ELSEVIER

Computer-Aided Design 32 (2000) 119–132

COMPUTER-AIDED DESIGN

www.elsevier.com/locate/cad

Managing digital libraries for computer-aided design

W.C. Regli[1,*], V.A. Cicirello

Geometric and Intelligent Computing Laboratory, Department of Mathematics and Computer Science, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA

Accepted 2 September 1999

Abstract

This paper describes our initial efforts to deploy a digital library to support computer-aided collaborative design. At present, this experimental testbed, *The Engineering Design Knowledge Repository*, is an effort to collect and archive public domain engineering data for use by researchers and engineering professionals. We envision this effort expanding to facilitate collaboration and process archival for distributed design and manufacturing teams.

CAD knowledge-bases are vital to engineers, who search through vast amounts of corporate legacy data and navigate on-line catalogs to retrieve precisely the right components for assembly into new products. This research attempts to begin addressing the critical need for improved computational methods for reasoning about complex geometric and engineering information. In particular, we focus on archival and reuse of design and manufacturing data for mechatronic systems. This paper presents a description of the research problems, an overview of the initial architecture of the testbed and a description of some of our preliminary results on conceptual design and design retrieval.

Keywords: Computer-aided design; Computer-aided engineering; Engineering knowledge-bases; Product data management; World wide web; Network-enabled CAD/CAE

1. Introduction

This paper describes our initial efforts to deploy a digital library that supports engineering design and manufacturing. This experimental testbed, *The Engineering Knowledge Repository*,[2] is an effort to collect and archive public domain engineering data for use by researchers and engineering professionals. This research attempts to begin addressing the critical need for improved computational methods for reasoning about complex geometric and engineering information.

Geometry, in the form of 3D solid models, is ubiquitous in a diverse array of fields including architecture, graphic arts, entertainment, medical informatics, computer-aided design (CAD), and engineering and manufacturing. There are presently over 20 billion (and growing) CAD models[3]—representing a digital library of immense scope, diversity, and importance. In engineering, it is conservatively estimated that more than 75% of design activity comprises case-based design [1]—reuse of previous design knowledge to address a new design problem. As illustrated in Fig. 1, CAD knowledge-bases are vital to engineers, who search through vast amounts of corporate legacy data and navigate on-line catalogs to retrieve precisely the right components for assembly into new products.

In this context, our research is primarily concerned with the libraries to support the design and manufacture of *mechatronic systems*: electro-mechanical systems that combine electronics and information technology to form both functional interaction and spatial integration in components, modules, products, and systems. Typical examples of mechatronic systems include automatic cameras, miniature disk drives, missile seeker heads, and consumer products like CD players, camcorders, and VCRs. These designs include mechanical, electronic and software components. The CAD knowledge includes product data models, such as the CAD model of a missile seeker assembly pictured in Fig. 2, and related metadata (process and assembly plans, documentation, etc.).

The long term goal of our Engineering Knowledge Repository Project is to develop the mathematical foundation and algorithmic tools to support content-based retrieval from large engineering knowledge-bases [2]. In

* Corresponding author. Tel.: +1-215-895-6827; fax: +1-215-895-1582.
E-mail address: regli@.drexel.edu (W.C. Regli).

[1] URL: http://www.mcs.drexel.edu/~wregli.

[2] Formerly the *National Design, Process Planning, and Assembly Repository* at the National Institute of Standards and Technology, now available at http://repos.mcs.drexel.edu

[3] Source: Autodesk, Inc.

0010-4485/00/$ - see front matter
PII: S0010-4485(99)00095-0

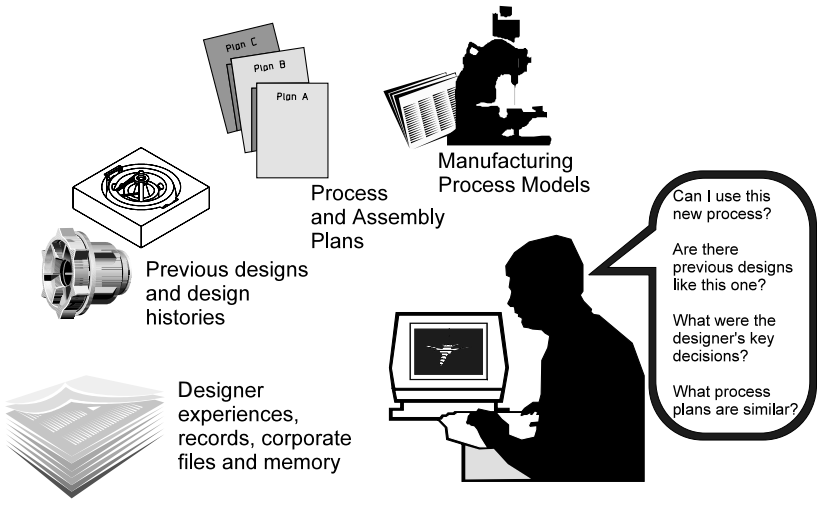


Fig. 1. Scenario for use of Design Repositories: engineers accessing libraries of project data to identify ideas and solutions to new problems.

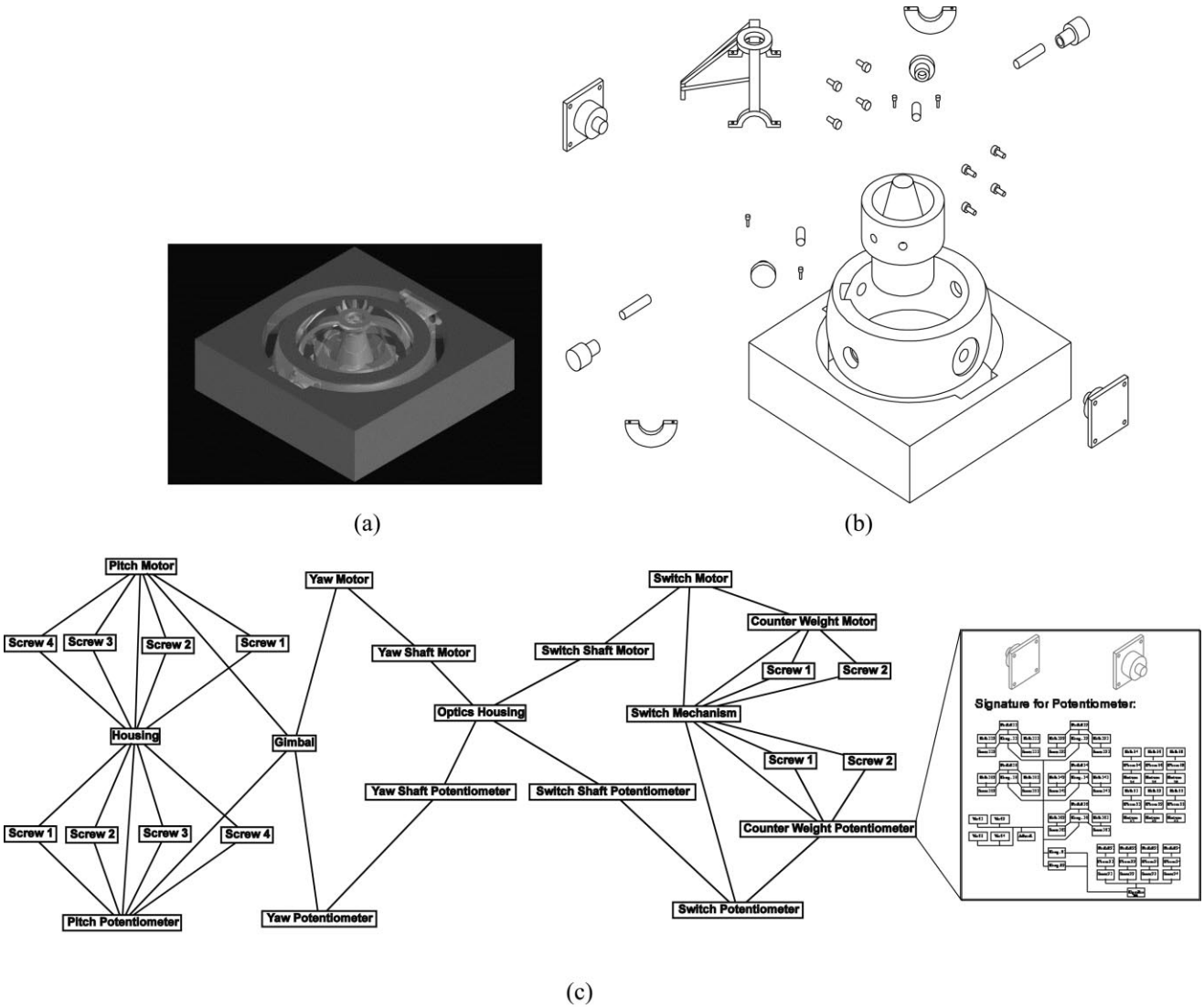


Fig. 2. An example of a mechatronic system: a simplified missile seeker assembly along with its assembly structure.

this effort, we hope to contribute to the understanding of how design knowledge, and the collaborative design process, will archived in future distributed work environments [3].

