

A non-interfering selective disassembly sequence for components with geometric constraints

HARI SRINIVASAN and RAJIT GADH*

Department of Mechanical Engineering, 1513 University Avenue, University of Wisconsin-Madison, Madison, WI 53706, USA
E-mail: gadh@engr.wisc.edu

Received September 1999 and accepted October 2000

This paper analyzes the problem of global disassembly of a selected component from a geometrically constrained assembly. The geometric constraints are: (i) spatial constraints due to the three-dimensional geometric interactions between the components in an assembly; and (ii) user-defined constraints, such as grouping of components as subassemblies and directional constraints on component geometry. A new algorithm has been proposed to determine a non-interfering disassembly sequence for a selected component, minimizing the number of simultaneous component removals. The algorithm analyzes both the spatial constraints of the assembly geometry and the user-defined constraints in evaluating the accessibility of components, which is followed by determining the topological disassembly ordering of the components to evaluate an optimal sequence. Preliminary implementation results of the algorithm for a geometrically constrained assembly are presented.

1. Introduction

Selective disassembly (SD) analysis involves determination of a sequence and directions to disassemble a selected set of components (C) from an assembly (A) (Srinivasan *et al.*, 1997). Applications for SD include assembling, maintenance and recycling. For example, aircraft engine maintenance requires the SD of the engine—which may involve prior disassembly of some components before disassembling the engine—but does not require the disassembly of the entire aircraft. Therefore SD analysis is an important area of research in disassembly planning (Srinivasan *et al.*, 1999).

Despite the advances in disassembly planning (Homem de Mello and Lee, 1991), there has been little investigation in automated SD of geometric models. In contrast to the previous work that focuses on disassembling of all the components in A , the current research focuses on SD analysis.

1.1. Problem analysis: geometric constraints and automation

This paper analyzes the problem of automatically determining the sequence (S) to disassemble a selected component from the geometric model of a constrained

assembly A . The geometric constraints in this paper are of two types:

- *Spatial constraints*: Constraints imposed in assembling or disassembling of a component due to the spatial position and geometry of all other components in A .
- *User-defined constraints*: Constraints imposed by the user on the component geometry that restricts some assembly/disassembly operations. User-defined constraints include component grouping (two or more components are grouped as a subassembly) and directional constraints (one or more possible assembly or disassembly directions for the components are constrained). For example, a user may wish to group a set of components based on the functionality of the subassembly. As another example, the user may wish to impose directional constraints to specify practical restrictions imposed in disassembling due to fixtures, the disassembly workspace, etc. To illustrate, when considering how to disassemble an automotive engine mounted within a car chassis, generally the user will want to restrict his or her analysis primarily to component removal directions that will allow them to be removed from the open hood of the car, rather than through its firewall, side panels or from the bottom of the car.

Spatial constraints on $C_i \in A$ refer not only to the constraints that result due to the mating components of

*Corresponding author

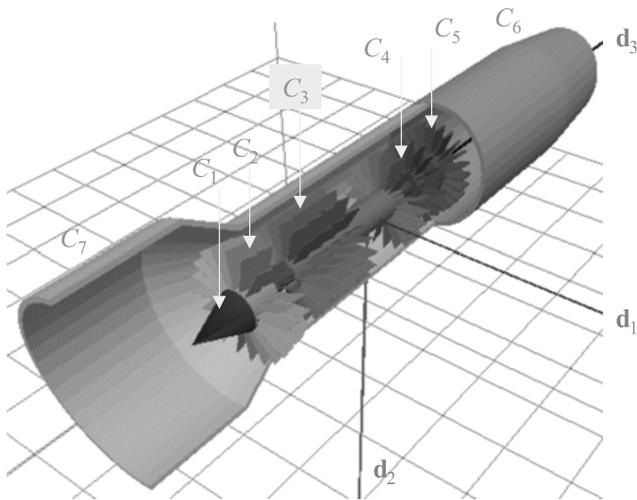


Fig. 1. Engine subassembly to illustrate SD for spatial constraints.

