fits well with statistical approaches. For example, various information measures can be defined and used to steer the analysis. Once we have established mechanisms for knowledge expression and use, we will explore alternative mechanisms for the exploitation of that knowledge (for example, Durrant-Whyte and colleagues [Durrant-Whyte 2003] have developed methods to maximize information gain with each observation action – this approach might give good results here).

4.1 Knowledge about Engineering Drawings

Let's look in more detail at the knowledge that would be useful in this application. As for engineering drawings per se, Table 1 gives some of the useful information:

Subject	*Issues*	*Form of Knowledge*
Layout	Up/down, text orientation	Semantic network/ grammar
Symbols	Alphabet, digits, special	Dictionary; images; nets
References	Conventions for pointers, names, use of circles, etc.	Semantic net; image features
Characters	Language, numbers, measures	Semantic nets, feature vectors, images
Real world semantics	Manufacturing info, 3D, 2D projections, etc.	Semantic network

Table 1. Types of Knowledge in Engineering Drawing Analysis

As can be seen, most of this knowledge, if it exists, might be better expressed as a semantic network or in vector or image form. We are currently investigating the construction of a domain ontology, and hope to base it on the Standard Upper Merged Ontology (or SUMO) [Niles 2001]. In this way, we make the assumptions of the agents explicit, and provide a SUO-KIF [SUO-KIF] interface to other users and systems. However, it must be pointed out that our domain requires analogical forms of knowledge as well, including: images, 3D data sets from Coordinate Measurement Machines or laser scanners, etc. Some axiomatizations and ontologies for geometry exist (e.g., see [Asher 1995, Pratt 1997, and Tarski 1956], but their usefulness in this context remains to be seen.

Image analysis has its own set of concerns, including:

- 1D segments,
- pixels (digitization),
- relations, and
- realization of geometry.

Algorithms include: thresholding foreground/background, thinning, segment extraction, straight segment determination, geometric objects detection (e.g., boxes, circles), pointer detection, and text detection. Each of these must deal with thresholds, sensitivity analysis, quality estimates, complexity, and robustness with respect to other algorithms.

Finally, knowledge about goals may influence agent actions; here are some goals that the system may be asked to achieve:

- Find part name.
- Find label information.
- Extract references to other parts.
- Get dimension information for specific part features.
- Determine manufacturing constraints.
- Determine safety or other special descriptions in the text.

These various forms of knowledge should not be static, but should be adjustable over time, as more experience is gained. For example, the use of pointers in drawings can be quite creative, and these need to be cataloged and accounted for. At a minimum, threshold exploration should be possible and recorded.

Another issue is what needs to be communicated between agents (and/or users) which includes at least the following:

- the goal,
- the results of an agent; this includes the info produced, info about the production of the info, and some quality of result measures, and
- feedback to an agent; for example, "this data resulted in no solution" or "parallel constraint needs to be tighter" or "your results are not necessary for this goal"; this last feedback would lead to greater efficiency if agents know when they are unnecessary.

For example, the circle agent uses simple 1D segments (a set of pixels) as input and checks if the set of pixels forms a circle. However, this agent is not necessary for the analysis of the title block of a drawing; it is essential, however, for full drawing analysis. The result of the analysis is a list of point sets determined to constitute circles, and for each circle gives the center and radius, the segments or pixels involved, a quality measure of the circle, and the resources used to produce the circle (e.g., data files used, space and time complexity, etc.). It may also be necessary to include information about why the thresholds and parameters were selected. As an example of feedback that the circle agent may want to provide, suppose that it uses straight line segments to detect circles (i.e., a set of straight line segments form a circle if they are connected end to end and their points do not lie to far from a circle); if the straight segments are fit too coarsely, they may not form a circle, when in fact the pixel data would permit a circle. Thus, the circle agent may want to ask the segment agent to re-fit the data with a tighter linear fit threshold.

As a starting point, we have investigated the knowledge about thresholds and their interplay between entities produced, consumed, and the semantic tokens generated. Figure 5 shows the image analysis part of NDAS. Threshold utilization is indicated by the circled numbers. Table 2 gives the meanings of the thresholds.