This paper presents a description of the research problem and an overview of the initial architecture of testbed. Section 2 provides an overview of related work and sets the context for this research. Section 3 introduces a formalization of the problem, defines some of the technical issues, and describes the architecture (both hardware and software) for deploying the testbed. Section 4 presents the components of our current work on this project: system architecture, conceptual query interfaces and a detailed example of part retrieval from databases of solid models. Section 5 discusses the long-term impact of this project and some of our future research directions. Lastly, Section 6 offers some conclusions based on current results.

2. Background and related work

In engineering practice, indexing and storage of parts and part families had been done with group technology coding [4]. Group technology facilitated process planning and cell-based manufacturing by imposing a classification scheme on parts. GT codes specified classes using alphanumeric strings. These techniques were developed prior to the advent of inexpensive computer technology, hence they are not rigorously defined and are intended for human, not machine, interpretation.

Database developers and academic researchers are actively researching how to handle multimedia data [5]. This includes digital libraries [6] and commercial systems [7,8]. In general, the approach has been to develop domain-specific layers to be built on top of a standard relational (or object-oriented) databases—providing an API that is focused on the particular needs of an application area (such as solid modeling and engineering design). For example, Jain et al. and Virage Inc. [8,9] have methods for multimedia data such as pictures (GIF, JEPG, etc.). The approach draws on work in computer vision and is based on the creation of feature vectors from 2D images to capture concepts such as color, density, and intensity patterns. Their work in extending these techniques to 3D CAD data treats the CAD information as sets of 3D feature vectors.

In computer vision research, a fundamental problem is the identification and matching of models of interest in images. There have been many more research efforts in this direction than can be cited here; however, some of those more relevant to the recognition of engineering data and solid models include Refs. [10–14]. These efforts address a different problem than the one introduced in this paper—one in which the main technical challenges focus on the construction of models from image data obtained from cameras and range finders. One example is 3D Base [15] from Dartmouth, which operates by converting CAD models (a solid model or surface model) into an IGES-based neutral file then deriving a voxel (3D grid) representation. The voxels are used to perform pair-wise comparisons among the CAD files using the geometric moments of the voxels and by comparing other per-computed part features (such as surface area). These vision-based matching techniques are highly dependent on data types relating to pixels, range information, color and texture, hence they are not directly applicable to domains in which one has ready access to exact representations of geometry and topology in the form of solid models.

Specific to engineering applications, database management has been an area of active study for many years. Will et al. is pursuing an ontology-based approach to catalogs [16], though at present not tightly coupled with geometric data or with representation of tolerances and features. In the domain of civil engineering and architecture-engineering-construction (AEC), Eastman et al. [17–20] have been developing methods for linking design entities (such as windows and doors) with semantic information to manage design constraints among multiple users operating simultaneously on a project.

While a great deal of work exists on geometric databases and digital libraries for Geographic Information Systems (GIS) [21], relatively little exists on digital libraries for the specific domain of 3D CAD and solid models. Part libraries and catalogs has been an area of active study by the standards community [22,23]. A survey on geometric databases in general can be found in Ref. [24]. Hardwick et al. [25] have merged databases with the Internet and STEP-based standards. Lastly, over the past three years the author has initiated the National Design Repository [26,27], a publicly accessible collection of engineering designs and engineering data.

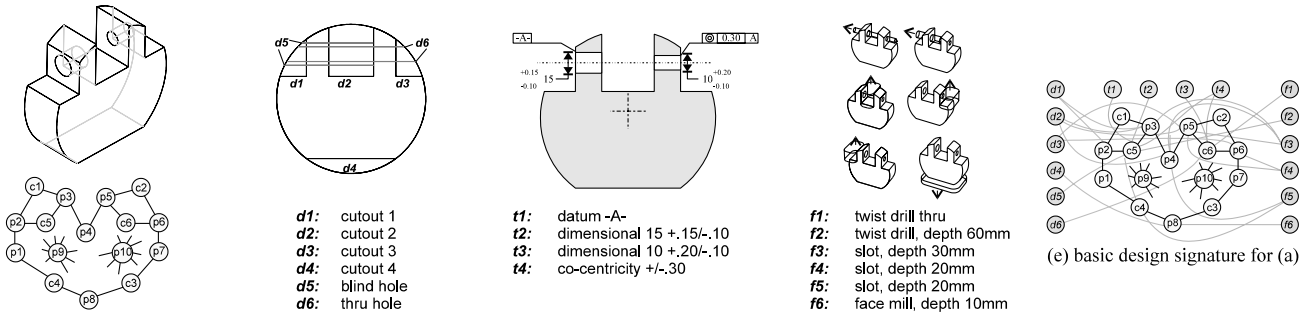
3. Research approach

Previous research in diverse areas such as computer-aided process planning, case-based design, and AI have developed many techniques for modeling the function, intent, and behavior of mechatronic systems. The common elements in the vast majority of these representations are symbolic models of the design, manufacturing operations, plans, etc.

Our approach is to develop structures for capturing the relationships among design attributes (geometry, topology, features) and symbolic data representing other critical engineering and manufacturing data (tolerances, process plans, etc.). This section gives a brief overview of our techniques for associating heterogeneous sources of engineering data with the geometric and topological description of the CAD or solid model defining the mechatronic artifact.

3.1. General problem formalization

We can consider a *design*, D, for a mechatronic artifact (i.e. a part, assembly, etc.) to be defined as a tuple $D =$



(a) part model and BREP (topology only) (b) design features (c) machining tolerances (d) machining features (e) basic design signature for (a)

Fig. 3. Signature structures: an illustration of (a) a design of a simple bracket and its attributes (b)–(d) and a design signature (e).

$\langle P, \mathscr{D}, A, M, L\rangle$ where: P is the geometric/topological model of the artifact consisting of the component's (or the assembly's) boundary representation; $\mathscr{D}$ is a finite set $\mathscr{D} = \{D_1, D_2, \cdots, D_n\}$ of zero or more designs that are subcomponents of $\mathscr{D}$; A is a finite set $A = \{\alpha_1, \alpha_2 \cdots, \alpha_n\}$ of zero or more design attributes. Intuitively, a design attribute is a symbolic piece of information about the design—examples include design and manufacturing features, tolerances, functional models, etc.

$\mathscr{M}$ is a finite set of methods (functions) associated with the design attributes in A,

$$\mathscr{M} = \{M_{\alpha_1,1}, M_{\alpha_1,2} \cdots, M_{\alpha_1,j}, M_{\alpha_2,1}, M_{\alpha_2,2} \cdots, M_{\alpha_2,k} \ldots\}.$$

The functions in $\mathscr{M}$ are specific to the particular attributes being modeled; for example, $alpha_1$ is a engineering tolerance, $M_{\alpha_1,1}$ returns the tolerance value, and $M_{\alpha_1,2}$ the type of the tolerance (e.g. planarity). The attribute set can also represent relationships among components in an assembly design: $alpha_2$ can be an assembly joint, $M_{\alpha_2,1}$ notes the subcomponents of $\mathscr{D}$ that are mated, and $M_{\alpha_2,2}$ the relationships and transformations among these components.

Lastly, L is a mapping $L : A \rightarrow 2^P$ that relates the attributes to subsets of the geometric and topological elements in the boundary model, P, of the design. Given a design attribute α_1, $L(\alpha_1)$ returns the collection of boundary model entities associated with the design attribute α_1. A design D is a primitive design if the set $\mathscr{D}$ of subcomponents is empty.

3.2. Signature structures

A mechatronic design and its design attributes can be represented as a graphical structure we call a *design signature*. The *design signature* for a design D, S_D, is a hypergraph $H(V,E)$ with labeled edges, where V is the set of vertices $V = \{v_1, v_2, \cdots, v_n\}$, and E is the set of edges $E = \{e_1, e_2, \cdots, e_m\}$, $e_i = (v_j, v_k)$ and $e_i \in E$ if $v_j \in A$ and $v_k \in L(\alpha_1)$. This implies that, in the design signature, all vertices representing design attributes are connected to the vertices representing the entities in the boundary model that attributes refer to. Two design signatures, D and D', are equivalent ($D = D'$) if their hypergraphs are isomorphic. We shall explore a specific example of equivalence and similarity of design signatures in Section 4.3.

Fig. 3 presents a simple example for illustration purposes: the design of a bracket. Fig. 3 illustrates a design signature (highly simplified) for a very simple example of a machined bracket [28]. Fig. 3(a) shows the solid model of the bracket and the topological relationships in its boundary model. Fig. 3(b)–(d) shows the design attributes for this model: design features, tolerances, and machining features. Fig. 3(e) shows the hypergraph of the basic design signature, S_D, for the bracket.

