

TEXT ANALYSIS FOR CONSTRUCTING DESIGN REPRESENTATIONS

ANDY DONG, ALICE M AGOGINO
University of California at Berkeley
Department of Mechanical Engineering
5136 Etcheverry Hall
Berkeley, CA 94720-1740
adong@jerry.ME.Berkeley.EDU, aagogino@euler.ME.Berkeley.EDU

Abstract. An emerging model in concurrent product design and manufacturing is the federation of workgroups across traditional functional “silos.” Along with the benefits of this concurrency comes the complexity of sharing and accessing design information. The primary challenge in sharing design information across functional workgroups lies in reducing the complex expressions of associations between design elements. Collaborative design systems have addressed this problem from the perspective of formalizing a shared ontology or product model. We share the perspective that the design model and ontology are an expression of the “meaning” of the design and provide a means by which information sharing in design may be achieved. However, in many design cases, formalizing an ontology before the design begins, establishing the knowledge sharing agreements or mapping out the design hierarchy is potentially more expensive than the design itself. This paper introduces a technique for inducing a representation of the design based upon the syntactic patterns contained in the corpus of design documents. The association between the design and the representation for the design is captured by basing the representation on terminological patterns in the design text. In the first stage, we create a “dictionary” of noun-phrases found in the text corpus based upon a measurement of the content carrying power of the phrase. In the second stage, we cluster the words to discover inter-term dependencies and build a Bayesian belief network which describes a conceptual hierarchy specific to the domain of the design. We integrate the design document learning system with an agent-based collaborative design system for fetching design information based on the “smart drawings” paradigm.

1. Motivation

The design of complex mechanical systems requires an intimate understanding of the interactions among the different disciplines and subsystems so that cross-

disciplinary tradeoffs can be made. Any change that might have been precipitated explicitly by modifying a requirement or implicitly by observing a failed simulation will propagate a chain of interaction between designers, manufacturing engineers, process planning engineers, and sales and marketing professionals. Knowing the role of individual functional and physical design elements and their association to other elements in the overall design helps the product design team “understand” the design from the perspective of other members.

In reality, to “know” the interaction between design elements, designers expend a considerable amount of effort in accessing and absorbing design information. One can characterize this scenario roughly as a three-step process. First, the designer looks for possible related elements such as inter-dependent design functions or physical components. Next, the designer analyzes and interprets the relations between them, relations that might be explicitly stated in mathematical equations, rules, or implied by design standards and “best-practices.” Finally, the designer decides which of the associations is plausible in some sense. If there is no reason to reject or defer, then the association is accepted (Baya et al., 1993). Unfortunately, few CAD applications have begun to address the problem of reducing the time designers spend understanding the design, including absorbing design information, keeping up with design changes and reconciling problems or sharing information (Toye et al., 1993). According to Akman (1994), only systems which embed advanced reasoning capabilities will be able to deal with the complexity arising from the management of large quantities of design data.

Since this assessment is typically achieved by reading natural language texts such as memos and design specifications associated with the design model (Ullman, 88), we would like to build a program to automate this process. This research introduces an automated technique to acquire a representation of the design based upon contextual clues in the design documents. By allowing the current context of the design to influence the representation, we eliminate the *a priori* determination of a structured hierarchy or design language and permit dynamic updating of the design vocabulary.

The research was motivated by a desire to take advantage of existing design information to assist in collaborative design. Current CAD tools adequately capture the final design details such as specifications and analysis results. Still, we need to develop tools that learn the interconnections between well-documented design elements so that federated workgroups can have access to relevant information without necessarily having to be an expert in each area of the design. The underlying aim of the research then is to discover the terminological patterns in design text as a basis for constructing a meaningful engineering model of the design.

2. Prior Research

The kernel of design information systems is the ontology which describes the product model. The ontology is a repository of information and provides a means by which concurrency in design may be achieved. The evolving STEP standard (ISO CD 10303-1) highlights the thrust towards product modeling and a common ontology in product models. Product modeling-based systems have been quite successful at setting up complex rules which describe in detail the possible underlying structures of a design (Wong and Sriram, 1993) (Szykman and Cagan, 1992); at the same time ontology-based systems are trying to define semantic relations and to model the functional and behavioral structures underlying the synthesis of a design for representing stereotypical information (Olsen et al., 1995) (Shah, 1993). A similar design-document learning system to the one proposed here is being pursued by Reich (1993) except that the relationships between words are not learned but rather negotiated by the designers. How the words fit together into a structure communicated an idea.

We agree that an ontology provides a means for sharing information. However, the approach presented in this research differs from that taken by other researchers in the design community who developed specialized grammars and shared ontologies (Gruber, 1992) or product-models in that it derives from the design documents. Information models should capture and represent product information to give the reader an “understanding” of the design the model represents. But they must also be dynamic to reflect the evolutionary nature of design. Even though one could argue that the addition of new ontology and negotiated agreements makes the ontology-based or product modeling-based systems dynamic with the design, since the “meaning” of design elements changes with an evolving design, modifying the model or adding new ontology to reflect the changes in real-time might be difficult. In fact, the evolutionary and uncertain nature of design require representations that operate on meaning, not expression (Wood and Agogino, 1995).