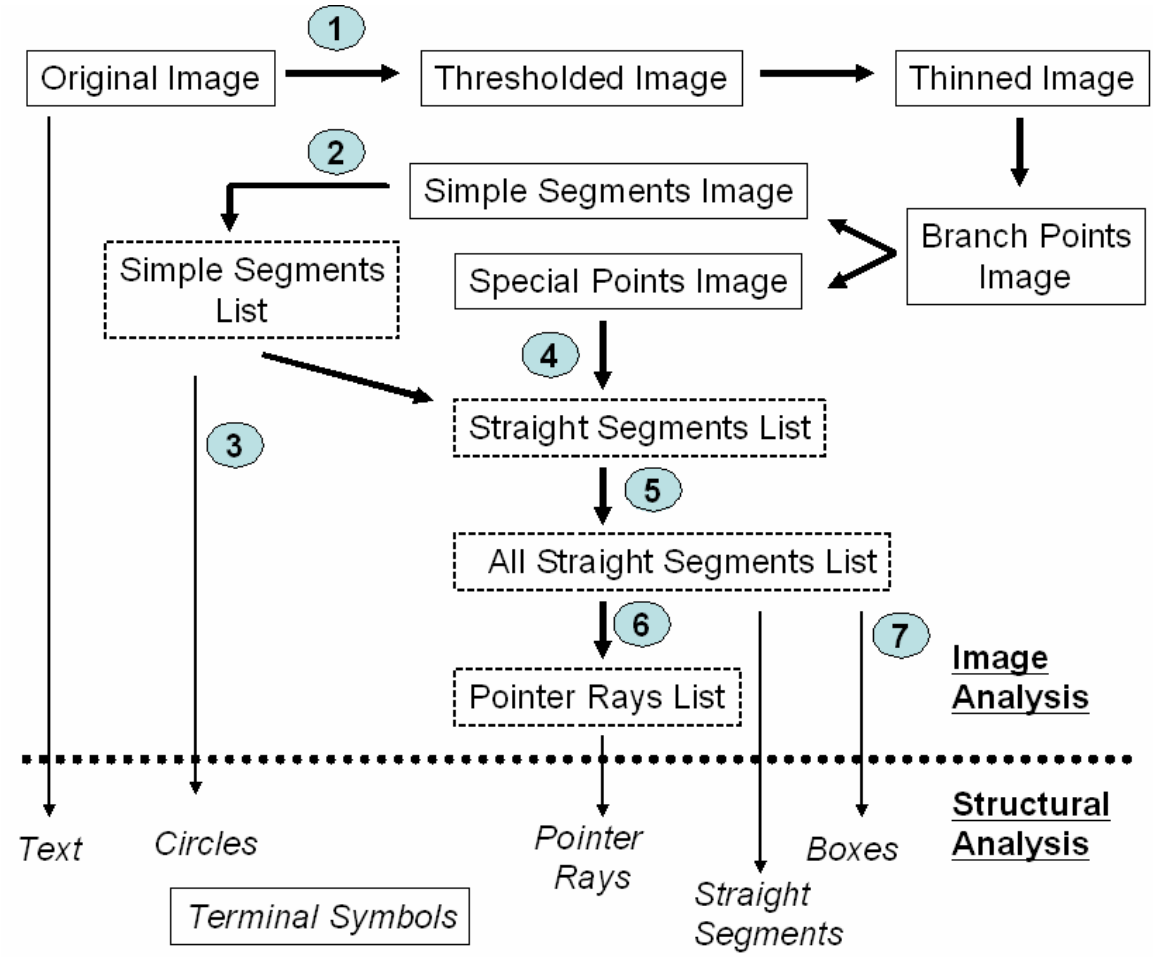

Figure 5. The Image Analysis Agents and the flow of data between them.

