Internet-based collaborative product design with assembly features and virtual design spaces

N. Shyamsundar*, Rajit Gadh

Department of Mechanical Engineering, University of Wisconsin, Madison, WI 53706, USA

Received 1 November 1999; revised 15 September 2000; accepted 15 October 2000

Abstract

Typically, during product design collaborating, designers share design data and assembly models. However, due to the limited bandwidth of the Internet transfer of detailed CAD based assembly models is slow. Therefore, tools based on polygonized representations of assembly models have been utilized during collaborative design. These models are primarily used for visualization, which is an important activity during modeling. However, providing the designers with the ability to perform real-time geometric modification, assembly constraints specification and concurrent design of different components/sub-assemblies will enable designers to collaborate effectively during design. This paper presents a new geometric representation called the AREP and a prototypical system (cPAD) that uses this representation, which implements the collaboration paradigm by supporting these activities. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Collaboration design; Assembly models; Form features

1. Introduction

In a typical product design scenario, different components or sub-assemblies of the product are designed by different groups of designers at geographically different locations. This is because, original equipment manufacturers (OEM) are out-sourcing engineering activities performed internally [1,2] to remain competitive. However, due to outsourcing of the design activity specification conflicts can arise. By effectively collaborating, these conflicts can be resolved early in the design stage thereby reducing the product development lead-time and manufacturing cost to a large extent [3].

To accomplish this, a compact and comprehensive representation that effectively disseminates product assembly design knowledge, and a tool that provides an integrated interface through which this knowledge can be accessed and manipulated is required. Such a representation and tool is the focus of this paper.

With the increasing availability of the Internet/Intranet, tools that aid designers to share design data and models over the Internet utilize a variety of Internet and web based utilities such as e-mail, web pages, Internet-based conferencing utilities and utilities for collaborative viewing of 3-D product models. This indirect approach of integrating design tools makes collaboration a tedious, unorganized and extremely error-prone. Developing a representation scheme and an internet-centric product assembly design tool based on this representation is the focus of this paper. This tool provides the ability to directly share assembly models and design knowledge encapsulated in terms of features during distributed design of a product assembly, thereby establishing a seamless, integrated and collaborative design space. The representation proposed in this research utilizes the concept of features to minimize the size of the assembly model and at the same time encapsulate assembly constraints in the form of interface assembly features. The use of interface assembly features provides the designers with a new paradigm for collaboration, which can ensure that the components and sub-assemblies designed by different designers are compatible with each other.

A typical scenario for application of the direct approach is when an auto-manufacturer outsources both the design and manufacturing of a sub-system, such as an engine. Prior to outsourcing, the manufacturer typically would request a cost estimate from the supplier. The manufacturer therefore provides the supplier with an approximate shape of the model (the virtual design space, a concept that will be explained in detail later in the paper) that is produced before the detailed model of an engine. In addition, the manufacturer provides the specifications of ‘mating assembly features’ or
‘hard’ points (the interface assembly features, a concept that will be explained in detail later in the paper) where the engine is attached to the rest of the chassis of the automobile. The virtual design space, which the supplier receives, is the envelope within which the supplier must design the engine such that the engine design satisfies the specifications of the mating assembly features that interact with the rest of the product.

2. Related research

This section reviews the previous work on collaborative design as well as assembly representation. Research has been performed to demonstrate the feasibility of collaborative design and deduce the requirements for collaborative design in the Madefast program [4]. The Madefast program validated the Internet as an information infrastructure for collaborative engineering. Based on a study conducted by Cutkosky et al. [5] the authors conclude that in: (a) distributed design several groups communicate utilizing a predefined protocol and (b) during concurrent engineering integration of multitude of models required for the complex design process must be considered. The authors suggest a flexible representation scheme that can represent the product in varying levels of detail. A review of current practices and future trends in the development of a network based computer aided design framework has been presented by Regli [6]. The author presents two technologies that are key to the development of network centric data: (a) CAD browsers currently implemented based on the use of visualization tools based on polygonized models and (b) Internet-aware CAD data were hyper links to related design data are embedded. Maxfield et al. [7] discuss a virtual environment that allows real time collaboration between designers for assembly modeling using constraints. In the research work by Roy et al. [8], collaborating designers create concept models of parts, and iterate until the part design is complete prior to manufacturing analysis of the part.

Floriani et al. [9] and Requicha and Whalen [10] present representations for modeling assemblies. Cutkosky et al. [11], Shah and Rogers [12] and Fazio et al. [13] utilize assembly representations that are based on fixed dictionary of assembly features. The results of their investigations indicate that utilizing the assembly features for design can effectively capture the information necessary for assembly tasks. Wedeking [14] proposes that assembly design start with a spatial envelope of the assembly, which is revised as the iterative design process proceeds. Considerable work on assembly representation for assembly sequencing [15,16], and disassembly sequencing [17,18] has been done.