Part of the problem of these systems is that they assume that the “meaning” of a design could be computed as a function of the constituents. To “understand” a design, designers must take advantage of a variety of mechanisms that use all sorts of knowledge to fill in any necessary information. In making a computer model of design knowledge, this presents a serious problem. On the one hand, it is impossible to isolate all aspects of domain-dependent knowledge from the others. On the other hand, it is clearly undesirable to give the program all the knowledge related to the design. In this research, the dilemma is resolved by inferring plausible conclusions by relating the various elements of the design using the design documents themselves as a complete and accurate representation of the current state of the design.

We propose the architecture of an intelligent agent in a collaborative design environment which dynamically learns the current status of the design. One application of the agent is the retrieval of relevant information to the current needs

of the designer. The system achieves the learning and understanding of the design using the design documents as the “model of the world.” We present a theory of design discourse as a theoretical premise for generating a model of the design based on the design documents, and illustrate how to integrate learning the design within a collaborative design framework for bringing relevant design information to the decision-maker based on the “smart drawings” system presented in a prior paper (Dong et al., 1995).

3. Methodology

3.1. GENERAL THEORY

In discourse, people take advantage of a variety of mechanisms that depend upon the existence of an intelligent hearer who will use all sorts of knowledge to fill in any necessary information (Wilensky, 1983). To make an intelligent agent understand the design as communicated by the designers through design documents, then, we must construct a framework within which the agent has a sufficient search space to formulate an adequate understanding of the design (Dong et al., 1995). In order for the agent to fill in necessary information regarding the design, though, it must learn the connections between the functions or components of the design. Currently, the solution strategy is to have experts construct both the ontology and describe the decomposition of the design to the agent. However, we argue that this information is in fact available and contained in the design documents themselves. Research in full-text retrieval systems (Lewis, Croft and Bhandaru, 1989) verify how certain syntactic patterns in documents refer to meaningful concepts, and how language-oriented techniques for information retrieval can build the relationships between categories, category instances and relations of those concepts. These categories define a model of the design. By reading the design documents periodically, the agent excerpts the current connections between the different design elements.

In building this agent, we assume to a first-approximation that the linguistic content (words) of the design documents provide a useful index to the composition and structure of key design concepts at the current state of the design. Second, it is assumed that every statistical association derives from causal interaction; therefore logical coherence is based on statistical coherence. Based upon these assumptions, we propose the following theory of design discourse as the theoretical foundation for the learning algorithm:

A theory of design discourse - The content of design documents is related to a conceptual structure of the design, whose communication comprises the goal of the designer.

The claim is the agent can induce a model of the design, including the functions and components of the design and their relations by learning over the design text

associated with the product. Eastman (1991) identifies several criteria for describing engineering product models: (1) the semantics, which describe the functions, components and attributes of the design; and (2) class structures, which describe both the generalizations (properties relevant to any design element), and the decomposition (how functions and components are inter-related) of the design.

While the product model derived by this method is not the same as that proposed by Eastman, these criteria serve as a guideline for learning the design. In essence, the sequence of operations in the program are (as shown in Figure 1) to: (1) extract the natural language text annotations to CAD drawings to excerpt the semantics of the design; (2) generate the class structures describing which properties are relevant to any function of a design using clustering; and (3) build a decomposition of the design which in this method is accomplished with belief networks.

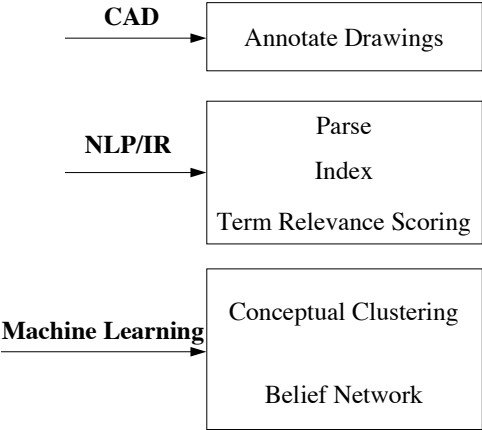


Figure 1. Process Flow Chart - The figure outlines the sequence of operations of the program in learning the content and structure of the design. The research proposes a methodology for annotating CAD documents to create “smart drawings,” techniques for extracting the design vocabulary from the design text using natural language processing and information retrieval, and model learning and inference by applying machine learning.

3.2. STAGE 1: TEXT ANALYSIS

The general method to discover terminological patterns in design documents, which act as a basis for constructing the design model, is to parse the document text, cluster inter-term dependencies and build a conceptual hierarchy.