C_i but also due to the components that may collide with C_i in different directions. If only contact-geometry for SD analysis is considered, interference arising during linear global assembly/disassembly motions will not be detected. To illustrate this, consider A shown in Fig. 1 with $C = \{C_3\}$. Let a direction vector be denoted as \mathbf{d}_j . C_3 is in contact with C_4 (along the cylindrical surface). Performing the assembly/disassembly analysis based on contact-geometry alone will result in a solution saying that C_3 is disassemblable along \mathbf{d}_3 and $-\mathbf{d}_3$. However, C_3 is disassemblable locally (infinitesimal translation is possible along \mathbf{d}_3 and $-\mathbf{d}_3$), not globally, i.e., C_3 is spatially constrained by $\{C_4, C_5\}$ along \mathbf{d}_3 and by $\{C_1, C_2\}$ along $-\mathbf{d}_3$. Therefore, SD analysis based on spatial constraints becomes necessary to ensure global assembly/disassembly of components.

1.2. Objective and assumptions of the current research

This paper proposes an algorithm for global SD of the components by analyzing the geometric constraints imposed on them. The components that can be removed independently of each other are simultaneously disassembled. The objective is to use simultaneous removals in place of sequential removals wherever possible, and then to minimize the number of simultaneous removals for SD of components. These objectives equate to the minimization of disassembly operations, and are a measure of the difficulty of disassembling. Therefore, S with minimum simultaneous removals is defined as an Optimal Sequence (OS).

The assumptions for the current research are:

1. The relative motions of the components are determined without considering the tools, fixtures or robots required to achieve these motions.

2. Assemblies are assumed to be rigid (non-deformable), frictionless and defined by nominal geometry (no tolerances).
3. Components are 1-disassemblable (single linear motion to be removed from A).
4. Disassembly sequences are monotonic (components are totally removed while disassembling), and non-destructive.

Assumptions 1–4 are standard assumptions in automated assembly/disassembly analysis (Woo and Dutta, 1991; Beasley and Martin, 1993; Wilson *et al.*, 1995). Assumption 1 requires fixture elements to be modeled based on the sequence determined, or modeled as constraints to components, or modeled as a component with constraints (Homem de Mello and Lee, 1991). The 1-disassemblable assumption is reasonable, since design for manufacturing recommends simple motions for disassembly (Anderson, 1990). The 1-disassemblable assumption is utilized because automated disassembly that allows general disassembly motion is computationally expensive (Goldwasser *et al.*, 1996). Moreover, this assumption is realistic for some real world examples, as indicated by some existing assembly planning systems (Kaufmann *et al.*, 1996).

1.3. Overview of the current research

A new approach called the *Global Selective Disassembly* (GSD) Algorithm is proposed to determine a non-interfering (collision free) optimal sequence to disassemble a selected component from a geometrically constrained assembly.

The algorithm models both the spatial constraints and the user-defined constraints as geometric constraints to components, which were free to move in any disassembly directions in the absence of constraints. The algorithm analyzes the geometric constraints in determining the accessibility of components, which is followed by determining the topological disassembly ordering of the components to evaluate an optimal sequence.

The algorithm has polynomial computational complexity and is implemented and tested in a prototypical software system *Assembly and Disassembly in Three-Dimensions* (A3D). The results from preliminary testing of automotive and aerospace subassemblies for maintenance and assembling applications show that the GSD algorithm evaluates an optimal solution for SD analysis.

2. Prior approaches

One potential approach to determining OS for SD is an exhaustive enumeration of all the possible sequences and the selection of OS with minimal removals. However, this analysis is computationally expensive (typically exponential) and is not recommended.

Another possible approach is to determine **OS** for **C** from a Complete Disassembly (CD) approach which involves disassembling all the components in **A**. Several representations allow evaluation of CD and assembly sequences: (i) an *Assembly Sequence Diagram* (De-Fazio and Whitney, 1987), which represents the ability or inability to assemble a part to a subassembly; (ii) an *AND/OR Graph* (Homem de Mello and Sanderson, 1991), which establishes conditions and precedence relationships between components; (iii) an *Abstract Liaison Graph* (Lee and Shin, 1990), which represents the stability of part interconnections and the directional constraints of the motions that bring two parts together; (iv) a *Non-Directional Blocking Graph* (Wilson *et al.*, 1995; Jones *et al.*, 1997), which describes part interactions from the blocking nature of components by utilizing the concept of graph partitioning; and (v) a *Constraint Graph* (Beasley and Martin, 1993; Arkin *et al.*, 1989; Mattikalli and Khosla, 1992; Xu *et al.*, 1995; Halperin *et al.*, 1998) which uses computational-geometry techniques to determine the constraints of the assembly for sequencing. Although **S** can be obtained from a CD, it does not give an optimal solution (Srinivasan and Gadh, 2000). Therefore, a separate approach for SD analysis is necessary.