In summary, existing collaborative tools for assembly modeling typically utilize polygonized models or detailed CAD model of the assembly. Polygonized model support only visualization and do not support activities like geometric modification and constraint specification between components. There are several existing boundary-based or feature-based based assembly representation schemes, which have the drawback that they are detailed representations and are therefore not compact enough for transmitting over the limited bandwidth of the Internet during collaborations. The representations utilized during assembly analysis are typically optimized for the algorithm that analyzes the assembly and cannot be directly utilized for collaborative design. These representation schemes abstract out from the detailed assembly model information required for assembly analysis and store it in a suitable data structure and hence are not suitable for collaborative design.

Therefore, this research utilizes an intelligent polygonized format that maintains pointers to the corresponding entities in the detailed CAD model of assembly. Previous research has validated the Internet as the medium for collaborative engineering providing designers access to computing resources. Also a need for tools that will provide an integrated interface for designers to access online engineering information and services is also reported in existing literature.

3. Requirements for a representation to support collaborative product design

To enumerate the requirements for a representation scheme to support collaborative assembly modeling, it is necessary to examine the framework for a collaborative assembly-modeling tool. Typically, collaboration requires the sharing of programs and data between the designers utilizing networked computers. Distributed architectures have to address critical design issues that are not present in stand-alone systems [19] like, locating programs and data, establishing and maintaining inter program communication, coordinating and synchronizing distributed applications, and failure recovery and security. A model that satisfies these requirements is a ‘client–server’ model [19] which is utilized in this research.

Currently the best medium to accomplish collaboration is the Internet/Intranet. Due to the widespread availability of the high bandwidth Internet design teams can communicate design data. Though the available bandwidth of the Internet may be in Megabits or in some cases Gigabits the effective bandwidth [20] available is much lesser. During collaboration several teams are working on different products and utilize the same network that effectively reduces the available bandwidth. Also as the bandwidth increases, development and use of Internet centric applications also increase thereby reducing the effective bandwidth. Therefore, in spite of development of high bandwidth networks the effective bandwidth available is limited and its usage needs to be optimized.

Summarizing from the previous discussion, the product assembly representation scheme should be: (a) compact, (b) encapsulate the assembly design specific information and...
(c) provide a unified approach to represent the product model on the server as well as the client.

4. Definitions and assembly representation

The assembly representation (AREP) consists of four classes of information, namely: (a) a directed acyclic graph, DAG (called an assembly hierarchy graph, AHG) that specifies the hierarchy in the assembly, (b) relational graphs, (c) assembly level information and (d) information related to an individual unit within an assembly (called an Assembly Unit or AU which can be components or sub-assemblies or envelopes).

From existing literature on assembly representations the assembly hierarchy can be a flat relational representation capturing the geometric and functional relations between components or a hierarchical relational representation relating components and sub-assemblies or two sub-assemblies.

A hierarchical relational representation (HR) is more efficient for representing the assemblies as compared to a flat representation.

In this paper, a modified hierarchical representation (AHG) is proposed. The differences between the hierarchical representation and the modified representation are: (a) a HR is represented as a tree, while a DAG represents the AHG and (b) each node in a HR can be either a component or a sub-assembly, while in AHG it can be an AU. The AHG gives greater flexibility because it can represent multiple interpretations of the same assembly. If HR is utilized a new tree has to be constructed if the hierarchy changes. In contrast, if a DAG is used to represent the assembly hierarchy new sub-assemblies can be subsequently appended to the existing representation. Given an assembly A, the set of all components in A is the elemental set \( A^E \), which is the most detailed and complete representation of the assembly geometry, it lacks the design intent due to the absence of sub-assemblies. For an A there can be \( k \) representations of the model each representing a different interpretation of the assembly model:

\[
A^E = \{C_1, ..., C_n\} = \{AU_1, ..., AU_n\}, \quad \text{where} \quad AU_1 = C_1, ..., AU_n = C_n. \quad (1)
\]

\[
A^{(1)} = \{C_1, ..., C_n\} \cup \{SA_1, ..., SA_n\} = \{AU_1, ..., AU_{n1+n2}\}, \quad (2)
\]

where \( AU_1 = C_1, ..., AU_{n1} = C_{n1}, \quad AU_{n1+1} = SA_1, ..., AU_{n1+n2} = SA_{n2} \)

\[n1+n2 = SA_{n2} \]

\[A(k) = \{C_1, ..., C_n\} \cup \{SA_1, ..., SA_{nk}\} = \{AU_1, ..., AU_{nk} + nk\}, \quad (3)\]

where \( AU_1 = C_1, ..., AU_{nk} = C_{n1}, \quad AU_{nk+1} = SA_1, ..., AU_{nk+1} = SA_{nk} \)

\[SA_{nk+1} = SA_{nk} \]

Several interpretations are required because A is utilized during analysis, maintenance, marketing, and disassembly for disposal in addition to assembly modeling where different assembly hierarchies may be required during modeling and design of different components and sub-assemblies. This can be accomplished by using the AHG. In Fig. 1 two interpretations of an assembly A are illustrated:

\[
A^{(1)} = \{AU_1, AU_2, AU_3, AU_4, AU_5\} \quad (4)
\]

\[
A^{(k)} = \{AU_4, AU_5, AU_6, AU_7, AU_8\} \quad (5)
\]

In addition the relationship graph for the AUs is also illustrated. Since the AHG represents assemblies in varying levels of detail in a hierarchical manner, the representation
supports assemblies with a large number of components. This makes the representation scalable.