First, the text was passed through a parser and indexer, freeWAIS-sf¹ (Pfeifer and Huynh, 1994) *waisindex*, which returns a dictionary of every word in the text

¹ One advantage to using a WAIS (Wide Area Information Server) program such as freeWAIS-sf for full-text document parsing, indexing and retrieval is that documents can be queried and retrieved over the Internet using the Z39.50 V2 protocol.

except for common “stop words.”² We then filter this set of terms to develop a set of content-carrying terms. The filtering process is based upon a word score metric similar to that described by the CLARIT method (Evans et al., 1991). The scoring equation is based on the freeWAIS-sf term relevance score (TRS) metric shown in Equation 1. The primary statistics include (1) a frequency count of the number of times the word was encountered in individual documents in the corpus; and (2) an inverted weighted distribution measurement for the number of documents containing a particular term. The idea is that the frequency measurement correlates with the text semantics. Words that occur often in a text are better indicators of what the text is about. More terms can always describe the document concepts better, but too many terms dilute the importance of any individual concept. Thus the distribution (or inverted document frequency) of the terms in the documents captures the intuition that words which have high frequency across documents are “general” in the domain and do not serve as good discriminators of concepts.

$$TRS = \frac{(\log(tf) + 10) \times idf}{\text{number_of_terms_in_a_document}}$$

$$tf = 0.5 + \frac{0.5 \times \sum_{doc} word}{\max \sum_{doc} word}$$

$$idf = \frac{1}{\sum_{doc} word}$$

Equation 1. The freeWAIS-sf TRS Metric - The TRS metric is based upon the term frequency *tf* which counts the number of times the word appears over all documents, the inverted document frequency *idf* which counts the number of documents containing the word (a measurement of distribution) and normalized by the number of terms in a document, to account for the rarity of a word.

The score does not account for variations in author style or the presentation of the text. For example, one might score words which are typed in bold face or italicized or words from more recent design documents higher than others. Other factors such as the person who wrote the document, paragraph headers or document titles could be used as additional word weights; however, the efficacy and numerical value of these weights is difficult to quantify. Further, this complicates the clustering. For example, “recent” terms might be associated by time rather than meaning which violates the purpose of the algorithm. Thus the algorithm has limited sensitivity to the organization and presentation of the text.

Then, the program computes the average score and standard deviation. Words whose score fall above the mean become the inventory of index terms for the corpus, the certified terminology. The system filters out words which are relatively

² Stop words include conjunctions and articles such as “a”, “the,” “since” and other words frequently used in natural language to connect terms but not necessarily to distinguish topics or provide contextual cues for topics.

frequent, have less value in forming good topic discriminators than relatively rare words, and words which are seldom used since they are probably not conditionally dependent upon the concepts described or vice versa. We will explain later why this conditional relevance is important in building a dependency matrix of concepts which forms the basis of the representation.

Finally, based on the set of certified, content-carrying terms, the system determines their contextual similarity by measuring the frequency of occurrence of any two of the certified terms in the documents. That is, the program generates a $n \times n$ matrix, where n is the number of certified words, which scores how “often” the certified words co-occur. This matrix is created by executing a *waisquery* consisting of the query string “[word-A] AND [word-B]”. The query sums up the score for similarity between the query string and the document base. The conjecture is that if the query string appears frequently over the entire document base then the words have a shared contextual dependency. In freeWAIS-sf, document similarity is measured as a vector product formula. The similarity between the query string Q and the document D is given by

$$\text{similarity}(Q, D) = \sum_k (w_{qk} \times w_{dk})$$

Equation 2. The freeWAIS-sf Similarity Metric

where w_{qk} is the weight assigned to term k in the query and w_{dk} is the weight assigned to term k in the document D .

3.3. STAGE 2: CLUSTERING AND INDUCING A BAYESIAN NETWORK

Once the system has developed a prescribed vocabulary, the program maps the terms into context descriptors. The words themselves have no “meaning” outside the context in which they appear. In fact, research in full-text information retrieval has shown that words which appear in the same context tend to have a shared dependency (Gardiner, Riedl and Slagle, 1994). Thus, we need to map the relevance between the assigned terms and the context in which they appear.

For this process, we apply two machine learning techniques. In the first portion, we classify related terms into “conceptual cells” using unsupervised learning. These cells represent terms which are self-similar in the documents. This determination is based upon the observation that terms which appear together (in the same context) in documents typically connote similar meaning (Gardiner, Riedl and Slagle, 1994) (Lewis, Croft and Bhandaru, 1989). Since the matrix measures closeness based on the spread of data or distance between words, a convenient distance-based clustering technique is the K-means algorithm shown in Table 1 (Duda, 1973). The variable x_i is the score in the matrix for the pairwise occurrence of two words in the document collection.

TABLE 1. K Means Algorithm

```

procedure K_MEANS
  ( Initialize the cluster centers  $w_j, j=1, 2, \dots, N_1$  )
  ( repeat
    ; Group the patterns with the closest cluster center
    ( for all  $x_i$  do
      ( Assign  $x_i$  to  $\Theta_{j^*}$ , where  $w_{j^*} = \min_j \|x_i - w_j\|$ 

      endloop )
    ; Compute the sample means
    ( for all  $w_j$  do
      
$$w_j = \frac{1}{m_j} \sum_{x_i \in \Theta_j} x_i$$

      endloop )
    until there is no change in cluster assignments from one iteration to the next )
end ; { K_MEANS }

```

Next, the goal is to obtain a decomposition that explicitly reveals as much information regarding the conditional independence of design elements as possible. The key feature of belief networks is their explicit representation of the conditional independence among events (Pearl, 1988). That is, they can explicitly and compactly represent the dependency of design elements. Topological transformations (through arc reversals and node absorption for example) can answer questions concerning possible causal relations or dependencies between design elements. Since the Bayesian network conveys an intuitive understanding of how the reasoning process works, the designer can also follow the reasoning process of the design based upon the dependencies/independencies of the events to determine how the change in any one element might affect any other element.