3.3. Knowledge storage

Generation of design signature, operators and their storage in the knowledge-base is accomplished via feature recognition from the CAD models [29,30]. In practice, information about tolerances, design features and other knowledge is already associated with the design data in some fashion. For these cases, feature extraction is a relatively straightforward translation of the attribute data for the CAD model into attribute data in the design signature graphs. This can be implemented using the native functions for the particular CAD system the data is stored, or using standard data exchange formats (such as STEP). For example, if the data was created in SDRC I-DEAS, one could create an I-DEAS plug-in (using their CORBA-based Open I-DEAS development API) to export the needed data and relationships.

In cases where the data is not associated with the CAD model a priori, feature recognition can reconstruct the needed indexing information. Working feature identification systems have been demonstrated for finding manufacturing, design, and some types of assembly features [29,30].

3.4. Knowledge retrieval

Knowledge retrieval from the Repository is accomplished using graph matching and approximation algorithms to compare design signatures. The basic process is to *hash* the signature structures and insert them (along with the CAD data and attributes) into the knowledge-base. The

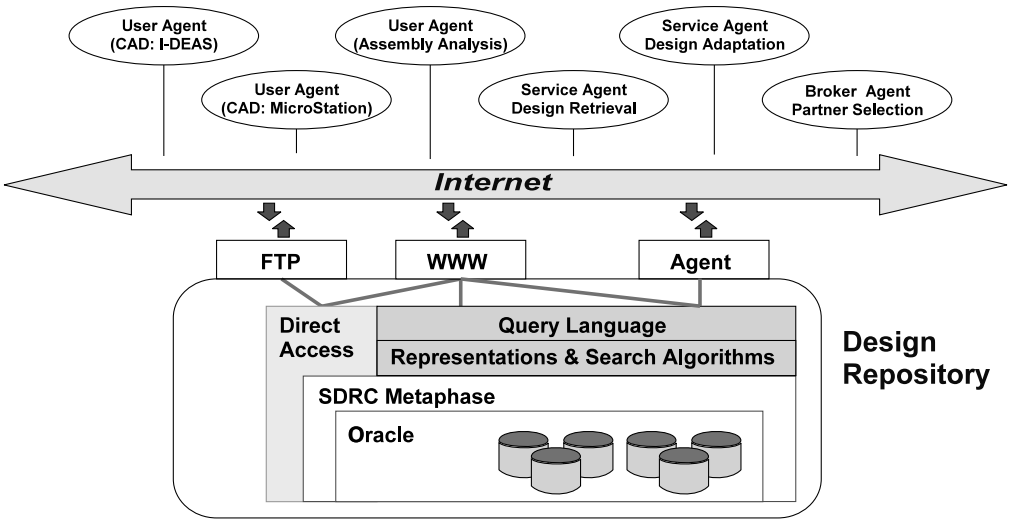


Fig. 4. System architecture.

operators, such as those described in Fig. 3, are used to form an index into the knowledge-base.

While the core of the retrieval process is NP-hard (testing graph isomorphism to determine if pairs of signatures are equal), based on our experiments described in Refs. [31,32] and in Section 4.3, we know that making extensive use of engineering domain knowledge and domain-specific heuristics significantly improves the performance, both reducing the number of isomorphism-based equivalence checks and directing the search toward application of more promising operators.

In addition, these experiments have indicated that the task of computing isomorphism among signature graphs is largely a matter of symbolic comparisons on integer data; and proves to be considerably cheaper than the extensively floating-point geometric computations required for reasoning directly with complex CAD data models.

4. Current research results

Engineering Knowledge Repository project began in 1998. In the following sections, we present the current research results in three areas. System Architecture, Conceptual Design and Query Interfaces, and Structure Matchers for Retrieval of individual CAD models.

4.1. Repository system architecture

The Repository is accessible in two ways: (1) through standard file transfer protocol (FTP); and (2) as a web-based service, through hypertext transfer protocol (HTTP). Fig. 4 shows the current system architecture for the Knowledge Repository. The site runs on a multi-processor Sun Microsystems UltraSPARC workstation that can accommodate 300 gigabytes of storage running the Apache Web Server. Over the course of this project, we plan to migrate this platform to include the SDRC/Metaphase Product Data Management (PDM) system.[4] By employing commercial PDM and database tools (our Metaphase site runs on the Oracle Version 8 database), we hope to build a commercial-strength, scalable infrastructure for the Repository. We also believe that the use of industrial-strength systems will free us to perform research on a more complex and realistic scale.

4.2. Conceptual query interface

To navigate intricate product data management (PDM), part and assembly data, and case-based design knowledge-bases, we require an interface that provides designers with the ability to conceptually describe engineering artifacts. The conceptual specification can then be employed in the search for detailed design and manufacturing information in large repositories of previously archived designs.

We have created a Java-based environment for Conceptual Understanding and Prototyping (CUP) to be used as a query interface to the Design Knowledge Repository (http://repos.mcs.drexel.edu). CUP enables users to interrogate large quantities of legacy data (models and assemblies) and identify artifacts with structural and functional similarities—allowing designers to better perform case-based and variant design [33,34].

[4] PDM system is a special database layer specifically tailored to the collecting and sharing of engineering data (of any form) throughout the engineering enterprise. This includes all information created to describe, configure and build a product—including relationships between data and the product structure. PDM systems enable storing and tracking of data throughout the product life cycle. This includes data such as: workflow and document routing; review and approval processes; change/version control; work orders and instructions; engineering bills of materials; configuration management; support for the maintenance of different engineering views, design alternates and substitutes.

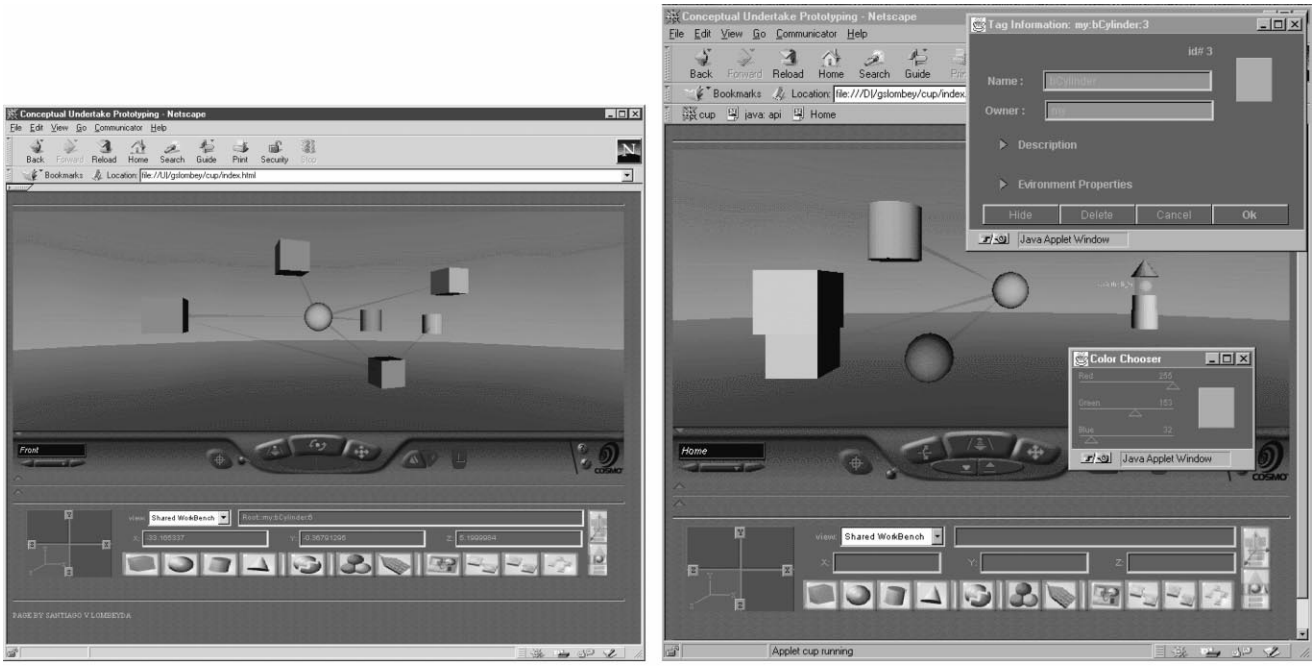


Fig. 5. Screen shots of CUP and some of the facilities for linking and tagging objects.

4.2.1. Approach

During the Conceptual Design phase of product development, teams of designers may begin to develop a new product by sketching its general shape on paper. This "back of the envelope" approach is a key aspect of the creative process—once completed, one has a clearer idea of what one is creating and can proceed to drafting or CAD activity.

Our approach is to provide designers with a conceptual design environment consisting of a very elemental CAD functionality. In this environment, the user can design without focus on details and yet be able to introduce enough information so that a full design may evolve from this work. Further, the CUP environment provides for creation and manipulation of three-dimensional primitives (blocks, cylinders, etc.); for the description and annotation of structural, functional, and behavioral properties of objects; and for the definition of relationships between these objects. Some of this functionality is shown in Fig. 5: primitives can be inserted into a scene, links represent relationships among primitives, and prototype objects grouped together.