Class 2 information consists of a graph in which the nodes represent AU and the edges represent relationships between AUs. The relation represented by the edge can be based on geometric entities, like faces, or other higher-level abstractions, for example, form features consisting of relations between AUs. Derivation of this graph is explained in detail in the subsequent sections.

The assembly level information (Class 3) is the number of AUs contained in the assembly, list of AU forming sub-assemblies. The AU level information (Class 4) includes a representation of AU and related information explained in the subsequent section. The following section explains the derivation of the AU graph (Class 2), which in this case is an assembly feature relation graph (AFRG).

4.1. Assembly features

Shah and Mantyla [21] define a component feature as a property (for e.g. a blind hole) of a single component. Some of the definitions of assembly features presented in the literature are listed below.

- An assembly feature is an association between two form features on different parts, i.e. geometry that belongs to different parts, expressed in a canonical form [12].
- A feature is any geometric or non-geometric attribute of a discrete part whose existence or whose dimensions are relevant to the product’s or part’s function, manufacture, engineering, analysis, use, etc. or whose availability as a primitive or operation facilitates the design process. An assembly feature is the elementary relation between components extended with some assembly information [13].
- Grouping of various feature types to define assembly relations, such as mating conditions, part relative position and orientation, various kinds of fits and kinematic relations [21].

In this paper an assembly feature is defined as a property of an AU with respect to (or in the context of) other component(s), which provides assembly related information relevant to the design, manufacture or function of the product assembly. Assembly features can be classified into relational assembly features and assembly form features. Relational features indicate a specific relation between two geometric features (GF), which can be form features or geometric entities like the axis or a surface of revolution. For example, joints, spatial positioning of one component with respect to the other and clearances are relation assembly features. Also, certain shape features belonging to two components can be joined together to form another shape feature (for example, a peg or a hole) and these are called as assembly form features. These are two basic types of assembly features. These two features can be combined to form a derived type called the combination features.

In this paper an assembly feature is denoted by \( AF_{ij} \), where AF is the \( r \)th assembly feature of AU and AU. If AF is a binary relational assembly feature, a feature which relates the geometric feature GF on AU and GF on AU by a relation \( \Psi \). If AF is a form assembly feature, then the combination of the form features GF on AU and GF on AU have a specific geometric or topological property. The same concept of relational and form features can be generalized to relations involving more than two geometric features. A feature that is a combination of a form feature as well as a relational feature is a combination feature.

4.2. Specifying assembly features

In this paper, an assembly feature definition language (AFDL) is utilized to specify these constraints. The vocabulary of AFDL consists of geometric features (GFs), relational constraints, and form constraints. In this paper, a GF is assumed to consist of a group of faces on an AU. The relational constraints (denoted by \( \Psi \)) include mating, alignment and offset of two faces which are typically used to orient one AU with respect to another. The form constraints (denoted by \( \bigcup \)) involve specification of relational constraints between two GFs to result in a specific shape or form, typically used to satisfy some functional aspect of the assembly or to be combined with additional relational constraints to result in a combination assembly feature.

The syntax for defining a relational AF is \( GF_{i}^{a} \Psi GF_{j}^{b} \ldots \) and for a form AF is \( GF_{i}^{a} \bigcup \Psi^{c} GF_{j}^{b} \). Since GF is defined in terms of faces belonging to an AU. Let \( GF_{i}^{a} = \{ F_{a}, F_{b}, F_{c} \} \) and \( GF_{j}^{b} = \{ F_{a}, F_{b}, F_{c} \} \). Then \( AF_{ij}^{c} \) which is \( GF^{a} \Psi^{c} GF^{b} \) can be represented by its elemental relations in terms of the faces as \( \{ (F_{a}, \Psi_{c}, F_{a}), (F_{b}, \Psi_{c}, F_{b}), (F_{c}, \Psi_{c}, F_{c}), \} \). Therefore, to specify an AF the GFs should be specified first and then the elemental relationships, which are denoted by \( \Psi_{c}, \Psi_{b}, \Psi_{c} \) (in the previous example) need to be specified. It is to be noted that the order of the corresponding faces belonging to \( GF_{i}^{a} \) and \( GF_{j}^{b} \) are preserved. Depending on the nature of the constraints, specification of \( AF_{ij}^{c} \) may completely constrain the position of AU with respect to AU or there might be one or more degrees of freedom depending on the design intent.