Circle no. in Figure 5	Thresholds/parameters	Related Impact
1	foreground/background	extra/missing pixels; connectivity of segments
2	pixel curvature parameters	corner detection, straight segment endpoints
3	circle fit parameters	circle detection, reference detection
4	line fit parameters	number and quality of segments
5	collinear; line fit parameters	large-scale object detection
6	endpoint distances; segment lengths; collinear	pointer ray detection, dimension analysis, references

7	segment length, separation threshold, parallel, perpendicular, duplicate threshold	box detection; document block analysis, text analysis

Table 2. Image analysis agent thresholds and parameters and their impact.

We term the image analysis knowledge given in Figure 5 as *superficial knowledge*, since it concerns only the external relations between the agents and their products. Thus, information about the organization of modules, which use the data from which others, their production information, the quality measures on the data, the amount and trends of data production, and the system activity all fall under this term.

Opposed to that is *deep knowledge*, which concerns the inner workings and decision rationales for implementations, threshold settings, etc. This then includes an ideal description of the process, an explanation set of how the implementation differs from the ideal, a characterization of the likelihood of the variances from the ideal, and the relation of the variations to further processing, including semantic token (terminal symbol) creation and semantic analysis.

To clarify these ideas, let's consider the image thinning process. There is a mathematical notion of a valid thinning operator on point sets, but implementations may vary from this ideal for many reasons and with different implications. Consider the four versions of the thinned partial segment in Figure 6.

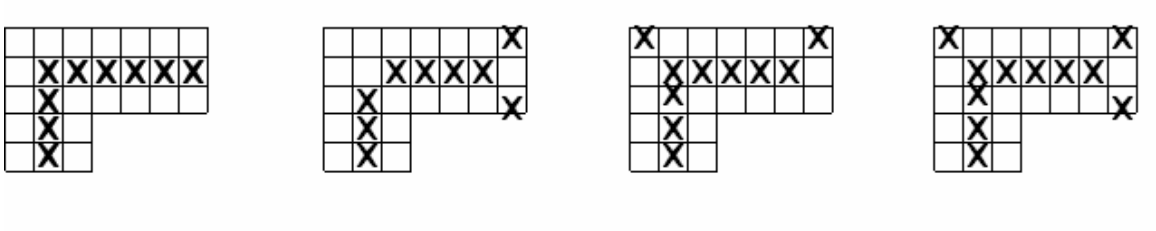

Figure 6. Four variations of a thinning operation.

Which of these is produced may significantly impact later analysis; e.g., abstractions based on end point, branch point and straight line segment relations can be radically different. Figure 7 shows a set of relation graphs for the thinned objects above.

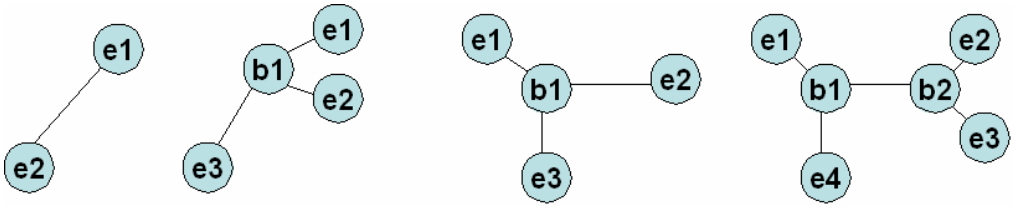

Figure 7. Graphs of connectivity between end points (e1,e2,…), and branch points (b1,b2,…) of thinned objects from Figure 6. (Between every pair of nodes is a, not necessarily straight, line segment.)

As can be seen, the number of line segments, the position of their endpoints and the geometric relations between them (distance, parallel, etc.) can all be greatly affected by these differences in the thinning. Thus, what might be viewed as a *local* or *minor* algorithm issue, may lead to a radical change in performance (including increase in complexity if lots of small segments are generated) if there is no knowledge of how one process impacts other processes through shared analysis objects. It is of great interest to understand these relationships, and to declare them when the system is designed and implemented, but even if that is not possible or accurate (the developers may not understand the impact!), it would be good to allow the system to determine some of this knowledge as various algorithms are executed with different parameter values.