4.2.2. Example

Consider this example scenario of how CUP might be used: a design team, faced with the task of creating a new missile seeker, might want to interrogate the CAD knowledge-base and examine previous design cases that might be relevant to this new problem. Examining this legacy data can prove time consuming and tedious, unless one knows exactly what one is looking for. CUP can be used to sketch a simplified seeker assembly, which can then be the target for a query to search for similarities among the many dozens stored in a corporate design data/knowledge-base.

As illustrated in Fig. 6(a) and (b), CUP allows a designer to quickly sketch out, in 3D, the major components and structural relationships in the assembly. Rather than performing detailed CAD to create a draft design (detailed CAD modeling for even this simple model took several days), designers can, in a matter of minutes, build a conceptual design. This conceptual design can then be used as a starting point for further refinement or as a query to the design knowledge-base. CUP also, via the attributing, tagging and labeling features, helps designers to capture the design intent and to build a structure–function–behavior (S–B–F) model of the artifact.

4.3. Structure matching for model retrieval

The conceptual model created by CUP can be represented as an attributed directed hypergraph. Executing a query to the Repository is a matter of searching the Knowledge Repository for other models and assemblies with similar graph-based structures. In this effort, we have created a specific instance of the general problem described in Section 3 for indexing and retrieval of individual solid models based on their design features [31].

4.3.1. Approach

For individual solid models, we create a signature structure called a Model Dependency Graph (MDG). This graph is a directed acyclic graph, which has some unique

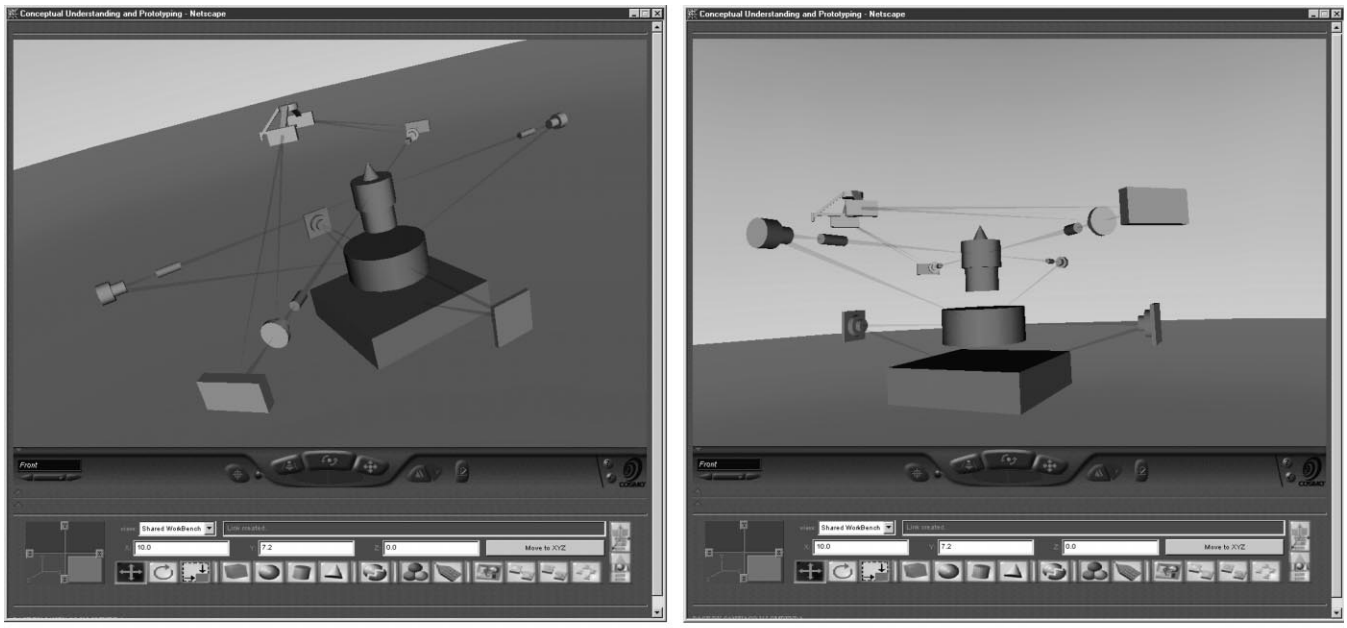


Fig. 6. Illustrations of a conceptual design for the missile seeker pictured in Fig. 2.

characteristics. The model history,[5] MH, is defined as $\mathrm{MH} = \{m_0, \cdots, m_n\}$. The m_i is the complete model at stage i of the design. That is, m_i represents the solid model after feature f_i is applied to the model. There is an ordering inherent in the design history graph. In the case where it is not clear which operation or feature came before the other we impose a left-to-right ordering on the operations. The m_i may be generated and stored at design time. Or they may be easily generated from the design history. Let $vol(f_i)$ represent the "solid" volume that is either added or removed from the complete model by the application of feature f_i.

Definition 1 (*Model Dependency Graph—basic definition*). A Model Dependency Graph (MDG) is defined as $G = (V, E)$. The vertex set is defined as $V = \{f_0, \cdots, f_n\}$. The indices on the f_i represent the order that the features were applied during the design process. The edge set can be defined as $E = \{(f_i, f_j)$ such that $i > j$, $vol(f_i) \cap vol(f_j) \neq \emptyset\}$. Note that $\cap$ is not a regularized intersection. This implies that two vertices share an edge if the features they represent touch each other in some way.

One limitation with the MDG as it has been defined in Definition 1 is that it assumes an explicit ordering on the features or design operations. In many cases this may be captured in the solid modeling application in the form of a design history. But can the MDG be used when dealing with CSG trees? The answer is yes and can be obtained by extending the definition of the MDG to work recursively down the CSG tree.

Definition 2 (*Model Dependency Graph—non-linear definition*). Let $T = (op\ left\ right)$ be a CSG tree or some non-linear design history where *op* is an operation and *left* and *right* are CSG subtrees or primitives shapes. Let $G_1 = (V_1, E_1)$ be the MDG of *left* that results from either the basic definition or the non-linear definition. Let $G_2 = (V_2, E_2)$ be the MDG of *right* that results from either the basic definition or the non-linear definition. Then the MDG of T can be defined as $G = (V, E)$ such that $V = V_1 \cup V_2$ and $E = E_1 \cup E_2 \cup E_3$ where $E_3 = \{(v_2, v_1), v_1 \in V_1, v_2 \in V_2$ s.t. $vol(v_1) \cap vol(v_2) \neq \emptyset\}$. Note that $\cap$ is not a regularized intersection.

An example of a solid model with different possible design feature histories is shown in Fig. 7. There is a property of the MDG that I will exploit in our similarity assessment of solid models: *digraph D-morphism*. For a given pair of graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ a *D-morphism* is formally defined in Ref. [35] as a function $f : V_1 \rightarrow V_2$ such that for all $(u, v) \in E_1$ either $(f(u), f(v)) \in E_2$ or $(f(v), f(u)) \in E_2$ and such that for all $u \in V_1$ and $v' \in V_2$ if $(f(u), v') \in E_2$ then there exists a $v \in f^{-1}(v')$ for which $(u, v) \in E_1$.

Theorem 1 (*D-morphisms of Model Dependency Graphs.*). *Let G_1 and G_2 be two MDGs for the same solid model resulting from different orderings of a feature*

[5] This concept is similar to that of a "design history" that has been much addressed in the engineering design community.

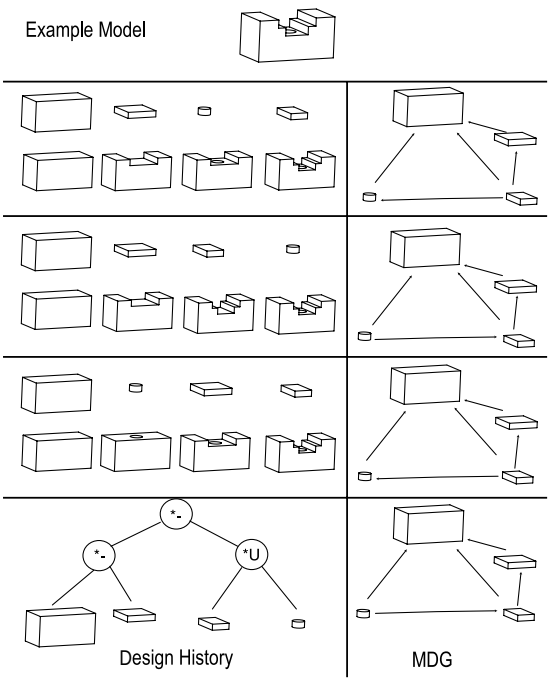


Fig. 7. Pictured is a single solid model and several alternative design feature histories and one possible CSG tree that can produce it. On the right are the MDGs for each of these alternatives—note that they are all D-morphic to one another.

set $F = \{f_0, \cdots, f_n\}$ (such as shown in Fig. 7). G_1 and G_2 are D-morphic.