Examples of relational AFs is shown in Fig. 2 where AU and AU are constrained such that GF, which is the cylindrical hole and GF the cylindrical hole in AU are spaced at a fixed distance and their axes are aligned and form a spatial alignment AF. In Fig. 3(a) GF (the cylindrical boss) on AU (turbine) and a similar boss GF on AU (fan) are related such that they are mating along the end faces and the axes of the bosses are aligned and result in a bridge AF. In Fig. 3(b), when AU and AU are joined together they form a ‘hole’ assembly form feature, while in Fig. 3(c), when AU and AU are joined together they form a ‘boss’
assembly form feature. It is to be noted that all the AF examples relate to only two components and the relationship is binary. However, AU1, AU1, and AU2 (Fig. 3(b)) together form an insertion AF (GF1,1 < GF1,2) and therefore, this AF is a ternary relation. The next section presents a data structure for storing the AF information for a product assembly.

5. Interface assembly features

The interface assembly features (IAF) are a subset of the assembly features present in a product assembly. The interface assembly features are those assembly features that are utilized during the design phase to ensure that the designs of related sub-assemblies or components are compatible. The mating points of two sub-assemblies or components are examples of interface assembly features.

The interface assembly features AF1,9;10, AF1,9;11, AF1,10;11 (Fig. 4) which capture interactions between the (AU9 and AU10), (AU9 and AU11) and (AU10 and AU11) are defined subsequently. This assumes that AU9, AU10, and AU11 are all designed by different collaborating designers. These interface assembly features are 'hard constraints' that the designers cannot modify unilaterally during the design of AU9, AU10, and AU11. Any changes in the corresponding geometric features (interface geometric features, IGF) that comprise the interface assembly features have to be negotiated with other designers. Due to the detailed specification of the interface assembly features, once the concurrent design of the assembly is complete, the design of the different AUs will be dimensionally compatible.

6. Assembly feature relational graph (AFRG)

The AFs and the relations between the AFs can be represented by a graph called the Assembly Feature Relation Graph (AFRG), which is a type of AU relationship graph. An example of an AFRG is shown in Fig. 5. The nodes represent AU1 and AU2, and the edge represents AF1,2. This graph is a hyper graph since it represents AFs that can relate to more than two AUs. In Fig. 5, there are seven AUs and there are seven relational assembly features that have been specified in the AFRG. These features are sufficient to specify the relative positions of each AU with respect to the other AUs.

Having proposed a data structure for storing the relationships based on assembly features, the next section explains how the assembly features are utilized to produce a special kind of envelope called the virtual design space.

6.1. Virtual design space for collaboration

Since each component or sub-assembly is designed by
different designers or groups of designers (some of them belonging to different companies or suppliers), it is essential that these designs be compatible with each other for reducing the design lead-time. This implies that there is no volumetric intersection between components of an assembly. Therefore, designers need to negotiate the design space within which they will be designing their respective components or sub-assemblies and decide on the dimensions of interface assembly features before starting the final design phase. Alternatively, given the final product volume, the volume is partitioned into sub-volumes within which the components or sub-assemblies comprising the product are to be designed. These sub-volumes are also called as the virtual design spaces, VDES, of C_i or VDES, of SA_i. The virtual design space is a special kind of envelope with interface geometric features. The rest of the section focuses on envelopes and use of interface geometric features to define a VDES.

There are different kinds of envelopes ranging from approximate envelopes like the convex hull of components to tighter envelopes that comprise of the external faces of a component or sub-assembly. The approximate envelopes are commonly utilized and include the bounding sphere, bounding box, convex hull and other bounding shapes. These geometric shapes are approximate estimates of the volume of the component or sub-assembly they envelop.

A tight envelope is based on defining a simplified model of an existing part or sub-assembly. There are two ways to accomplish this: (a) create a schematic model of a component or sub-assembly or (b) simplify the existing model of a component or sub-assembly.

For example, consider a motor that is part of a fluid pumping system. The approximate model of the motor consists of the outer casing, shaft (Fig. 4) and external components of the motor, while the internal components of the motor like the rotor and stator are not part of the tight envelope of the motor.

However, if the exact model of the component or sub-assembly is available, then there are several methods for constructing a tight envelope. For a given component, features that are critical to assembly modeling, namely interface geometric features are included in the tight envelope while other features are eliminated from the model. Most commercial CAD systems are feature-based modelers.
and provide the functionality to delete specific shape features from the model. Examples of some features that are not critical for assembly modeling include chamfers and fillets that are not located on assembly features.

For a given sub-assembly, the exact CAD model of the sub-assembly can be simplified in two steps. In the first step components not required for assembly modeling at a particular level of abstraction (or a given interpretation) are eliminated (for example internal components). This is a special case of hierarchical simplification that is applied to a sub-assembly to obtain a simplified model of the sub-assembly. In the second step features that are not interface geometric features are eliminated resulting in a simplified model of the component or sub-assembly.