In terms of the thinning operation, we might proceed as follows:

Ideal definition of thinning: One example of this is the medial axis transform [Blum 1973]. This is the set of points such that a circle centered at the point touches the boundary of the object in at least two distinct places.

Algorithm difference from ideal: the algorithm may approximate the ideal definition in order to reduce computational complexity and because the ideal notions don't apply perfectly to digital geometry. The following differences may occur:

1. Ends of segments may be fragmented.
2. Corner regions of segment may be fragmented
3. Medial axis may be displaced from actual corner location.

Measures of difference: Several possibilities exist to measure the three differences listed above. There are two levels of measure, however. First, it is of interest to measure individual errors in terms of the number of extra segments produced, or the distance a thinned set is displaced from a point of interest in the original point set. In addition, it is useful to have some statistics over the whole population. For example, this might be either (1) a likelihood on the number of extra fragments expressed as a mean and variance or in other forms, or as a function of the original segments, the features of the segment or those of the thinned segment. For example, if the thinned segment is perfectly straight, then it is most likely that it perfectly represents the ideal.

Model Calibration

This approach also affords the opportunity to generate controlled test data and to obtain very good estimates of how well the model works. This would work as follows. A CAD model is developed for some artifact. This is then printed, possibly submitted to various degradations, and then scanned. Since the actual CAD model is available, it is possible to know the perfect set of pixels that should have been printed and then scanned. Once the thinned objects are determined, they can be compared to the perfect set of thinned objects, etc. We call this *model calibration*, as it can be used to determine how well the process model measures the true state of affairs.

5.0 Examples

We have performed many experiments with the image analysis part of NDAS. Figure 8 shows part of a typical scanned engineering drawing, and the thinned image in Figure 9.

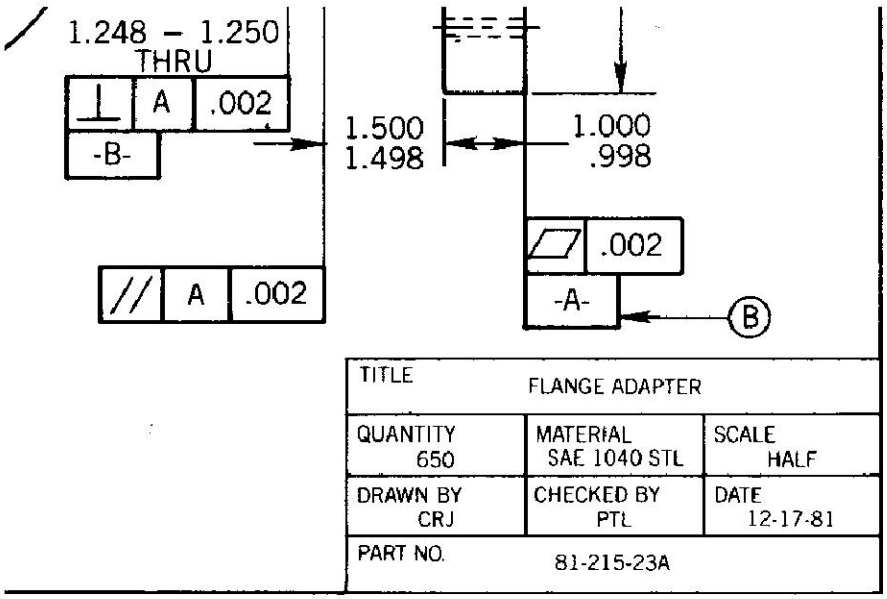

Figure 8. Part of a typical scanned engineering drawing.