The general method for constructing belief networks is to draw arcs from causal nodes to effect nodes and then attach a probability to that arc (Russell and Norvig, 1995). While techniques exist for constructing the most probable belief network B_S given a database D of instances (often called the *maximum a posteriori* structure) based on assumptions of a uniform distribution of belief network structures (Cooper and Herskovits, 1992), the Bayesian Dirichlet likelihood equivalent metric (Heckerman and Geiger, 1995) and minimum description length (Lam and Bacchus, 1993), we generate an initial network using a heuristic approach. We plan to apply one of the metrics to optimize the network locally about a network structure which correctly represents the design.

The heuristic used to construct the Bayesian network is based upon the conjecture that seeing a lower TRS word with respect to a word that it shares contextual similarity causes the system to update the belief that the higher TRS word will appear (Evans et al., 1991). This causal influence and contextual

similarity is found by pairing words with the highest TRS in the co-occurrence matrix. The strategy for building the network is to link the highest associated words in their own clusters first then to link the words between clusters. The algorithm is outlined in the Table 2.

TABLE 2. Network Algorithm - In the first box, the table illustrates the general method for creating belief networks based on expert knowledge. In the bottom box, the table outlines the heuristic algorithm employed by the program.

General Procedure
<ol style="list-style-type: none"> 1. Choose the set of relevant variables X_i that describe the domain 2. Choose an ordering on the variables 3. While there are variables left: <ol style="list-style-type: none"> (a) Pick a variable X_i and add a node to the network for it (b) Set Parents (X_i) to some minimal set of nodes already in the net such that the conditional independence property is satisfied (direct causal influence) (c) Define the conditional probability table for X_i

Network Algorithm
<ol style="list-style-type: none"> 1. Define a variable X_i for each word 2. Order the variables X_i in their respective clusters by ascending TRS 3. While there are variables left in the cluster <ol style="list-style-type: none"> (a) Select the variable X_i with the lowest TRS and add a node to the network for it (b) Set X_j as Parent Of(X_i) where X_i and X_j have the highest similarity in the co-occurrence matrix and $TRS(X_j) > TRS(X_i)$ (c) Select next node in ordering as X_{i+1} and continue; repeat for each cluster 4. Order the clusters by ascending cumulative TRS 5. While there are variables left in the cluster <ol style="list-style-type: none"> (a) Select a variable X_i from the lowest TRS cluster (b) Set X_j as ParentOf(X_i) as the node from the next cluster with the highest similarity in the co-occurrence matrix (c) Select next node in ordering as X_{i+1} and continue; repeat for each node and cluster 6. Define the conditional probability table for X_i

3.4. AGENT ARCHITECTURE FOR DESIGN INFORMATION RETRIEVAL

Figure 2 depicts the agent architecture for learning the design based on the documents. The architecture augments the “smart drawings” system presented in a previous paper (Dong, Agogino, Moore and Woods, 1995).

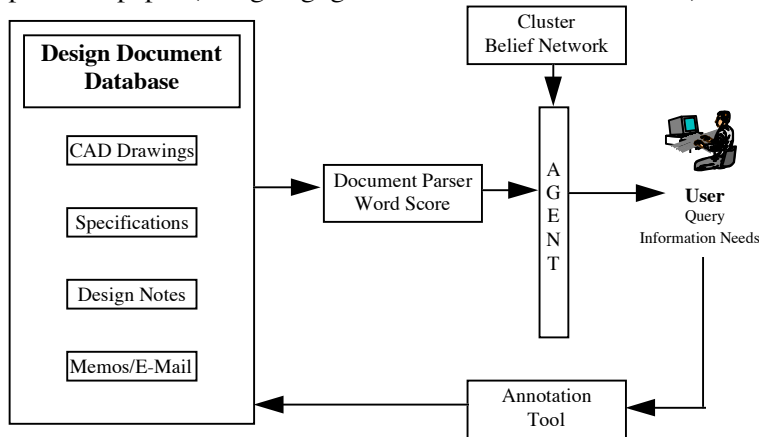


Figure 2. Agent Architecture - The user annotates and adds design documents to the document database. The agent interacts with the document database by parsing and scoring the words in the document. The agent uses the data to create the clustering and belief network to learn the connections between the design elements. The user can then ask for relevant information with respect to current information needs by having the agent search for related design components.

The agent environment consists of the database of design documents, including the CAD drawings, design specifications, design notes and memos and e-mails written between designers. The agent reads the text periodically to generate the list of content-carrying words. By manipulating the list and using the document database for additional data, the agent constructs the inter-term clusters and belief networks to build a model of the design. The model helps the agent to understand the design by finding out what properties are relevant to a function in the design and the decomposition of the design. In response to requests from the user, the agent can retrieve relevant design information.