This is illustrated utilizing a hard disk assembly shown in Fig. 6. The aim is to produce a VDES of the hard disk sub-assembly to be utilized during the design of a computer. The internal components of the hard disk and a bottom view of the assembly are shown in Fig. 6(a–c) respectively and the corresponding TE is shown in Fig. 6(d). In this simplified model of the assembly the internal components are eliminated and remaining components (AU1, the base and the connector assembly AU2, and the cover for the base) are simplified and are combined utilizing Boolean union to produce the TE of the hard disk. The detailed CAD model of the base (AU1) and its simplified model obtained are shown in Fig. 7(a–c) respectively.

During this process of geometric feature based simplification, those features (interface geometric features) that are required to mount the hard disk in the computer and other external connectors (for connecting the hard disk to the controller) that are required for assembly design of the computer are preserved while simplifying AU1 and AU2. This results in defining the TE of the hard disk and the subsequent specification of the IGFs results in the VDES of the TE.

If CAD modelers do not support design features, a set of faces are selected from the original model to serve as the TE of the component or sub-assembly. The definition of TE is not an automatic algorithmic process. It is created based on negotiation between designers and an additional factor is the level of abstraction as defined by the assembly interpretation required by the designer.

Once the IGFs are specified for an envelope (E, which can be an approximate envelope or TE), the IAFs are specified utilizing the corresponding IGFs on related envelopes of the sub-assemblies or components.

Previously, it was defined that the AU represents a component, sub-assembly or an envelope. It is to be noted that the AU can also represent a VDES since it is a special type of envelope. In the subsequent discussion it is assumed that the VDES is equivalent to an AU.

Fig. 8 illustrates the method of collaboration between the supplier and manufacturer utilizing VDES. As discussed previously, the supplier is provided with the VDES of the dash board which comprises of the envelope of the sub-assembly to be designed, in this case the dashboard and the attachment points, which constitute the interface assembly features. When the design is completed the detailed design of the cab produced by the manufacturer is combined with the detailed design of the dashboard produced by the supplier. Designing of the dashboard within the virtual design space ensures that there are no volumetric intersections between the components of the dashboard and the
rest of the cab. The specification of the interface assembly features ensures that the hard constraints like mating, clearance etc. across the two designs are compatible. A detailed explanation of the architecture of the implemented system is discussed in the next section.

7. Implementation

It is envisioned that during Internet-based collaborative product design, both the design knowledge and expertise, which are distributed at geographically different locations, can be brought together into a common collaborative design space. The architecture of a prototypical system, cPAD in short for collaborative product assembly design system, that supports creations of a common collaborative design space is proposed in this paper (Fig. 9) and includes the following modules.

- An intelligent server (IS) that redirects requests from the clients to the appropriate application server.
- Application servers (ASP) that offer a particular service for the client.
7.1. Intelligent server

The intelligent server or the broker is responsible for directing the client request to the appropriate application server. The IS maintains a list of available ASPs and the service they provide. Additional information such as the number of requests an ASP can process and a list of backup ASPs that offer the same service are stored. If a fault tolerance mechanism is implemented in the broker, if an ASP crashes it is automatically detected and the client request is forwarded to another ASP that provides the same service. In the current research, this server is implemented in Java.

7.2. Application servers

The following are the application servers which are utilized in cPAD (Fig. 9).

- A web server (ASPwww) where the designer can download the Java client application/plug-in that provides the client interface. A standard web server (Apache web server) is utilized in the system.
- A solid modeling server (ASPmodeling) that utilizes the Parasolid kernel for modification of the product model. The solid modeling kernel provides access to basic geometric manipulation tasks on the solid model of a component. Several commercial CAD systems utilize this kernel for implementing their CAD systems. In addition, during the creation and modeling of an assembly one component is to be positioned with relative to the other components utilizing spatial constraints like mating or alignment between two faces. When such constraints are specified, the constraint equations must be solved to obtain the transformation to be applied on one object to transform it relative to the other object. For 3-D constraint solving, a commercially available constraint solver is utilized. This is utilized in a number of commercial CAD systems for constraint solving. This server is implemented in C++.
- A CAD database (ASPdbase) server stores the product model in a Microsoft Access database. This server is implemented in Java and access to a Microsoft Access database is provided using JDBC (Java Data Base Connectivity).
- A catalog server (ASPcatalog) maintains a repository of existing design data and models. This server is implemented in Java and access to a Microsoft Access database is provided using JDBC.
- A visualization server (ASPvisual) provides a triangulated model of the product model for display at the client end. This also needs to utilize the Parasolid kernel that provides the functionality to facet or triangulate the solid model to produce the display file format file for the object. This server is implemented in C++.
- A server for processing miscellaneous client requests (ASPmisc) like authentication, file retrieval and other requests. This server is implemented in Java.
7.3. Client interface

The client interface in the cPAD system was developed using Java. The graphical user interface is based on Swing and the 3-D visualization and interaction is accomplished using Java3-D. The operations supported by the client are as follows.