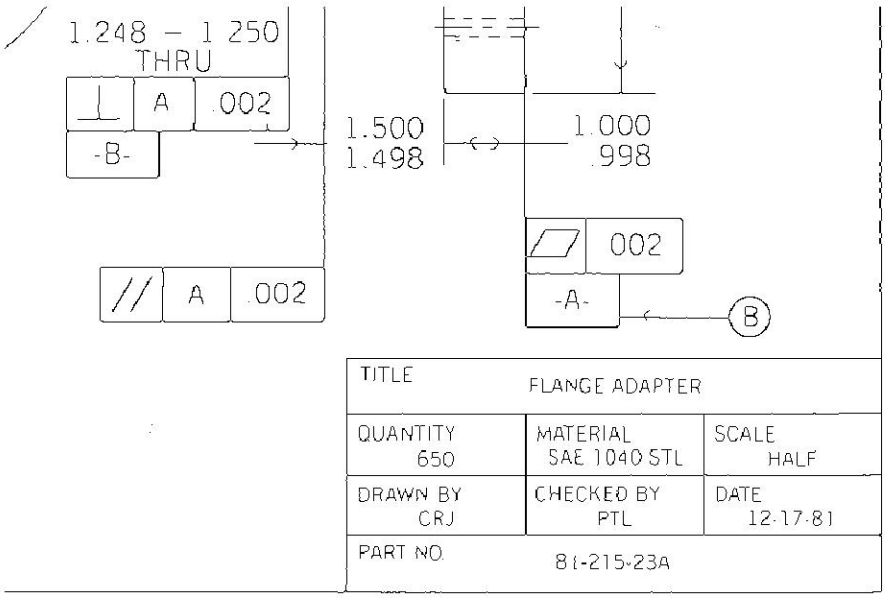

Figure 9. The thinned version of the image in Figure 8.

One thing to notice is how the arrowheads in the original image have been changed into line segments. Also, the corners of boxes have been displaced several pixels from where they should ideally be located. Figure 10 shows the boxes detected in the image, so it can be seen that it is still possible to find them, however, this may cost a great deal in computational or algorithmic complexity, or the algorithms may in fact be tuned for one image and not work very well on another. This is the kind of knowledge we would like to gain and record for better exploitation of the system,

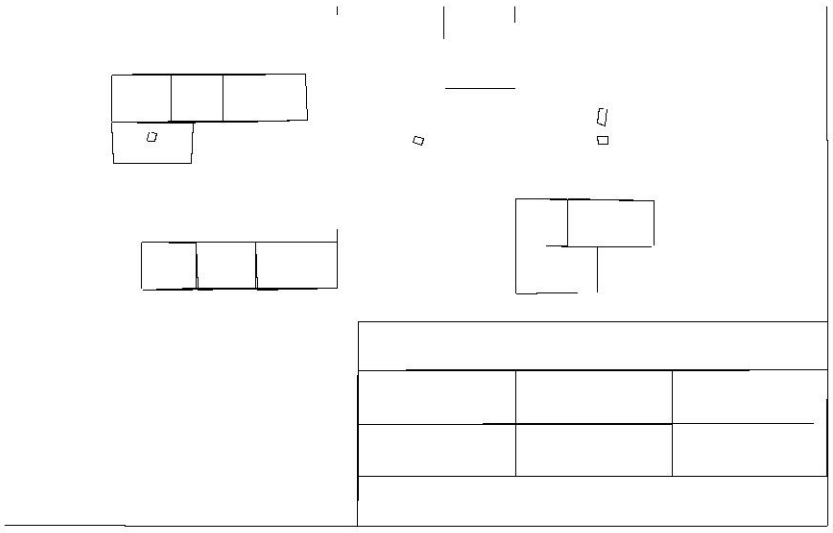

Figure 10. Boxes found based on the thinned image in Figure 9.

A change in thresholds of 2 pixels in length, and parallel segment overlap of 10 % more, results in missed boxes.

Now consider in detail the kind of information to be gathered and characterized about the thinning algorithm. (Note: the ground truth locations of corner points for the boxes in the image have been given by hand.) We would like to model the impact of the algorithm on:

- True corner existence.
- Segment recovery (particularly, endpoint location).
- Box detection and localization.