4. Experimentation and Results

For this project, we created a machine-readable form of *The Mechanical Engineers' Handbook* (Kutz, 1986) which was scanned and run through an optical character recognition (OCR) software (“dirty”)³ to output the final text. The program was then run on the chapters on controller design to derive a model of controllers.

³ “Dirty” OCR refers to documents un-modified after the OCR process, i.e. no spell check.

The cluster results are shown in Figure 3. These clusters indicate which properties are relevant to any particular function or element of the design, giving the agent knowledge of relevant issues in the design. The clusters indicate, for example, that the main content of the documents is the *design* of a *controller* or the *control* of a *system*. The third cluster reveals that the performance of the system is influenced by the *gain* and *order* of the control as well as any *damping* in the system while the sixth cluster indicates that the *position* seems to be the *variable* to be controlled in the system as it is tightly related to the *feedback*, *input* and *output*. One critique of the clusters is that *zero* appears with *performance* and *root* appears with *stable*, whereas it is known that both the *zero* and *root* of the system affect the stability. However, in the document collection, *zero* statistically appears more often with *performance* and *root* with *stable* since, for example, the documents discuss more often that a zero affects steady-state error (a measurement of controller performance) whereas the closed-loop roots determine the stability of the system. The cluster results agree with known knowledge of the relevant properties of the functions and attributes of controllers.

((system design controller control)
 (transfer function time error state signal response plant)
 (zero integral gain order damping performance steady action)
 (stable root frequency process model loop)
 (valve pressure power pneumatic motor displacement)
 (variable value position feedback input output)
 (disturbance diagram constant) ...)

Figure 3. Cluster Results - The cluster results for the chapters on controller design.

Finally, the system generates the belief network shown in Figure 4 and the conditional probability table associated with the network. The states for each of the event nodes (words) are 0, when the word (or design element) is not present in the document, and 1 otherwise. The conditional probability table for the network is based on frequency counts. For example, the probability of the word *controller* co-occurring with the word *design* is given by:

$$P(\text{controller}|\text{design}) = \frac{\# \text{ occurrences controller and design}}{\# \text{ occurrences design}}.$$

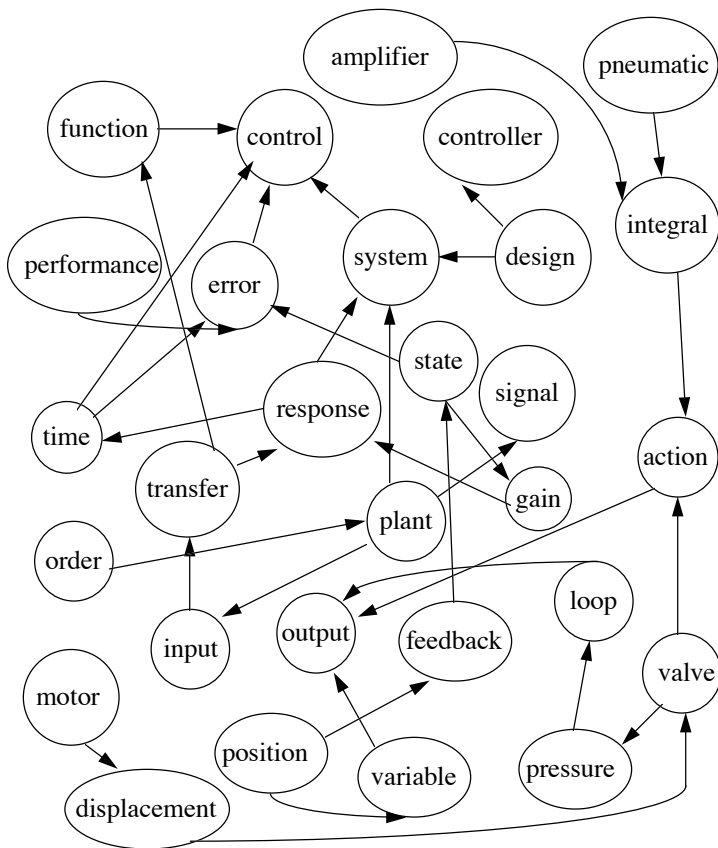


Figure 4. Initial Belief Network (partial) - This figure illustrates parts of the belief network generated using the heuristic algorithm. For purposes of clarity, not all arcs and nodes are shown.

For clarity, not all nodes and their associated arcs are shown in Figure 4. For the nodes shown, the arcs are complete. One can read some interesting inferences off of the network.

The first inference expresses the dependency of the design elements. The expression of dependency describes the decomposition of the design.

1. The *system* to be controlled is characterized by the desired *response* and the *controller design*. The *control* law is conditioned on the *transfer function*, the *error* and the desired *response* of the *system*.

The second inference illustrates the degree of dependency between design elements. These types of inferences relate both information and the degree of relevance based on the amount of evidence available.

2. The concept of *system response* is more dependent upon *gain* in this controller design than the specific *input* criteria.