Another approach for SD is the construction of a Disassembly Tree (Woo and Dutta, 1991), which is proposed for CD and single component disassembly. The tree is designed to model the 'Onion Peeling' abstraction: recursively disassembling removable components, starting from the boundary of **A** and proceeding inwards. The Disassembly Tree approach is proposed for 2.5D objects and the analysis is based on the contact geometry. However, the algorithm is only applicable for assemblies in which every component is disassembled by removing none or one of its mating adjacent components. Therefore, the above approach is too restrictive for our use.

Most of the previous work on automated assembly and disassembly planning has focused on CD. However, there has been little investigation of techniques for SD. In view of the prior research, there is still a need for an efficient way to determine SD sequences rather than CD sequences, and more particularly optimal SD sequences. Preferably, any methods for determining such sequences should be computationally efficient, allow analysis of assemblies in up to 3D using input geometric models rather than a continuous stream of user inputs, and should be highly automated in all other respects as well. Furthermore, any such method should allow the determination of SD sequences, which take into account not just the spatial constraints of the components, but also the user-defined constraints.

In Srinivasan and Gadh (1998, 2000), algorithms analyzing the contact-geometry for single and multiple component(s) SD were presented. A new concept, a 'disassembly wave', was introduced that denotes the disassembly ordering of components in order to determine a

sequence with minimum component removals. However, the disassembly ordering was based on contact-geometry analysis, i.e., global interference was not checked and therefore local disassembly sequences were determined.

In continuation of our previous research, this paper analyzes global SD of components accounting both the spatial and user-defined constraints, and presents a new algorithm and implementation results in determining a non-interfering optimal sequence.

3. Automation analysis

This section presents the *Global Selective Disassembly* (GSD) algorithm to determine an **OS** for global SD of a selected component by analyzing the geometric constraints of the components in **A**. First the geometric attributes related to the algorithm are defined, next the GSD algorithm accounting for spatial constraints is presented, and then the modifications to the algorithm to incorporate user-defined constraints are discussed. The proposed algorithm is applicable for both 2D and 3D assembly geometry.

3.1. Geometric attributes for global selective disassembly

The current research assumes a fixed set of disassembly directions, since determining spatial constraints is a time-consuming procedure (Lozano-Perez, 1983). Let **U** denote the universe of disassembly directions for the components in **A** and U_i denote a set of disassembly directions of C_i after accounting for the directional constraints imposed on C_i . For example, in Fig. 2, $U = \{\mathbf{d}_1, \mathbf{d}_2\}$. With no directional constraints applied to any other components $U_i = U$, i.e., $U_1 = U_2 = U_3 = U_4 = U_5 = \{\mathbf{d}_1, \mathbf{d}_2\}$. However if, for example, C_1 is constrained to move and C_2 is constrained along \mathbf{d}_1 , then $U_1 = \{\}$, $U_2 = \{\mathbf{d}_2\}$, $U_3 = U_4 = U_5 = \{\mathbf{d}_1, \mathbf{d}_2\}$.

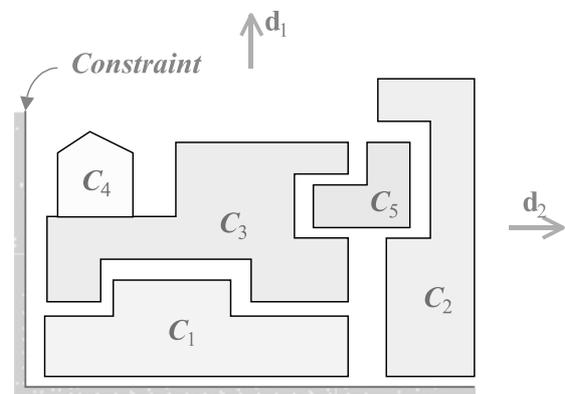


Fig. 2. A test assembly to illustrate the GSD algorithm.