For the image in Figure 8, the histogram in Figure 11 shows the ideal corner points distance from the thinned pixel set:

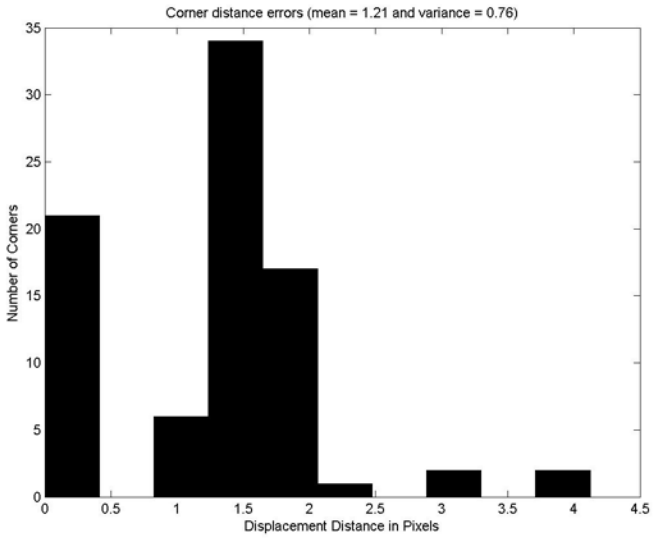

Figure 11. Histogram of ideal corner point displacement in thinned image.

The segment endpoint distance histogram is given in Figure 12.

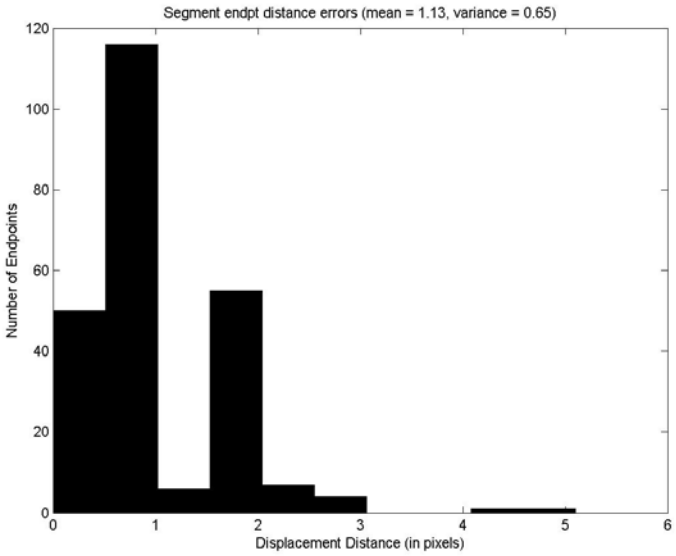

Figure 12. Histogram of Ideal Box Segment Endpoint Distances from Detected.

This data is for ideal segments such that there exists a segment produced by the image analysis whose endpoints are within 10 pixels of the ideal segment. (The number of missing segments is eight; i.e., eight ideal segments have no counterpart in the segments extracted from the image.)

Figure 13 gives the histogram for the distance of ideal box point corners from detected data.

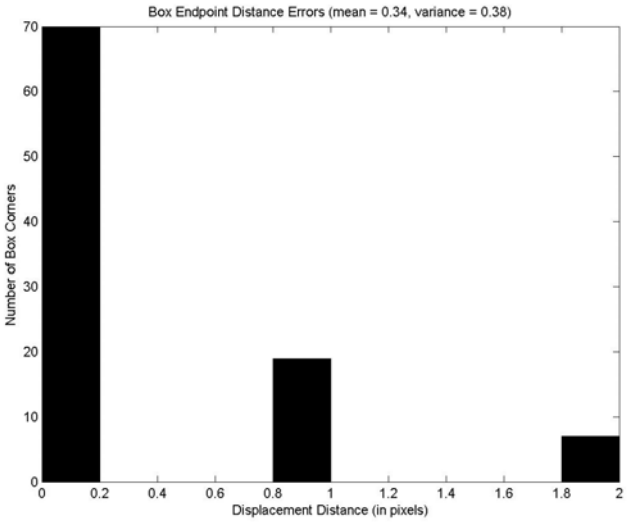

Figure 13. Histogram of Ideal Box Corner Distances from Detected.