- File operations: Secure saving and retrieving of a product model in personal as well as group database. Retrieving models for viewing and editing. Uploading a CAD model from the client to the server for modification and storing in the server database.
- Visualization operations supported are transformation, scaling and zooming.
- Creation of basic primitives and envelopes, modification of the model utilizing Boolean operations, specification of geometric features and subsequently specifying the assembly features utilizing these geometric features, attaching annotations to the models, setting up a hierarchical model (based on AHG) of the product that specifies the product in multiple levels of detail.
- Integration of the design with engineering catalogs which provides the designer with an interface to browse catalogs and download the required models.

7.4. Implementation of the AREP

As discussed earlier, the AREP consists of four classes of information, which are implemented utilizing a collection of modular data structures called the AU–REP as opposed to a monolithic data structure that is typically utilized in standalone CAD systems. There exists an AU–REP for each AU. The AU–REP (Fig. 10) for an AU consists of information belonging to three classes.

1. AU level information: (a) the node identifier associated with the AU, (b) the number of GFs defined for the AU (≥m, where m is the number of assembly features defined for a given AU), (c) the boundary representation of the AU and (d) annotation text associated with AUi. For every face constituting the AU, the following information is stored: (a) the face identifier, (b) surface type and (c) GF information. For any given AU, there will be several GFs specified each is defined as a collection of faces. A given face can be included in the definition of r such GFs. The GF information includes (a) r and (b) the list of the identifiers of each GF and the associated elemental relation is stored.

2. AHG link information: The AHG is a DAG and the set of terminus nodes of all edges whose origin is the current node, V is stored (in Fig. 10, the set V for a sample AUi is illustrated).

3. AFRG edge information: In the AFRG let the number of edges incident on node AUj be m and the set of neighbors of AUj be W. For each AUj ∈ W there exists an edge E such that it defines AFj = GFjΨGFj (where Ψ represents Ψ or Ψ) and the set of all GFj is X. The set W and the set X are stored. (The set W for a sample AUi is illustrated in Fig. 10.)

The AREP is implemented utilizing AU–REP as follows.

- Assembly level information: For a given interpretation A(0), it represents a set of AUs constituting that interpretation. This can be represented as a one level tree consisting of a single root node and child nodes that are the elements of the set A(0). The root node is an envelope of the entire assembly, which can be stored in the AU–REP data structure previously discussed. The list of all AUs constituting A(0) is stored in the AHG link information field of the data structure. The remaining fields of the AU–REP data structure are not utilized.

- The information pertaining to each AUs constituting A(0) is stored in an AU–REP data structure.

- To construct the AFRG of A(0), the necessary information is contained in the AFRG edge information. The set of all edges comprising the AFRG is represented by the set of all AFRG edge fields of all the AUs comprising the set A(0).

These three classes of information provide a complete representation of A(0).

There exists only one AHG for the entire product model. To
obtain the AHG that is a DAG, the AHG link information field of each AU is utilized. The set of all edges comprising the AHG is represented by the set of all AHG link information field of all the AUs comprising the sets $A^{(1)}$, $\ldots$ $A^{(k)}$, and $A^E$.

There are two versions of AREP for any given interpretation $A^{(i)}$ that are utilized in the cPAD.

1. The first version of the representation is the primary representation that is stored on the server. This consists of the complete representation of $A^{(i)}$ as previously discussed. The boundary model field of the AU–REP for each AU comprising $A^{(i)}$ consists of the KFF (kernel/Parasolid file format) based representation of the geometry of AU.

2. The other version is the secondary representation that is utilized in the client. This is different from the primary representation in that the boundary model field of the AU–REP for each AU comprising $A^{(i)}$ consists of the compressed GFF (graphics/polygonized file format for display) based representation of the geometry of AU.

It is the client side representation, the secondary representation that is transmitted across the network to the client and hence has to be compact.

As previously discussed, utilizing a GFF based representation to represent the geometry of AU is suitable only for visualization. For modification and other related tasks, access to KFF based description of AU is required. Such a representation is stored in the primary representation of AU. Therefore, an augmented GFF is utilized in the secondary representation. This mapping is at the AU level as well as at the face level.

- AU level mapping: The node identifier field of the AU–REP provides a mapping between the primary and secondary representations of an AU.
- Face level mapping: The face identifier field of the AU–REP provides a mapping between the faces of the primary and secondary representations of an AU.

These mappings overcome the limitation of utilizing a polygonized representation of assembly as a primary representation and operations like modifying the geometry and querying geometric information is possible utilizing the representation proposed in this paper. Simple activities like viewing of the model can be accomplished on the client utilizing the augmented GFF while complex activities like modification of the geometry is accomplished on the server utilizing the exact geometry stored in KFF. Further in this research the polygonized mesh produced is compressed utilizing a utility built into Java3-D further reducing the size of the secondary representation.