The third is perhaps the most interesting since it shows how the system could infer the interaction of several elements in the design which produce a certain function. Therefore, if the designer were interested in increasing the pressure in the controller, one of the design elements to modify is the motor and followed by checking if the valve could handle the increased stress. More notable is that without explicitly telling the system these design element connections or the design topic, the system correctly extracted from the text that these chapters discussed controller design using pneumatic devices.

3. The *motor* changes the *displacement* of the *valve* which affects the *pressure*.

While the arc directions could change through topological transformations, the above network and associated inferences illustrate two important ideas. First, inspection of the network indicates that the heuristic generates a network with arcs between elements in the direction of physical causality, as illustrated by the third example in that the *motor* causes *displacement* rather than vice versa. Second, the network illustrates the more important problem of capturing the dependencies between design elements. By capturing these dependencies, the system is more efficient in searching for meaningful and relevant design information. The combination of the cluster information and the belief network augments the search by finding closely associated design elements (cluster information) which may not actually appear in the designer's query while removing less relevant information if less evidence supports the association between the design elements (belief network).

The program was then integrated with a "smart drawing" (Dong et al., 1995) system as shown in Figure 5. Some preliminary tests were conducted to test how well the system learned the design data. One of the tests asked the system to retrieve relevant information to the "Lyapunov stability of the controller."⁴ Based on the clustering results, the program knows that the *roots* of the system affect *stability*, so that documents which discuss *roots* frequently should also be returned and scored high in relevance. By expanding the query to include closely related terms which in this example indicate closely related attributes to the stability of the system, the program can find documents related to *stability* that may not mention the word *stability* in the document. Those elements which have shared dependencies in the belief network are scored higher. Without expanding the query based on the learned data (i.e. a standard freeWAIS query), only documents which frequently discuss both *stability* and *controller* would have scored high. That is, the dynamically learned design structure augmented retrieval to include information not explicitly cast in the query but which should be reported together by virtue of

⁴ One aesthetic limitation of the current implementation is that the user is given only the path to the document rather than the document title, for example. By selecting one of the documents, though, the system automatically brings up a viewer for the document type, such as a text file or AutoCAD drawing.

design dependency. In this case, design documents which discuss any property shown relevant based on the clusters to *stability* or *controller* score high.

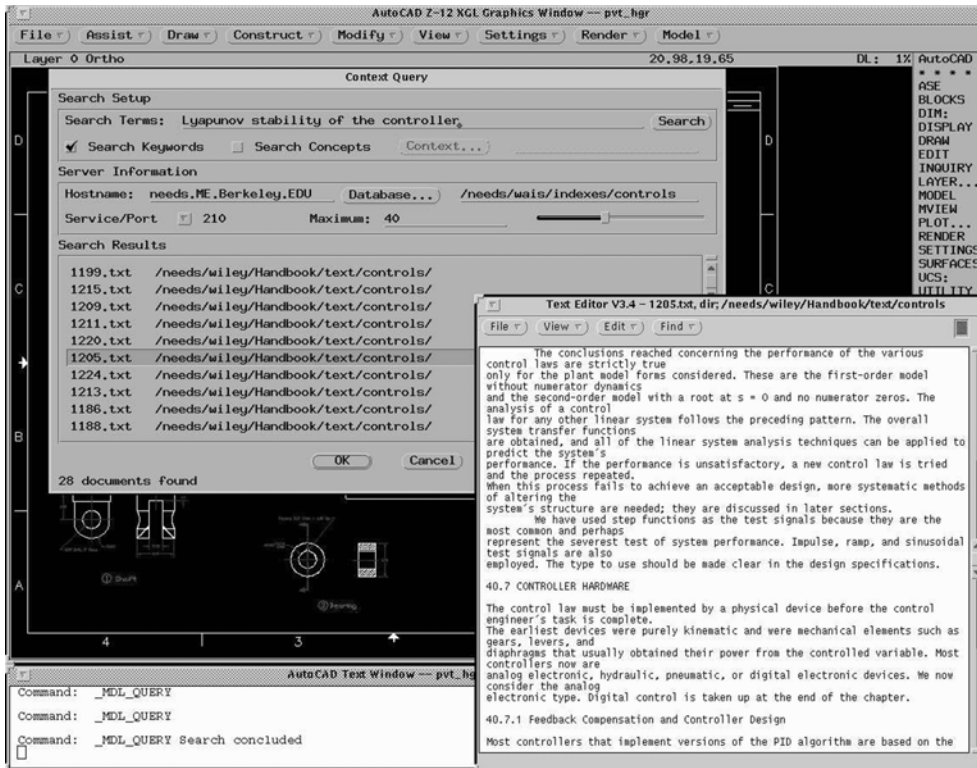


Figure 5. Smart Drawings Desktop - The agent learns the content of the design data based on the design documents using the learning methodology outlined above. Then, when prompted, the agent can retrieve relevant design information based on the current information needs of the designer using the information content of the active document as the query.