All the ideal boxes were found, and the error is very low. The data shows that the box detector algorithm is insensitive to endpoint displacement in the thinning and segment detection algorithms. Moreover, even a missing segment does not preclude the detection of a box, so long as there is a reasonably long segment found on each side. This depends, of course, on the thresholds in the box detector agent. (Also, note that the box agent discovers 'boxes' in the image that are not included in the ideal set; e.g., the upper part of a letter 'B' in the text.)

6.0 Conclusions and Vision

We have to this point tried to convey a sense of the kinds of knowledge that interest us, and how they can be used in engineering drawing analysis. We have also given a high-level summary of our proposed theoretical framework (see Section 2), and have enumerated some advantages that may result from this approach.

A larger issue is the use of other types of information in the reverse engineering scenario; e.g., 3D scanner data, photos, manufacturing information, etc. These analogical forms of data must be integrated into the re-design as well, and this should be done so as to allow rapid iteration, and fast exploration of the design space.

We believe that the state estimation control framework espoused here may be quite useful in the design, development and exploitation of general purpose large software systems, particularly, those characterized by parameterized modules, objects or agents interacting rather loosely and which allow multiple passes through the processing with different sets of parameters. We hope to have some experience with a prototype system by the time of the workshop.

Acknowledgment: This work was supported in part by ARO grant number DAAD19-01-1-0013. I would also like to thank my colleagues for their comments and interaction, and in particular, Jim de St. Germain.

7.0 References

[Asher 1995] N. Asher and L. Vieu, "Toward a Geometry of Common Sense: A Semantics and a Complete Axiomatization of Mereotopology," IJCAI, Montreal, 1995.

[Blum 1973] Harry Blum, "Biological Shape and Visual Science," Journal of Theoretical Biology, 38:205-287, 1973.

[Catlin 1989] Donald E. Catlin, Estimation, Control, and the Discrete Kalman Filter, Springer-Verlag, New York, 1989.

[Durrant-Whyte 2003] Ben Grocholsky, Alexei Makarenko, Tobias Kaupp and Hugh Durrant-Whyte, "Scalable Control of Decentralised Sensor Platforms," Proceedings Information Processing in Sensor Networks, Palo Alto, pp. 96-112, April 2003.

[Emery 2001] A.F. Emery, "Using the Concept of Information to Optimally Design Experiments with Uncertain Parameters," ASME Journal of Heat Transfer, Vol. 123, pp. 593-60, 2001.

[Emery 2002] A.F. Emery, "The Relationship between Information, Sampling Rates, and Parameter Estimation Models," Journal of Heat Transfer, Vol. 124, No. 6, pp. 1192-1199, 2002.

[Henderson 2003a] Thomas C. Henderson and Lavanya Swaminathan, "NDAS," SDUIT, Arlington, VA, May 2003.

[Henderson2003b] Thomas C. Henderson and Lavanya Swaminathan, "Symbolic Pruning in a Structural Approach to Engineering Drawing Analysis," Int Conf on Documant Analysis and Recognition, Edinburgh, Scotland, pp. 180-184, Aug 3-6, 2003.

[Henderson2003c] Thomas C. Henderson and Lavanya Swaminathan, "NDAS," KIMAS, Boston, Oct 1-3, 2003.

[Niles 2001] Ian Niles and Adam Pease, "Towards a Standard Upper Ontology," *Proceedings of the 2nd Internal Conference on Formal Ontology in Information Systems (FOIS-2001)*, 2001.

[Pratt 1997] Ian Pratt and Oliver Lemon, "Ontologies for Plane, Polygonal Mereotopology," University of Manchester Technical Report, UMCS-97-1-1, 1997.

[SUO-KIF] "Standard Upper Ontology Knowledge Interchange Format," see the formal standards document at http://suo.ieee.org/suo-kif.html

[Swaminathan2002] Lavanya Swaminathan, "Agent Based Engineering Drawing Analysis," MS Thesis, University of Utah, December 2002.

[Tarski 1956] A. Tarski, “Foundations of the Geometry of Solids,” in Logic, Semantics, Mathematics, (ed) J. Corcoran,, Oxford University Press, Oxford, pp. 24-30,1956.