For any given product assembly there exist $k$ interpretations $A^{(1)}$, $\ldots$ $A^{(k)}$, in addition to $A^E$. Let the set one of the $k$ interpretations available $A^{(1)}$, $\ldots$ $A^{(k)}$. The universal set $U$ consists of all the AUs comprising the interpretations $A^{(1)}$, $\ldots$ $A^{(k)}$, and $A^E$. The server maintains the primary representation version of the AU–REP for each AU, $\in U$. This is primary product assembly representation and is stored in the ASPvisual and constitutes the complete AREP of the product assembly. At any given time the client has the complete representation of one of the interpretations of the assembly.

8. Results and discussion

As earlier indicated the use of the cPAD supports real time modification of the models over the Internet. An example where the bore diameter of a connecting rod is increased utilizing the system is illustrated in Fig. 11. The system provides a CSG based interface for model editing. These models can be created in the system or may be imported from other CAD systems, which utilize the Parasolid solid modeling kernel. To import a model into cPAD, the designer selects the model to be imported from a list of files available on the client. The data is then uploaded to IS, which then transmits the data to ASPvisual. ASPvisual produces the secondary AU–REP for the model and sends the data to the client through IS. The designer is then able to visualize the model and interact with the model. The visualization activity is performed entirely on the client to provide high degree of interactivity to the user while detailed geometric operations are performed on the server utilizing the mapping between the secondary and primary representation.
The specification of assembly features for connecting rod assembly is illustrated in Fig. 12. Here the geometric features GF$_1^1$, and GF$_1^2$ are specified and the constraint between the geometric features are specified resulting in definition of GF$_{1,2}^1$. In this example, the face mapping between the secondary and primary representation is utilized providing the designer with the ability to highlight the specific geometric features on AUs and specify constraints. The constraint solving is performed on the server utilizing the feature highlighted on the client and the resulting transformations are returned to the client and one of the AUs is repositioned with respect to the other.

A typical scenario during the design of a connecting rod, piston assembly utilizing the VDES is illustrated in Fig. 13. Here the piston is to be designed. The assembly is represented in terms of the VDES of the connecting rod, piston and pin. The interface geometric features consisting of the top bore of the connecting rod, the diameter of the pin and the diameter of the pin’s bore in the piston are defined. Then the piston design is performed on a CAD modeler and the detailed design is integrated into the model of the connecting rod, piston assembly in the cPAD system.

A comparison of the reduction in the file size when an assembly is represented utilizing different interpretations is illustrated in Fig. 14. In this figure the four different interpretations of a connecting rod, piston assembly are shown where $A^{(1)}$ is the most simplified representation while $A^{(3)}$ is a more detailed representation and finally $A^{E}$ is the elemental assembly the most detailed representation. As indicated previously it is the compressed polygonized representation of the AUs that is transmitted to the client from the server through the Internet and the mapping between the secondary and primary representation provides the designer with the ability to modify the geometry in real time, specify assembly features and design the product utilizing the VDES. A 365 KB CAD based representation of $A^{E}$ has been compressed to a 47 KB (87.12% reduction) compressed geometry based assembly representation when $A^{(1)}$ is utilized, to 107 KB (70.68% reduction) when $A^{(2)}$ is utilized and to 110 KB (69.86% reduction) when $A^{(3)}$ is utilized. Hence, significant compression of the assembly model can be accomplished using the proposed representation scheme.

As indicated previously, the VDES is produced as a result of negotiations between the collaborating designers. For such an activity the initial estimates of the VDES need to be made. Typically, all components are not redesigned; some standard
components are utilized as indicated previously. Often, entire sub-assemblies are standardized. On the other hand some of the components are redesigned from existing designs. This implies that the shapes of most of the components in the product or that of the entire sub-assemblies are known and a few of them are unknown. The designs of few of these unknown components or sub-assemblies cannot take arbitrary shapes. Typically there are design rules that such new designs must comply with. These are useful in estimating the shapes of all the VDES of the product before the designers can proceed with the design of the product.

The use of the VDESs decomposes the design process in a hierarchical manner and also ensures compactness of the representation that is essential for Internet-based collaborative design. The interface assembly features ensure that the designs that are produced are compatible with each other. The AU that represents the VDES based on envelopes and interface assembly features form the basis of the AREP, which from the previous discussion has been specifically designed for a collaborative environment and hence provides a new paradigm for collaborative product design.

It is evident from the previous discussion that due to the limited bandwidth of the Internet, transferring the detailed CAD model of the product over the Internet would be a very slow process. In the rest of the section a CAD representation of the product is compared with an envelope based representation like the AREP.

The second type of data, the non-geometric data is also required during the design process. As in the case of geometric data, the non-geometric data for all the components and the sub-assemblies is not required. Only each designer requires a sub set of this data. Therefore, suitable abstractions should be developed to extract the required information for the design activity being performed and appended to the AREP. This is not discussed in this paper and would be an area of future research. The contributions of this research are summarized in the next section.
In summary the AREP has the following features.