Proof. Pick any two orderings of the set $F = \{f_0, \cdots, f_n\}$ arbitrarily. Let these orderings be $L = \{l_0, \cdots, l_n\}$ and $H = \{h_0, \cdots, h_n\}$ where $\forall f_i \in F$, $\exists l_j \in L, h_k \in H$ such that $f_i = l_j = h_k$ and $\exists i, 0 <= i <= n$ such that $l_i \neq h_i$. Let $G_1 = (V_1, E_1)$ be the MDG that results from L and let $G_2 = (V_2, E_2)$ be the MDG that results from H. It is clear that $V_1 = V_2$. By the definition of the MDG, these vertex sets must be equal to the set F. Now take any two vertices $v_k, v_l \in V_1$. Pick out the vertices $v_m, v_p \in V_2$ such that $v_k = v_m = f_i$ and $v_l = v_p = f_j$. Note that $vol(v_k) = vol(v_m)$ and $vol(v_l) = vol(v_p)$. Therefore, $vol(v_k) \cap vol(v_l) = vol(v_m) \cap vol(v_p)$. Hence, from the definition of the MDG, if there is an edge $(v_k, v_l) \in E_1$ where $k > l$ then either $(v_m, v_p) \in E_2$ where $m > p$ or $(v_p, v_m) \in E_2$ where $p > m$. Therefore, G_1 and G_2 are D-morphic. □

We compare the similarity of two solid models by testing for a D-morphism or for a subgraph D-morphism. The general problem of determining if there exists a D-morphism for a given pair of directed graphs is NP-complete [35]. However, there are two aspects of this problem domain that we can exploit to significantly reduce this complexity:

- First, it is not necessary to completely solve the D-morphism problem: Since we are only concerned with similarity, knowing if two MDGs are "almost" D-morphic is sufficient. Hence, we can use a heuristic method for the D-morphism test. Specifically, we will develop an algorithm that is a variant of gradient descent (or hill-climbing) that exploits the feature information we have in the design feature history.
- Second, there is a great deal of domain knowledge present in the CAD model and in the feature history that can reduce the search space. For example, we will only consider mappings that compare similar feature types (i.e. holes map to holes, not to pockets). Additional constraints about vertex degree and size, location, and orientation can also be considered.

In testing for a D-morphism, initially we arbitrarily choose an initial set of pairings between the nodes of the two graphs (i.e. for each node of G_1 we choose at random a node of G_2 such that no two nodes of G_1 are "paired" with the same node of G_2). We then swap the pairings of the two nodes that reduce the value of our evaluation function by the greatest amount. If there is no swap that reduces the value of the evaluation function but if there are swaps that result in the same value (i.e. we have reached a plateau), we choose one of these at random. The algorithm ends when either every possible swap increases the value of the evaluation function or it makes P random moves on the plateau. We are currently experimenting with constant values for P.

We use as our evaluation function the count of the number of "mis-matched" edges. That is, the evaluation function, $H = |E|$ such that $G_1 = (V_1, E_1)$ is the smaller of the two graphs being compared, $G_2 = (V_2, E_2)$ is the larger of the two graphs, and $E = \{(u, v) \in E_1$ such that $((paired(u), paired(v)) \notin E_2 \wedge (paired(v), paired(u)) \notin E_2) \vee label(u) \neq label(paired(u)) \vee label(v) \neq label(paired(v))\}$. As a measure of similarity we employ the value $H^* = (\min\{H_1, \cdots, H_n\})/|E_1|$ where $H_1, \cdots, H_n$ are the values of H from up to n random restarts of the algorithm and E_1 is the edge set of the smaller graph. The function "paired(x)" above returns the node $y \in V_2$ that is currently paired with the node $x \in V_1$. The function "label(x)" used above returns the label of the node x.

4.3.2. Experimental results

We generated a family of 10,000 models using the ACIS 3D Toolkit running on a 450 MHz Pentium II running Microsoft Windows NT 4.0. These models were pseudo-random variations on the US Department of Energy's Technologies Enabling Agile Manufacturing (TEAM) Project test parts pictured in Fig. 8. These parts have a variety of standard feature types, such as pockets, slots, holes, counterbore holes and bosses; in addition, many of the features interact and intersect, leading to a variety of different possible orderings for design feature histories and manufacturing process plans. The two parts pictured have several

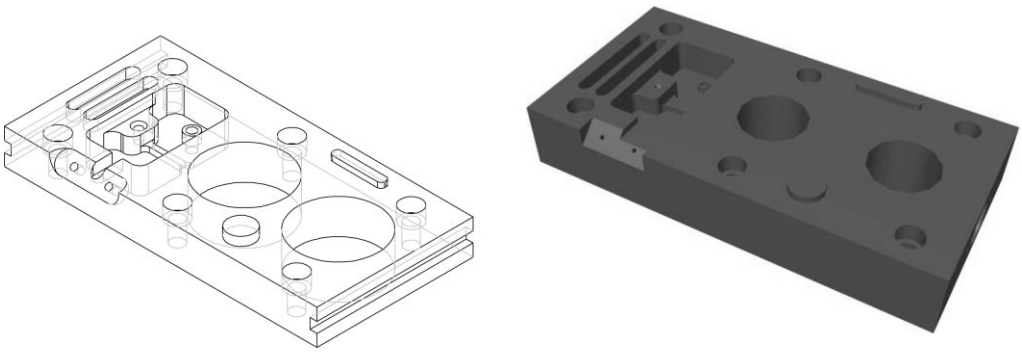


Fig. 8. Two of the test parts from the DOE TEAM Project. Both of these parts are available from the Repository at http://repos.mcs.drexel.edu.

subtle differences that make them a useful target domain for experimentation.

Our random "TEAM part" generator is based on the work of Alexei Elinson at the University of Maryland at College Park [32]. It operates by varying the number of features, the location features, and the number of different feature types over the part (depressions and protrusions, pockets, holes). For each of the 10,000 models generated, the design feature history of each model was stored. Using this feature information, the MDG for each model was generated.

Next, we selected two arbitrary Query Models from the set of 10,000 random models, shown in Fig. 9. The figure shows the design histories of these parts; the MDG signature graphs are shown in Fig. 10(a) and (b). Each of the query parts was compared to each part from the set of randomly generated parts. To perform MDG comparison, a random restart gradient descent algorithm was used (as described in Section 3) with number of restarts fixed at 1000. These matching tests searched for a subgraph of the larger of the query MDG and the given MDG from the set of 10,000 that was D-morphic to the smaller. The matching algorithms are implemented in C++ using the LEDA graph library. And the tests were performed on a 300 MHz Sun UltraSPARC 30 workstation running Sun Solaris 2.6.

Fig. 11 shows the results of these two queries. The histograms show that each query model partitioned the set of 10,000 random parts into distinct subsets, based on the result of the D-morphism test. For both query parts, there was a high percentage of parts found to be "similar." This is to be expected, since the set of parts consist of a family of parts generated at random from a limited set of operations based on the TEAM parts. For both queries, the query models each were among the set of models D-morphic to the query.

Results for Query Model 1. For *Query Model 1*, 3128 models were found such that their SMDGs were subgraph isomorphic to that of the query model or that the SMDG of the query model was subgraph isomorphic to it. Among this set was the query model itself. Also among this set was model (a) in Fig. 11. If you look at this model you will see that, like *Query Model 1*, it consists of two pockets each cutting through two faces, one with two holes and the other with three holes. Also common to both *Query Model 1* and (a) is a slot adjacent to one of the pockets. These two parts are very much alike. In fact, in this case, the parts were not only subgraph isomorphic, but were actually isomorphic.

Next, notice model (b). This model was among 2406 models where the ratio of "mis-matched" edges to total edges at the completion of the matching test was greater than 0 but less than or equal to 0.125. The actual value of this particular case was 0.07. Aside from the interaction between one of the pockets and the slot in *Query Model 1*,

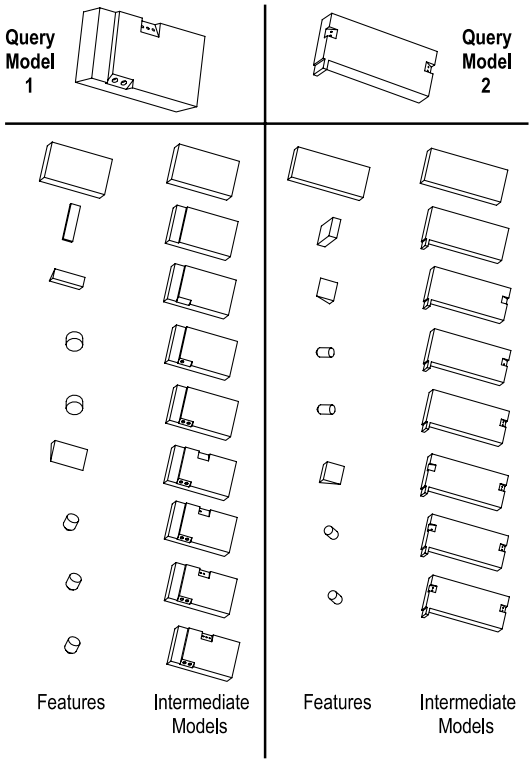


Fig. 9. Two randomly generated query models with their design feature histories.

the UMDG for the query model would be isomorphic to that of model (b).