- **Definition 1: (spatial disassemblability):** The spatial disassemblability of C_i , denoted as Δ_i , is TRUE if there exists a disassembly direction $\mathbf{d}_j \in \mathbf{U}_i$ along which C_i does not collide with any other $C_k \in A$ ($k \neq i$). A spatially disassemblable component is denoted as C_b . For example in Fig. 2: $\Delta_2 = \text{TRUE}$ and $\Delta_5 = \text{FALSE}$ for C_2 and C_5 , respectively.
- **Definition 2: (spatial boundary set):** The spatial boundary set, denoted as β_r ($r > 0$), is defined as follows: For $r = 1$, $\beta_r = \text{Set of } C_b\text{'s in } A$. For $r > 1$, $\beta_r = \text{Set of } C_b\text{'s in the assembly with components } A - (\beta_{r-1} + \beta_{r-2} \dots + \beta_1)$. For example, the spatial boundary sets in Fig. 2 are $\beta_1 = \{C_2, C_4\}$, $\beta_2 = \{C_5\}$, $\beta_3 = \{C_3\}$ and $\beta_4 = \{C_1\}$.
- **Definition 3: (spatial adjacent set):** The spatial adjacent set of $C_i \in \beta_r$ (denoted as α_i) is defined as $\alpha_i = (\beta_{r-1} \cup \beta_{r-2} \dots \cup \beta_1)$. For example in Fig. 2, $C_3 \in \beta_3$ and $\alpha_3 = \{C_2, C_4, C_5\}$. Similarly for $C_5 \in \beta_2$ and $\alpha_5 = \{C_2, C_4\}$.
- **Definition 4: (spatial removal influence):** Let C' denote a sub-set of components in A . If in the absence of $C' \subseteq \alpha_i$ in A , $\Delta_i = \text{TRUE}$, then the spatial removal influence of C' on C_i denoted as $RI_i^{C'} = \text{TRUE}$. For example, in Fig. 2, the spatial removal influence of (C_4, C_5) on C_3 is $RI_3^{4,5} = \text{TRUE}$.

3.2. Spatial constraints and sequencing

A Removal influence graph, RG (Srinivasan and Gadh, 1998), is constructed to determine the topological disassembly ordering of components. In RG, the nodes correspond to components in A and arcs correspond to the removal influence between the components. An arc $C_i \rightarrow C'$ in RG, with an attribute \mathbf{d}_j on the directed edge, indicates that C_i is disassemblable along \mathbf{d}_j after removing C' . For example, consider A shown in Fig. 2 with $C = \{C_3\}$ and $\mathbf{U} = \{\mathbf{d}_1, \mathbf{d}_2\}$. The RG is shown in Fig. 3. C_3 is spatially disassemblable along \mathbf{d}_1 after removing C_4 and C_5 ($RI_3^{4,5} = \text{TRUE}$). C_5 is spatially disassemblable

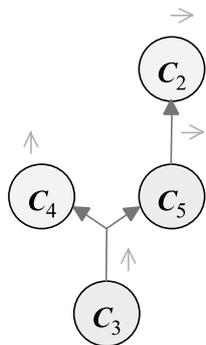


Fig. 3. RG for A in Fig. 2 $C = \{C_3\}$.

along \mathbf{d}_2 after removing C_2 ($RI_5^2 = \text{TRUE}$). Moreover, $\Delta_4 = \text{TRUE}$ and $\Delta_2 = \text{TRUE}$ for C_4 and C_2 , respectively.

A sequence $S = \{C_b \xrightarrow{\mathbf{p}} C_x\}$ ($C_x \in C$) is derivable from RG (where $C_b \xrightarrow{\mathbf{p}} C_x$ denotes a disassembly order from C_b to C_x) via topological sorting of RG: recursively removing nodes in RG with no directed edges until C_x is removed to determine a S . For example, from the RG shown in Fig. 3, a $S = \{C_2, C_4, C_5, C_3\}$ for C_3 . The GSD algorithm is listed below and an illustration of the algorithm is shown in Fig. 4.

Algorithm: GSD (Input: A, C ; Output: S)
 {

- Step 1. Determine the spatial constraints for every $C_i \in A$ along every $\mathbf{d}_j \in \mathbf{U}_i$.
 - Step 2. Determine the spatial boundary sets for A .
 - Step 3. Determine the spatial adjacent set for every $C_i \in A$.
 - Step 4. Construct the RG for C .
 - Step 5. Topologically sort RG and Compute S .
- }

3.3. Incorporation of user-defined constraints in the analysis

Component grouping can be incorporated in the above GSD algorithm by treating the components grouped as a single unit (while assembling/disassembling). For example, in Fig. 2 if C_2 and C_5 are grouped as a subassembly, then the group (C_2, C_5) is constrained along \mathbf{d}_1 by C_3 since C_5 is constrained by C_3 along \mathbf{d}_1 . However (C_2, C_5) is not constrained along \mathbf{d}_2 since C_2 is not constrained along \mathbf{d}_2 and C_5 is constrained only by C_2 along \mathbf{d}_2 . Figure 5 shows the RG for $C = \{C_3\}$ with $S1 = \{C_2, C_5\}$ and $S = \{C_4, S1, C_3\}$.