- The AREP is a compact representation, as the assembly model is compressed or simplified in two stages: (a) at component level and (b) at the feature level.
- The AREP provides an adaptive simplified representation scheme in which the details necessary for assembly modeling are encapsulated in terms of assembly features.
- The AREP provides a unified representation scheme for storing multiple interpretations of the assembly model as well as representing the assembly model both at the client and the server end.
- The use of an assembly feature definition language makes the assembly representations extendible. The user can specify additional assembly features as required by the application for which the representation is utilized.

In this research constraint propagation across multiple interpretations of the assembly model is not currently implemented. A mechanism to propagate assembly constraints that are encapsulated in the form of assembly features would be a useful addition to the system. The representation currently does not store the number of degrees of freedom of an assembly feature. If this information is available and the directions of motion of one AU with respect to the other can be captured by the representation and the analysis capabilities can be added to the system. In the current research assemblies consisting of nominal geometry was considered. Additional work is needed to incorporate tolerated geometry into the AREP representation scheme.

9. Contributions

The AREP is based on the hierarchy of the top-down design process and therefore provides a mechanism to resolve specification conflicts early in the design stage and enables concurrent design of the product. A new paradigm for collaborative design based on the AREP is proposed. The representation adaptively simplifies the model of the product thereby preserving the details (encapsulated in the interface assembly features) of a particular component or sub-assembly required during the product assembly design. Therefore, it is a compact representation for transmitting the product model over the Internet. In addition, the designers are able to interact with a product model that provides the required level of abstraction.

The cPAD system which implements the collaborative paradigm provides a networked environment for product assembly design that integrates collaboration, solid modeling and features based design. The cPAD client functions as a browser that interfaces with product design data and resources.

10. Summary

Due to the widespread use of computer aided design tools as well as the Internet, the product design process is being transformed into a collaborative activity wherein designers share product models as well as related data to work together despite geographically distant locations. The file sizes of CAD based product models are typically very large and their size constrains their use in a collaborative Internet centric environment. Therefore, the data transfer has to be minimized and a common approach to this problem is utilizing a representation, which abstracts the required information and transmit it across the Internet. In this paper a representation scheme that can effectively capture the necessary information was presented.

This representation of the assembly (called the AREP) stores the assembly hierarchy as well as relations between components and sub-assemblies with the use of a graph. The nodes of the graph represent the components or sub-assemblies or virtual design spaces and the edges of the graph represent the relation between components and sub-assemblies. These relations are defined in terms of assembly features, which capture the interaction of geometric features of the components/sub-assemblies. The assembly features are themselves defined in terms of geometric features that are available in a CAD model, or may have to be explicitly extracted. The assembly features called interface assembly features are used in conjunction with the virtual design space, which is obtained after negotiations between the collaborating designers to design the product. It is to be noted that this representation is scaleable as well. The assembly abstractions that constitute the AREP allow representation of the product assembly in a compact manner for Internet-based collaborative design involving companies and subcontractors. The collaborative product assembly design system based on this representation provides the designers not only the capability to view but also modify, specify assembly features and concurrently design assemblies thereby enabling designers (manufacturers and suppliers) to collaborate effectively during assembly design.

Appendix A. List of definitions

\( A^E \) Set of all components comprising the assembly (elemental set)

AFDL Assembly feature definition language

AFRG Assembly feature relational graph

AHG Assembly hierarchy graph

ASP Application server

AF\(r_{ij}\) The rth assembly feature of AU\(_i\) and AU\(_j\)

\( A^{(r)} \) Set of all components comprising the rth interpretation of an assembly

AREP Assembly representation

ASP\(_{database}\) Database server

ASP\(_{misc}\) Server processing miscellaneous request

ASP\(_{modeling}\) Solid modeling server

ASP\(_{visual}\) Visualization server

ASP\(_{www}\) Web server

AU\(_i\) \(i\)th Assembly unit
AU–REP Representation of an AU
BREP Boundary representation
C_i The ith component
cPAD Collaborative product assembly design
DAG Directed acyclic graph
GFF Graphics file format
GF_i The ith geometric feature of AU_i
HR Relational representation
IAF Interface assembly feature
IGF Interface geometric feature
IS Intelligent server
K Total number of interpretations of an assembly
KFF Kernel file format
M Number of assembly features defined for a given AU or number of edges incident on node representing a given AU
r Number of geometric features whose definition include a given face of an AU
SA_i The ith sub-assembly
TE Tight envelope
U The set of all AU comprising A(1),…,A(k), A_k
V The set of terminus nodes of all edges whose origin is the current node
VDES Virtual design space
W The set of neighbors of a node in the assembly feature relational graph
Ψ Relational constraint between geometric features
Ψ_u Relational constraint between faces
U # Shape constraint

References