The role of the clusters and belief network for design information retrieval is similar to the purpose of the decision dependency network presented by Garcia, Howard and Stefik (1994) in the Explanation interface to their ADD system. The Explanation interface displays related information by retrieving documents that are generally reported together. The key differentiation is that the dependency network is based on a pre-processed parametric design model for the design domain which seems to violate their thesis that the evolution of the design description via documents relates to the evolution of the design. For example, to capture design rationale, ADD prompts the designer for decisions which deviate from the preferred norm. This strategy for design rationale capture suggests that changes in the design affect how the design should have been modeled or parametrized, that, in fact, the design model dynamically evolves with the design.

Systems such as ADD and the one proposed which address the problem of accessing design information by employing a structured design model to augment the retrieval of unstructured design documents can improve recall over those which have only an unstructured model (such as freeWAIS-sf) or only structure (ontology-based systems). However, the important metrics for evaluating these systems should include both the overhead for creating the structured model to account for the dynamic nature of the design as well as the performance in retrieving relevant information compared to baseline systems which employ no structure. The design learning methodology proposed illustrates a preliminary system which addresses both metrics.

While this is only a preliminary test of how well the system learns the design data, what these tests suggest is the ability to augment design information search by finding related information based upon meaning, not just how the search request is expressed in the query. Second, the clustering and belief network open the possibility of organizing the retrieved data in a manner which is more meaningful to the designer than just straight frequency metrics, such as ordering by related concepts. We are currently investigating how to integrate the utility of the information to the designer based on the preferences of the designer and the structure of the design model in the belief network to improve the relevance ranking of the returned information beyond simple frequency count measures. In particular, we are implementing a “concept query” mechanism which more closely analyzes what concepts would be interesting to the designer and the cost of obtaining that information.

5. Summary and Future Directions

This research develops a computable learning method to extract the content of the design model to facilitate information sharing among designers. The premise of the methodology is that the design specifications and solutions as communicated through design documents are related to a model of the design. Certain combinations of the chosen properties of the design give rise to the corresponding combinations of design descriptions in the design text. Therefore, by learning these descriptions (words) through text analysis, the system induces a model of the design. The learning algorithm is based upon natural language processing text analysis to extract content-carrying terms, and then applying techniques from machine learning to cluster inter-dependent terms and decompose the design into dependent elements using belief networks. The model derived for the controller design example was plausible and correct based upon knowledge of controllers.

What this research emphasizes is that CAD systems cannot ignore the communication of design information with respect to the current and relevant information needs of the design based on the annotation of the drawings (Ullman, 1990). That is, the effect of techniques which implement inductive learning techniques such as the one proposed to generate new knowledge structures about

the design rather than techniques that improve the efficiency of problem-solving (explanation-based learning techniques) is tantamount to improving CAD systems. By putting the knowledge of design components in a form in which we can explicitly express the connections between the different parts of the system's knowledge, we enrich the possibility of interaction for collaborative design.

The methodology explored in this paper only begins to explore the possibilities of full-text analysis for deriving a model of the design and its application. For example, one could augment the learned design structure with formally derived ontologies or use the learned structure as the basis for a formal ontology (Gruber, 1993). In particular, enhancing the parsing ability of the program and augmenting the co-occurrence measurement strategy to consider the number of words between two contextually similar words (Grefenstette, 1992) promise to improve the efficiency of the algorithm and achieve finer granularity in representing the design data. We are currently investigating these issues as well as testing the relevance of the learned knowledge in design documents from mechanical engineering design courses.

6. Acknowledgements

The authors would like to acknowledge William H Wood III for his valuable comments and converting the scanned document images into ASCII text, and John Wiley and Sons, Inc. for their permission to scan and OCR the text used for research and testing. We would like to thank in particular our industrial partners, Sun Microsystems, Inc., and Autodesk, Inc., not only for financial and equipment support but for valuable collaboration. This research was sponsored by the NSF Concept Database grant #DDM-9300025.

References

- Akman, Varol, ten Hagen, Paul J. W., and Tomiyama, Tetsuo: 1994, Desirable Functionalities of Intelligent CAD Systems, in *Intelligent Systems in Design and Manufacturing*, Cihan H. Dagli and Andrew Kusiak, (eds.), New York: ASME Press, 119-138.
- Baya, Vinod, Gevins, Jody, Baudin, Catherine, Mabogunje, Ade, Toye, George, and Leifer, Larry: 1992, An Experimental Study of Design Information Reuse, in *Proceedings of the ASME Conference on Design Theory and Methodology*, DE-Vol. 42, 141-147.
- Cooper, Gregory R., and Herskovits, Edward: 1992, A Bayesian Method for the Induction of Probabilistic Networks from Data, *Machine Learning*, Netherlands: Kluwer Academic Publishers, **9**, 309-347.
- Dong, Andy, Agogino, Alice M, Moore, Frank and Woods, Cameron: 1995, Managing design knowledge in enterprise-wide CAD, *Preprints Advances in Formal Design Methods for CAD*, John S. Gero and Fay Sudweeks, (eds.), Key Centre of Design Computing, University of Sydney, Sydney, Australia, 330-347.