Geometric constraints such as one that restricts a component to move or one that constraints along one or more disassembly direction(s) can be added to the SD analysis. This can be done through the definition of \mathbf{U}_i for C_i . For example, in Fig. 1, if C_1 is constrained along $-\mathbf{d}_3$ then $\mathbf{U}_1 = \{\mathbf{d}_1, -\mathbf{d}_1, \mathbf{d}_2, -\mathbf{d}_2, \mathbf{d}_3\}$, therefore $RI_1^{1,2} = \text{FALSE}$ along $-\mathbf{d}_3$. Similarly, in Fig. 2, if C_5 is constrained to move then $\mathbf{U}_5 = \{\}$ and therefore $\Delta_5 = \text{FALSE}$. Therefore, $\beta_1 = \{C_2, C_4\}$, $\beta_2 = \{\}$, and C_3 cannot be disassembled.

3.4. Discussion

The sequence determined by the GSD algorithm is optimal, and is discussed as follows: Let n_r be the number of component removals in S and n_s be the number of simultaneous removals in S . The attribute Δ_x for C_x determines whether C_x is globally disassemblable by evaluating the spatial constraints of the components. If $C_x \in \beta_r$ then C_x could have not been disassemblable at β_p

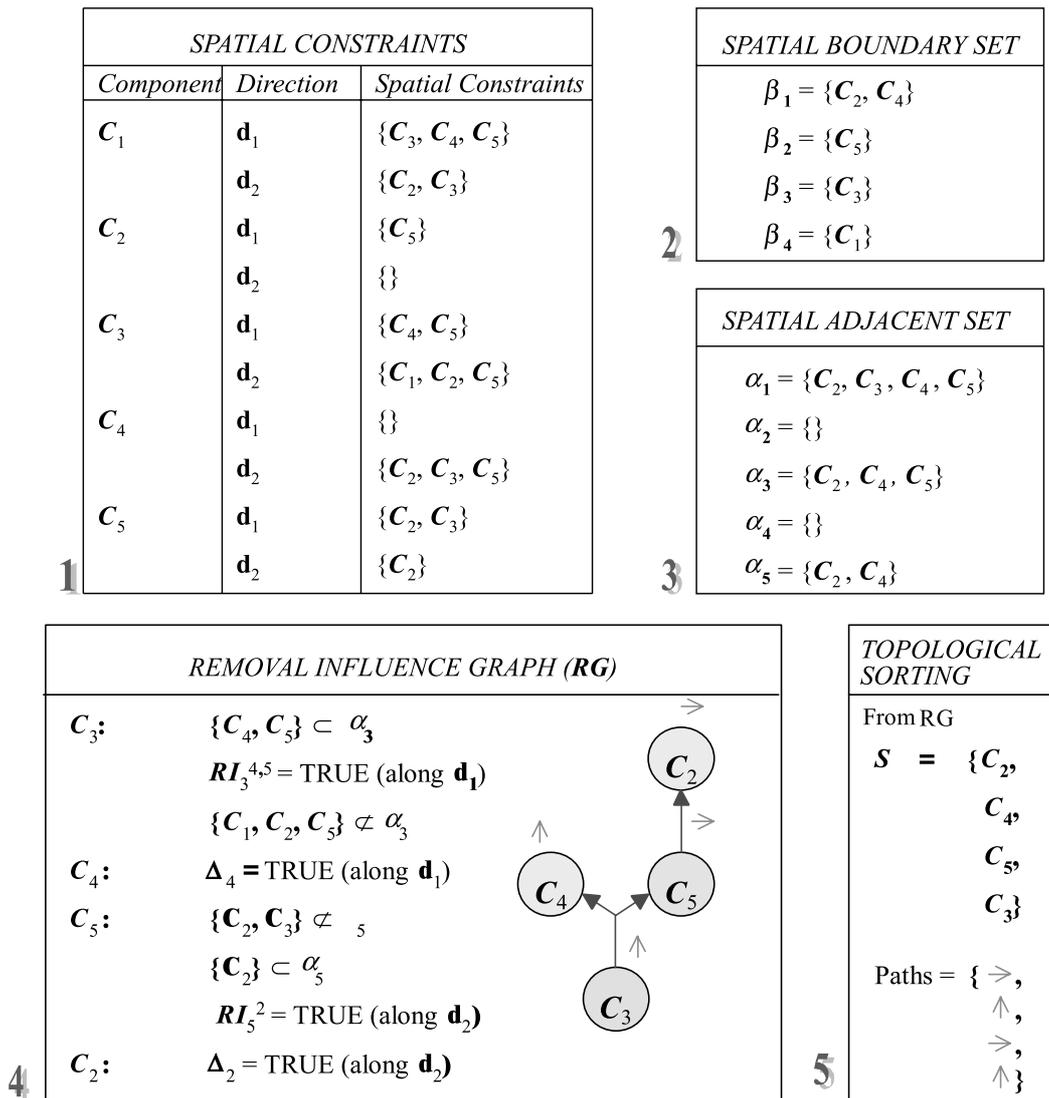


Fig. 4. Illustration of global selective disassembly algorithm for A in Fig. 2, $C = \{C_3\}$.

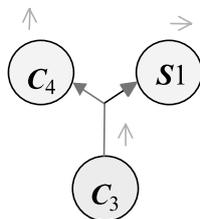


Fig. 5. RG for $C = \{C_3\}$; A in Fig. 2.

($p < r$) since, the disassemblability of $C_i \in A$ is evaluated for all of the disassembly directions in U_i . Moreover, in evaluating the spatial removal influence of components for $C_i \in A$, only $C' \subseteq \alpha_i$ are disassembled. Therefore for $C_x \in \beta_r$, $n_s = r$ for S determined by the algorithm and the sequence evaluated is an OS .

In addition, the spatial removal influence computation further refines the solution set for C_x by ensuring that all