Models (c)–(f) were in the next four groups shown on the histogram for query 1, respectively. Model (c) has an additive feature on one of its side faces while the query model had no such feature. Model (d) has five pockets and holes in each and lacks the slot that the *Query Model 1* has. Model (e) has two additive features on two of its side faces while the *Query Model 1* has no such additive features. None of the edges of the UMDG of model (f) matched any of that of the query model. This part has one additive feature at one end and no other features. The query model does not have an additive feature like this one.

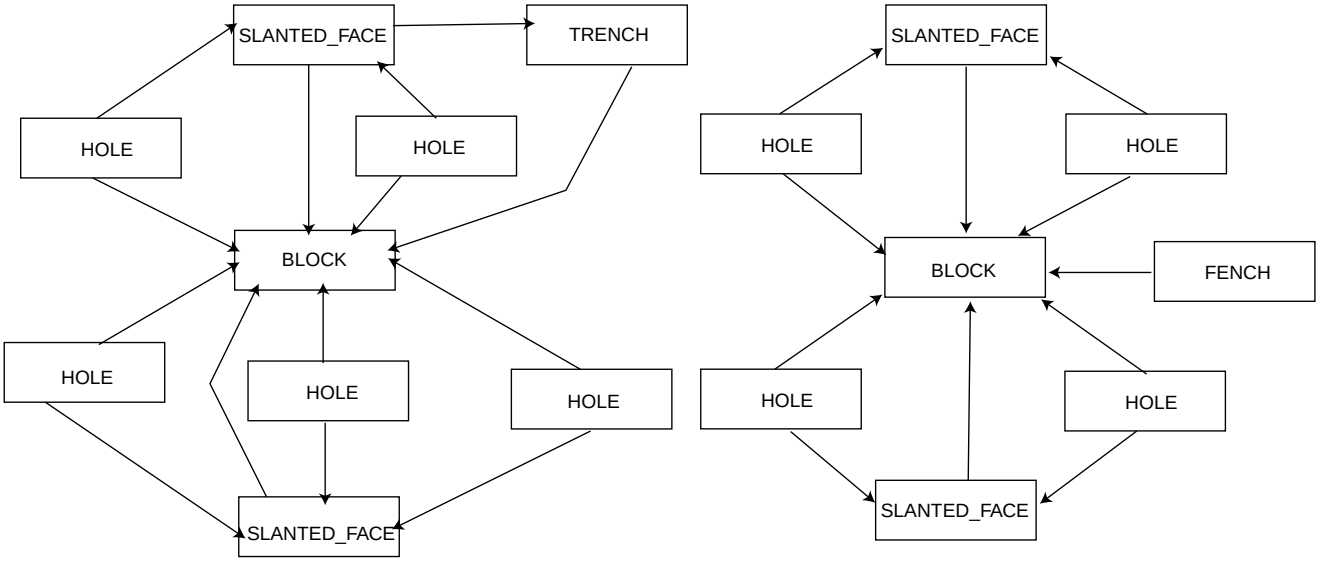
Results for Query Model 2. For *Query Model 2*, 2440 models were found such that they were subgraph isomorphic to the UMDG of the query model or that the UMDG of the query model was subgraph isomorphic to it. Among this set was the query model itself. Also among this set was model (g) in Fig. 11. If you examine these two models, you will see that each has an additive feature on one side face and each has two pockets each cutting through two faces with holes in each. They are very much alike.

Model (h) is one of 3923 models with a ratio of mismatched edges to total edges greater than 0 and less than or equal to 0.125. This ratio for model (h) was actually 0.09. The difference between these two models is that (h) has a slot while *Query Model 2* has an additive feature on of its side faces.

Models (i)–(l) are in the next four groups on the histogram. Model (i) has two slots not in the query model and the query model has the additive feature on one of the side faces. Model (j) is the same model as (d). This model was about the same in dissimilarity to both query models. The UMDG of model (k) is a 50% match to that of the query. This model has a pocket cutting through two faces with one hole through the pocket. Similarly the query model has a pocket like this. Model (k) also has two slots, but the query model does not. Every edge in the UMDG for model (l) was mis-matched when compared to that of *Query Model 2*. Model (l) has a slot in two of its side faces and no other features. The query model has no slots.

Observations. All 10,000 models used in this experiment along with their design history and associated MDGs are available as ACIS.sat files at http://repos.mcs.drexel.edu/CICIRELLO-THESISDATA.

To compare *Query Model 1* against *all* 10,000 models took a total of 23 h, 17 min, and 23 s of CPU time on the Sun UltraSPARC 30 (an average of 8.38 s per comparison). The fastest comparison took less than 0.01 s. The slowest comparison took 183.35 s. There were a few cases where the random initial starting point represented an isomorphism, but this was a rare occurrence. On average, the algorithm made 3153 swaps of node mappings with a high of 7699 node mapping swaps. To compare *Query Model 2* against *all* 10,000 models took a total of 14 h, 8 min, and 33 s of CPU time on the Sun UltraSPARC 30 (an average of 5.09 s per comparison). The fastest comparison took less than 0.01 s. The slowest comparison took 104.37 s. There again were a few cases where the random initial starting point represented an isomorphism, but again this was a rare occurrence. On average, the algorithm made 3026 swaps of node mappings with a high of 6493 node mapping swaps. In these experiments, only feature information was used and the parts contained approximately the same number and type of features (about 10–25 features). We expect that search and matching times can be vastly improved by employing additional engineering attribute information (such as tolerances and dimensions). This type of information was not available for the models generate for these tests.



(a) Query Model 1 (b) Query Model 2

Fig. 10. The MDGs for the randomly generated Query Models.

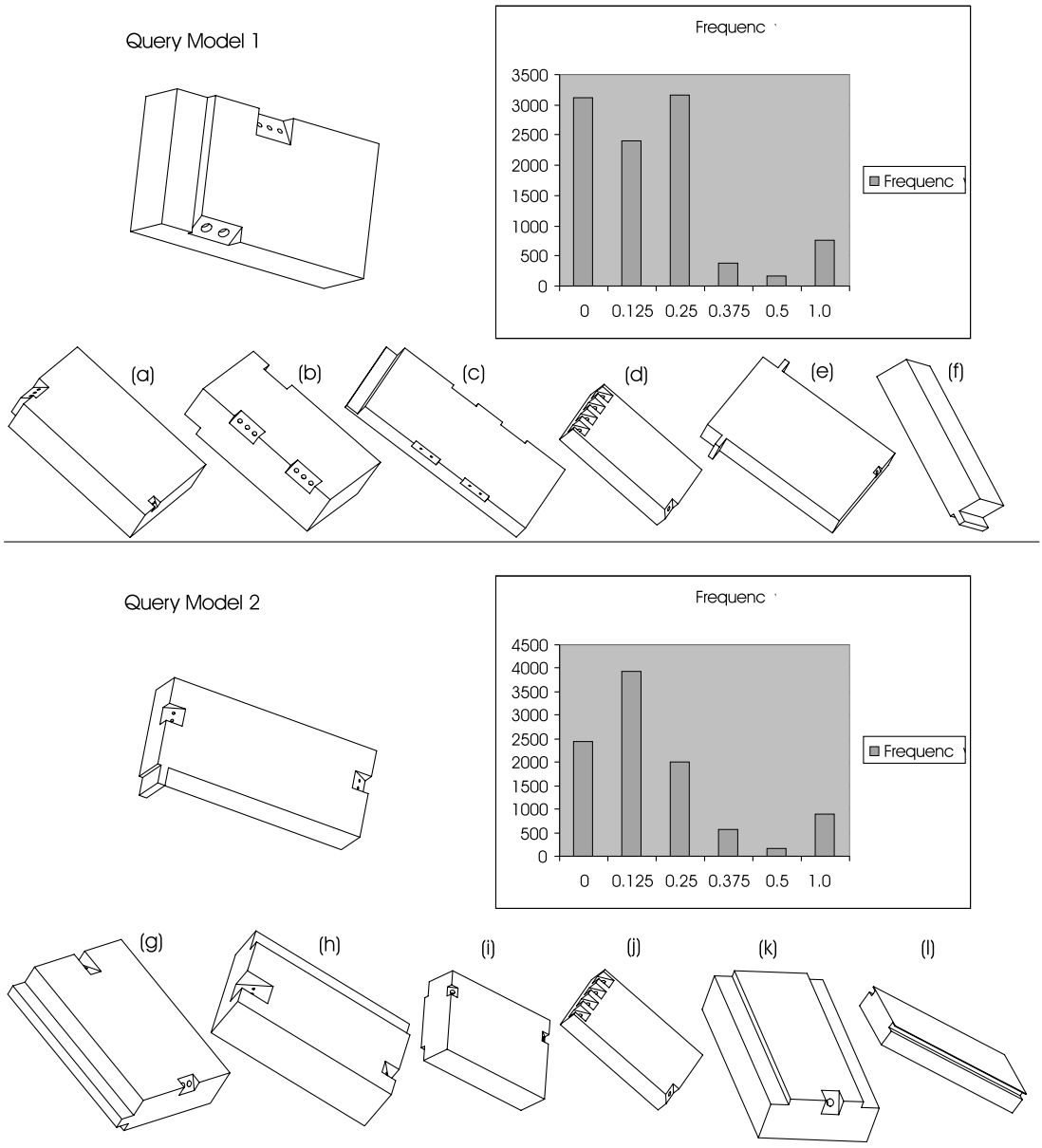


Fig. 11. Example output data from examining D-morphism over the database of 10,000 solid models for the two query models in Fig. 9. The histogram shows the number of models (from the 10,000 in the database) that fall into distance categories based on the D-morphism test. Read from left-to-right, the returned models are in order of decreasing similarity to the query model.