- Duda, R.O. and Hart, P.E.: 1973, *Pattern Classification and Scene Analysis*, New York: Wiley.
- Eastman, Charles M., Bond, Alan H., and Chase, Scott C.: 1991, A Formal Approach for Product Model Information, *Research in Engineering Design*, **2**, 65-80.
- David A. Evans, Kimberly Ginther-Webster, Mary Hart, Robert G. Lefferts, and Ira. A. Monarch: 1991, Automatic Indexing Using Selective NLP and First-Order Thesauri, in *Intelligent Text and Image Handling in the Proceedings of a Conference on Intelligent Text and Image Handling 'R'IAO 91*, Barcelona, Spain, A. Lichnerowicz, (ed.), 624-643.
- Garcia, A. Cristina Bicharra, Howard, H. Craig, and Stefik, Mark J.: 1994, Improving design and documentation by using partially automated synthesis, *Artificial Intelligence in Engineering Design and Manufacturing*, Clive L. Dym, (ed.), **6**(1), 335-354.
- Gardiner, David, Riedl, John, and Slagle, James: 1994, TREC-3: Experience with Conceptual Relations in Information Retrieval, *Proceedings of the Third Text Retrieval Conference (TREC-3)*, Gaithersburg, MD, November 2.
- Gruber, Thomas R.: 1993, Toward Principles for the Design of Ontologies Used for Knowledge Sharing, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Guarino and Poli, (eds.), Kluwer Academic Publishers, Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University.
- Gruber, T. R., Tenenbaum, J. M., and Weber, J. C.: 1992, Toward a Knowledge Medium for Collaborative Product Development, in *Proceedings of the Second International Conference in Artificial Intelligence in Design*, Pittsburgh, PA, June 22-25, John S. Gero, (ed.).
- Grefenstette, Gregory: 1992, Use of Syntactic Context to Produce Term Association Lists for Text Retrieval, *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Nicholas Belkin, Peter Ingwersen, and Annelise Mark Pejtersen, (eds.), 89-97.
- Heckerman, David, and Geiger, Dan: 1995, Learning Bayesian Networks, Microsoft Corporation Technical Report MSR-TR-95-02.
- Mechanical engineers' handbook*, Myer Kutz, (ed.), New York: John Wiley and Sons, Inc.
- Lam, Wai, and Bacchus, Fahiem: 1993, Using Causal Information and Local Measures to Learn Bayesian Networks, *Proceedings of the Ninth Conference Uncertainty in Artificial Intelligence*, David Heckerman and Abe Mamdani, (eds.), San Mateo, California: Morgan-Kaufman Publishers, 243-250.
- Lewis, David D., Croft, W. Bruce, and Bhandaru, Nehru: 1989, Language-Oriented Information Retrieval, *International Journal of Intelligent Systems*, John Wiley and Sons, Inc., **4**, 285-318.
- Olsen, Gregory R., Cutkosky, Mark, Tenenbaum, Jay M., and Gruber, Thomas R.: 1995, Collaborative Engineering Based on Knowledge Sharing Agreements, *Concurrent Engineering: Research and Applications*, **2**(3), 145-159.
- Pearl, Judea: 1988, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, San Mateo, California: Morgan Kaufmann Publishers.

- Reich, Yoram, Konda, Suresh L., Levy, Sean N., Monarch, Ira A., and Subrahmanian, Eswaran: 1993, New roles for machine learning in design, *Artificial Intelligence in Engineering*, K.J. MacCallum, G. Rzevski, T. Tomiyama, (eds.), Elsevier Applied Science, **8**, 165-181.
- Russell, Stuart J. and Norvig, Peter: 1995, *Artificial Intelligence: A Modern Approach*, Englewood Cliffs, New Jersey: Prentice Hall.
- Shah, Jami J., Bliznakov, Plamen and Urban, Susan D.: 1993, Development of a Machine Understandable Language for Design Process Representation, in *Proceedings of the ASME Conference on Design Theory and Methodology 1993*, DE-Vol. 53, 15-24.
- Szykman, Simon, and Cagan, Jonathan: 1992, A Computational Framework to Support Design Abstraction, in *Proceedings of the ASME Conference on Design Theory and Methodology*, DE-Vol. 42, 27-39.
- Pfeifer, Ulrich, and Huynh, Tung, FreeWAIS-sf, <ftp://ls6-www.informatik.uni-dortmund.de/pub/wais/freeWAIS-sf.1.0.tgz>.
- Ullman, David G., Wood, Stephen, and Craig, David: 1990, The Importance of Drawing in the Mechanical Design Process, *Computers and Graphics*, **14**(2), pp. 263-274.
- Ullman, David G., Dieterich, Thomas G., and Stauffer, Larry A.: 1988, A Model of the Mechanical Design Process Based on Empirical Data, *Artificial Intelligence in Engineering Design and Manufacturing*, Clive L. Dym, (ed.), **2**(1), 33-52.
- Wilensky, Robert: 1983, *Planning and Understanding: A Computational Approach to Human Reasoning*, Addison-Wesley, Reading, Massachusetts.
- Wood, W. H. and Agogino, Alice M.: 1995, A Case-Based Conceptual Design Information Server, *Journal of Computer Aided Design*, Rajit Gadh (ed).
- Wong, A., and Sriram, D.: 1993, SHARED: An Information Model for Cooperative Product Development, *Research in Engineering Design*, **5**, 21-39.