$C_i \in \alpha_x$ does not have to be disassembled for C_x and n_r for S is locally minimum among the sequences that can be determined for $C_x \in \beta_r$. In the example shown in Fig. 2 with $C = \{C_3\}$, an $OS = \{C_2, C_4, C_5, C_3\}$, $n_r = 4$ and $n_s = 3$ since C_2 and C_4 can be disassembled simultaneously. Similarly for A in Fig. 1, with $U = \{d_1, -d_1, d_2, -d_2, d_3, -d_3\}$ and $C = \{C_3\}$, the algorithm determines $OS_1 = \{C_1, C_2, C_3\}$ and $OS_2 = \{C_5, C_4, C_3\}$ with $n_r = n_s = 3$.

The computational complexity of the GSD algorithm to determine an OS for $C_x \in C$ is $O(d \times F^2 + d \times n^3)$, i.e., polynomial; where n is the number of components in A , $d = \text{Cardinality}(U)$ and $F = \text{total number of faces of all the components in } A$.

The GSD algorithm can be applied for both SD and CD. For SD of $s (\geq 1)$ components, the algorithm is applied to every $C_x \in C$ to obtain a combined RG. The topological sorting of the RG obtains an OS to disassemble the s

target components. However, for $s = n$ target components from A , a non-interfering CD sequence with minimal number of simultaneous removals is determined.

4. Implementation results and discussion

The GSD algorithm is implemented and tested in the prototypical system *Assembly and Disassembly in Three-Dimensions* (A3D), developed at the University of Wisconsin-Madison for assembly/disassembly analysis. This section describes the A3D implementation system and presents some of the preliminary implementation results and discussions on the GSD algorithm.

4.1. Implementation: A3D system

The A3D system (a snapshot of the interface is shown in Fig. 6) uses C++ as a programming language, OpenGL™ as a graphics library, WorldToolKit™ as a development library and ACIS™/PARASOLID™ for geometric computations. A3D is compatible to both Unix™ and Windows-NT™ operating systems. The A3D system reads in assembly models created in conventional CAD systems such as ProEngineer™ and UniGraphics™,

allows the user to select components for SD, group/un-group components and add/remove directional constraints, and performs the automated SD analysis.