We chose to use design features in these experiments primarily because we wished to perform a proof-of-concept, which involved having thousands of artifacts. Other feature information, such as machining features produced by a feature recognizer, could be used for future experiments.

5. Discussion

5.1. Anticipated impact

Engineering Design Knowledge Repository is intended to be a knowledge archive for engineers, researchers, and students. Hence, it is worth noting several user scenarios and describing the types of interactions with our store of engineering knowledge that the Repository will have to support. In particular, recalling Fig. 1, we envision the following user community:

1. *Students:* The Repository will eventually be a living textbook of examples and case studies that can be used for reference, training, and as a source for benchmark challenges. This activity will involve the participation of graduate and undergraduate students at many universities.
2. *Researchers and Developers:* The R&D community can

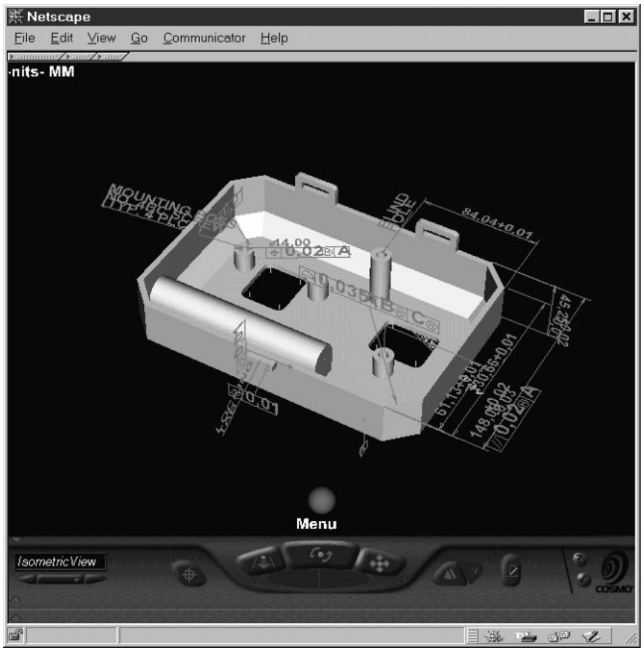


Fig. 12. VRML versions of the CAD models will be served up from the Repository, complete with engineering attributes and links. This model has been provided courtesy of SDRC.

now access case studies and new examples; benchmarks and standards can be defined; plus use the Repository as a vehicle for enhanced collaboration. This will allow researchers to focus on new research areas instead of on data acquisition. The CAD dataset has already been used extensively by those in the standards, data translation, and CAD visualization communities.

3. *Designers and Practicing Engineers:* Professionals will have access to online check lists for design-for-manufacture rules, manufacturing constraints, process capabilities, and example designs. This will enable them to perform better design-for-manufacture, access rapid prototyping services and browse a living online textbook of design experiences.

To deliver a prototype platform for this user community, we are developing data structures and methods for the creation and manipulation of design signatures. Of particular practical importance will be determining what the underlying ontology for the signature structures must include to enable the fast computation of distance measures, search heuristics, equivalence tests and operators. Initially, we are implementing tools to satisfy the following requirements:

1. *Navigation System:* We are developing an ontology for categorizing and classifying the design models in the Repository. Currently, our dominant data types and users are from mechanical engineering. Eventually, the Repository will encompass all means of CAD data (for example, Architecture/Engineering/Construction models). The initial navigation system will enable users to proceed hierarchically through the stored data and search for specific models of interest.
2. *Query Interface for Conceptual Designs:* The most significant impact of the Repository on design practice occurs during the conceptual phase of design, when designers do not have detailed CAD models—rather they are dealing with abstractions of the new design. These abstractions might include back-of-the-napkin-style functional component diagrams, performance specifications, and behavioral and geometric constraints.
3. *Collaboration Knowledge Networks:* As teams of designers work on a project together, the design evolves as communications flow among the design team members. We are working to integrate the CAD data with Computer Supported Collaborative Work environments in order to capture communications media (voice, email, instant messages) and associate the communications about design with the design data itself.
4. *Contributor Tools:* The last of our three near-term projects is to develop and evaluate tools for contributors to add designs and related information into the design knowledge-base.

5.2. Future research directions

Engineering Knowledge Repository is a multi-year effort that began during 1998–1999. This experimental research platform will enable us (and other researchers) to address the following long-term research objectives:

1. *Linking design signatures with functions and methods from other engineering data types:* These structures will capture traditional engineering data types, such as assembly relationships, tolerances, and features.We hope to extend these ideas to develop queryable graph-based relations that link geometric dimension and tolerance models, CBR-based structure–behavior–function models of design, and manufacturing plans with geometric representations, such as shown in Fig. 12. In the figure, the annotations are all active links pointing to other data about this design that is resident in the Repository and elsewhere. We are also working to include structure–behavior–function information about the design data [36–38].
2. *Automated generation of design signatures and geometry-intensive knowledge-bases:* We are implementing algorithms that generate design signature graphs given a design's solid model and engineering attributes. Our approach is to recognize engineering and manufacturing features from the design data. Existing systems for case-based design and knowledge-based engineering often require a large amount of human input to create the cases-bases of design knowledge with which to reason. This work will contribute to automating this process.

3. *Techniques to manage geometric complexity:* Computational operations on geometric information (such as CAD and solid models) are floating-point and memory intensive, placing unique burdens on software and hardware. Large knowledge-bases contain millions of assertions and facts that can occupy gigabytes of memory—but this data is primarily symbolic in nature. Individual CAD models can occupy megabytes of memory and single assemblies can occupy gigabytes. We are working on techniques to effectively manage large amounts of complex geometric and engineering data.
4. *Adaptable Search Interfaces:* Existing commercial systems for Product Data Management (PDM) and engineering databases support data management very effectively in closed enterprises where all users, user needs, and datatypes are defined (and delimited) a priori. Unfortunately, this is not a satisfying general situation; rarely do the pre-defined views provide all the perspectives that are needed and rarely can all datatypes be taken into account. We plan to develop an API through which agents can customize access to the knowledge archived in the knowledge-base.

6. Conclusions

This paper has presented our initial attempt to formalize the problem of managing knowledge-bases of highly geometric CAD data and related engineering metadata. We see this work filling an important need for digital library support for engineering design and manufacturing applications.

It is our observation that much of the current generation of digital library and database technology focuses primarily on pictorial and multimedia information: 2D digital images, movies, and geographic systems. Many existing techniques are not directly applicable to digital libraries of 3D solid models and engineering information. Existing work has not yet exploited the availability of 3D solid models or included important engineering information, often attached to the solid model, such as tolerances, design/manufacturing features and inter-part relationships in assemblies. Previous work has addressed only the gross shapes of single designs; none of the existing approaches is directly applicable to electro-mechanical (mechatronic) assemblies, where inter-part relationships and models of function are more significant.

It is our hope that our research expands the understanding of this new problem domain and lays the foundation for exploring new techniques to enhance our ability to search and retrieve 3D solid model data. Further, we believe that existing approaches to multimedia libraries can be augmented with geometric reasoning techniques that are tightly coupled with engineering knowledge and solid models—such as those developed in the future as part of this research.

Acknowledgements

This work was supported in part by National Science Foundation (NSF) Knowledge and Distributed Intelligence (KDI) Initiative Grant CISE/IIS-9873005, as well as by CAREER Award CISE/IIS-9733545. Additional support was provided by The National Institute of Standards and Technology (NIST) Grant 60NANB7D0092 and AT&T Labs, Internet Platforms Technology Office. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation or the other supporting government and corporate organizations.

References

[1] Ullman DG. The mechanical design process. New York: McGraw-Hill, 1992.
[2] Regli WC. Digital library support for engineering design and manufacturing, ASME Design Technical Conferences, 19th Computers and Information in Engineering Conference, New York, USA, 12–16 September, Las Vegas, NV. Las Vegas, NV: ASME Press, 1999 Special Track on Internet-Aided Design, Manufacturing and Electronic Commerce.
[3] Szykman S, Bochenek C, Racz JW, Senfaute J, Sriram RD. Design repositories: next-generation engineering design databases. IEEE Intelligent Systems 2000; in press.
[4] Snead CS. Group technology: foundations for competitive manufacturing. New York: Van Nostrand Reinhold, 1989.
[5] Picard RW, Pentland AP. Introduction to the special section on digital libraries: representation and retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence 1995;18(8):769.
[6] The University of California at Berkeley Digital Library Project. http://elib.cs.berkeley.edu.
[7] Flickner M, et al. Query by image and video content—the QBIC system. IEEE Computer 1995;28(9):23–32.
[8] Gupta A, Jain R. Visual information retrieval. Communications of the ACM 1997;40(5):71–9.
[9] Bach JR, Fuller C, Gupta A, Hampapur A, Horowitz B, Humphrey R, Jain R, Shu C. An open framework for image management, Storage and Retrieval for Still Image and Video Databases IV, February. SPIE, 1996.
[10] James de St. Germain H, Stark SR, Thompson WB, Henderson TC. Constraint optimization and feature-based model construction for reverse engineering. Proceedings of the ARPA Image Understanding Workshop, February 1996.
[11] Lamdan Y, Wolfson HJ. Geometric hashing: a general and efficient model-based recognition scheme. Second International Conference on Computer Vision 1988:238–49.
[12] Thompson WB, Own JC, James de St. Germain H. Feature-based reverse engineering of mechanical parts. Technical Report UUCS-95-010, The University of Utah, Department of Computer Science, July 1995.
[13] Thompson WB, Riesenfield RF, Owen JC. Determining the similarity of geometric models. Proceedings of the ARPA Image Understanding Workshop, February 1996.
[14] Wolfson HJ. On curve matching. IEEE Transactions on Pattern Analysis and Machine Intelligence 1990;12(5):483–9.
[15] Cybenko G, Bhasin A, Cohen KD. Pattern recognition of 3D cad objects: towards an electronic yellow pages of mechanical parts. Technical report, Dartmouth College, Thayer School of Engineering, Hanover, NH, January 1996.

[16] Kim J, Ringo Ling S, Will P. Ontology engineering for active catalog. Technical report, The University of Southern California, Information Sciences Institute, 1997.
[17] Eastman CM, Band AH, Chase SC. A formal approach for product model information. Research in Engineering Design 1991;2(2).
[18] Eastman CM, Jeng T. A database supporting evolutionary product model development for design. Automation and Construction 1997.
[19] Eastman CM, Parker DS, Jeng T. Managing the integrity of design data generated by multiple applications: the principle of patching. Research in Engineering Design 1997.
[20] Eastman CM. Managing integrity in design information flows. International Journal of Computer Aided Design 1996;28(6/7):551–65.
[21] The Alexandria Project. University of California at Santa Barbara Digital Library Project. http://alexandria.sdc.ucsb.edu.
[22] Heine L-R, Harrow P. Industrial automation systems and integration—parts library—part 31: geometric programming interface. Technical Report ISO DIS 13584-31, International Organization for Standardization, 23 February 1996. ISO/TC 184/SC4/WG2.
[23] Pierra G, Wiedmer HU. Industrial automation systems and integration—parts library—Part 42: methodology for structuring part families. Technical Report ISO DIS 13584-42, International Organization for Standardization, 30 May 1996. ISO/TC 184/SC4/WG2 N243.
[24] Kemper A, Wallrath M. An analysis of geometric modeling in database systems. ACM Computing Surveys 1987;19(1):1–45.
[25] Hardwick M, Spooner DL, Rando T, Morris KC. Sharing manufacturing information in virtual enterprises [Special issue on Computer Science in Manufacturing. Wozny M, Regli W (editors)]. Communications of the ACM 1996;39(2):46–54.
[26] Regli WC, Gaines DM. An overview of the NIST repository for design, process planning, and assembly. International Journal of Computer Aided Design 1997;29(12):895–905.
[27] Tuttle R, Little G, Regli W, Corney J, Clark DER. Common libraries for networked engineering applications. In: EUROPIA97: The Proceedings of the Sixth International Conference on Applications of Computer Networking in Architecture, Construction, Design, Civil Engineering, and Urban Planning, Edinburgh, Scotland, 2–3 April 1997. ISBN 2-909285-07-3. p. 1–11.
[28] Gupta SK, Nau DS. A systematic approach for analyzing the manufacturability of machined parts. Computer Aided Design 1995;27(5):323–42.
[29] Regli WC, Gupta SK, Nau DS. Extracting alternative machining features: an algorithm approach. Research in Engineering Design 1995;7(3):173–92.
[30] Regli WC, Gupta SK, Nau DS. Toward multiprocessor feature recognition. Computer Aided Design 1997;29(1):37–51.
[31] Cicirello V, Regli WC. Resolving non-uniqueness in design feature histories. In: Anderson D, Bronsvoort W, editors. Fifth Symposium on Solid Modeling and Applications, New York, 8–11 June, Ann Arbor, MI: ACM Press, 1999.
[32] Elinson A, Nau DS, Regli WC. Feature-based similarity assessment of solid models. In: Hoffman C, Bronsvoort W, editors. Fourth Symposium on Solid Modeling and Applications, New York, 14–16 May, Atlanta, GA: ACM Press, 1997 p. 297–310.
[33] Lombeyda S, Regli WC. Conceptual design for assembly. In: Gupta SK, editor. ASME Design Technical Conferences, Fourth Design for Manufacturing Conference, New York, USA, 12–16 September , Las Vegas, NV: ASME Press, 1999.
[34] Lombeyda S, Regli WC. Conceptual design for mechatronic assemblies. In: Anderson D, Bronsvoort W, editors. Fifth Symposium on Solid Modeling and Applications, New York, USA, 8–11 June, Ann Arbor, MI: ACM Press, 1999.
[35] Garey MR, Johnson DS. Computers and intractability: a guide to the theory of NP-completeness. New York: Freeman, 1979.
[36] Schlenoff C, Denno P, Ivester R, Libes D, Szykman S. An analysis of existing ontological systems for applications in manufacturing, ASME Design Technical Conferences, 19th Computers and Information in Engineering Conferences, New York, USA, 12–16 September, Las Vegas, NV, DETC99/CIE-9024. Las Vegas, NV: ASME Press, 1999.
[37] Szykman S, Racz JW, Sriram RD. The representation of function in computer-based design, ASME Design Technical Conferences, 11th International Conference on Design Theory and Methodology, New York, USA, 12–16 September, Las Vegas, NV, DETC99/DTM-8742. Las Vegas, NV: ASME Press, 1999.
[38] Szykman S, Senufaute J, Srirram RD. Using xml to describe function and taxonomies in computer-based design, ASME Design Technical Conferences, 19th Computers and Information in Engineering Conference, New York, USA, 12–16 September, Las Vergas, NV, DETC99/CIE-9025. Las Vegas, NV: ASME Press, 1999.

Vincent Cicirello is a first year PhD student in the Robotics Institute at Carnegie Mellon University. He graduated Summa Cum Laude from Drexel University with an MS in Computer Science and a BS in Mathematics in June of 1999. His masters thesis work dealt with the problem of retrieving solid models based on similarity. He has worked previously as a software engineer for Knight-Ridder Mediastream as well as for Automatic Data Processing.
Vince Cicirello's research interests include robotics, artificial intelligence, planning/scheduling, machine learning, solid modeling, manufacturing, and information retrieval. He is the recipient of an Honorable Mention in the National Science Foundation Graduate Research Fellowship Program (1999), a National Science Foundation Research Experiences for Undergraduates Award (1998), the Dean's Special Scholastic Achievement Award from Drexel University (1998,1996), the Golden Key National Honor Society/KPMG Peat Marwick Scholarship Award (1997), among other awards. He is a member of AAAI, ACM, IEEE, IEEE Computer Society, and SIAM.

William Regli is an Assistant Professor in the Department of Mathematics and Computer Science at Drexel University and Director of Drexel's Geometric and Intelligent Computing Laboratory. Dr Regli received the PhD in Computer Science in 1995 from the University of Maryland at College Park and a BS (Cum Laude) in Mathematics and Computer Science in 1989 from Saint Joseph's University in Philadelphia. He has previously been visiting research faculty at Carnegie Mellon Univeristy (1997) and a National Research Council Postdoctoral Research Associate at the National Institute of Standards and Technology (1996–1997). During his tenure at NIST, Dr Regli was involved in numerous programs bridging academia, industry, and government; he was the chair of industry–university workshops, including those on process planning and network-enabled design and manufacturing.
Dr Regli's current research interests include solid modeling, artificial intelligence, distributed design and manufacturing and Internet computing. These research efforts are supported by the National Science Foundation, DARPA, AT&T Labs and Sun Microsystems. He is the recipient of a 1998 NSF CAREER Award, the University of Maryland Institute for Systems Research Outstanding Graduate Student Award (1994–1995), NIST Special Service Award (1995), General Electric Corporation Teaching Incentive Grant (1994–1995), among other awards. He is a member of ACM, IEEE Computer Society, AAAI, and Sigma Xi and on the editorial board of the journal *IEEE Internet Computing*. Dr Regli has authored or co-authored more than 60 technical publications.