4.2. Spatial constraints and sequence determination

A $C_i \in A$ may constrain $C_j \in A$ both by contact and also spatially. For example, in Fig. 7a, ol33 is constrained by ol3 (both by contact and spatially). Therefore, to ensure global disassembly, even between two mating components, analyzing the contact-geometry alone is not sufficient and spatial analysis is necessary. The GSD algorithm analyzes the spatial constraints imposed by both contact and non-contact geometry, as illustrated by the following result.

Figure 7a shows the CAD model of an Augmentor subassembly and an $OS = \{ol11, ol1, ol33, ol22, s1, ol3, ol2, s21\}$ generated for $C = \{s21\}$. Figure 7b shows the corresponding RG generated by the GSD algorithm. The components in β_1 are disassembled simultaneously followed by the components in β_2 and β_3 . Therefore, the simultaneous disassembly groups are (ol33, ol22, ol11, ol1), (ol3, ol2, s1) and (s21) with $n_r = 8$ and $n_s = 3$. The SD sequence can be reversed and used for selective assembling of the components. For example, the simulta-

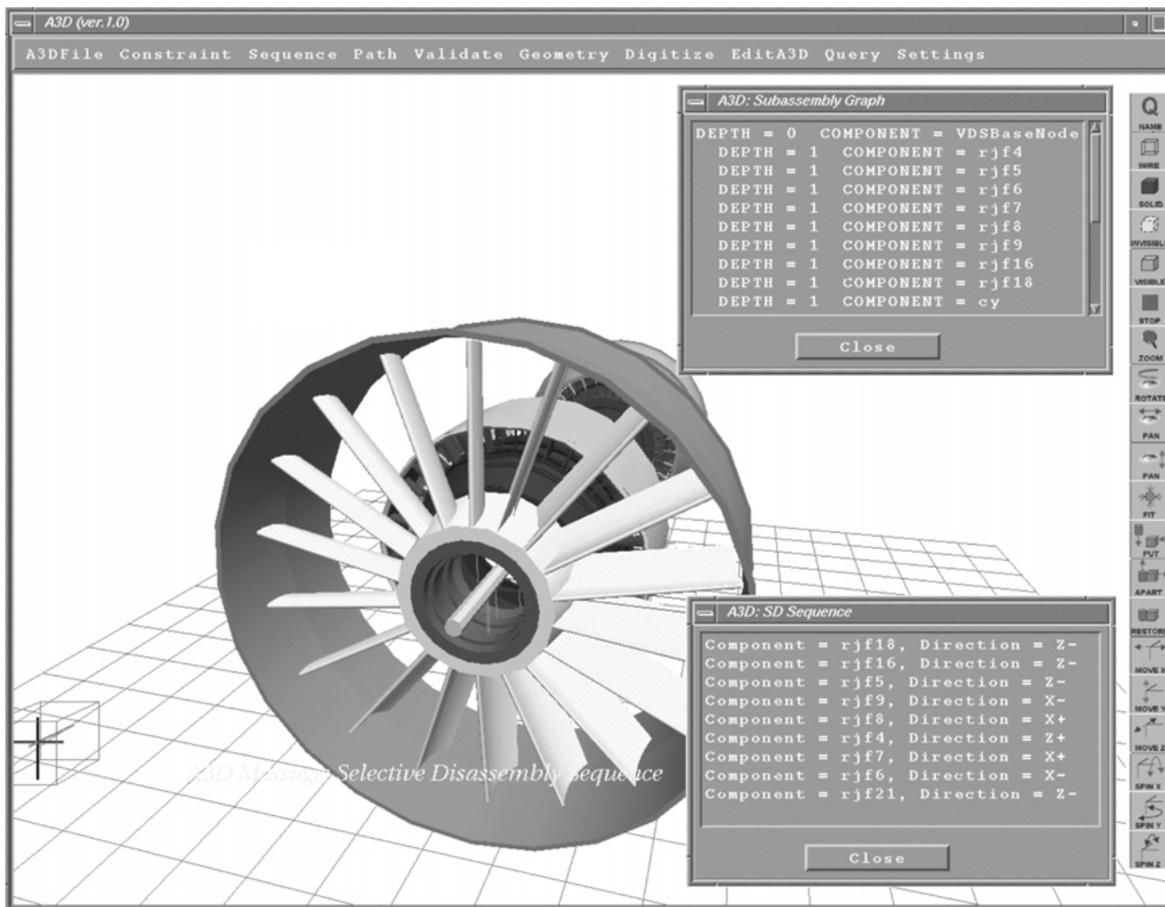
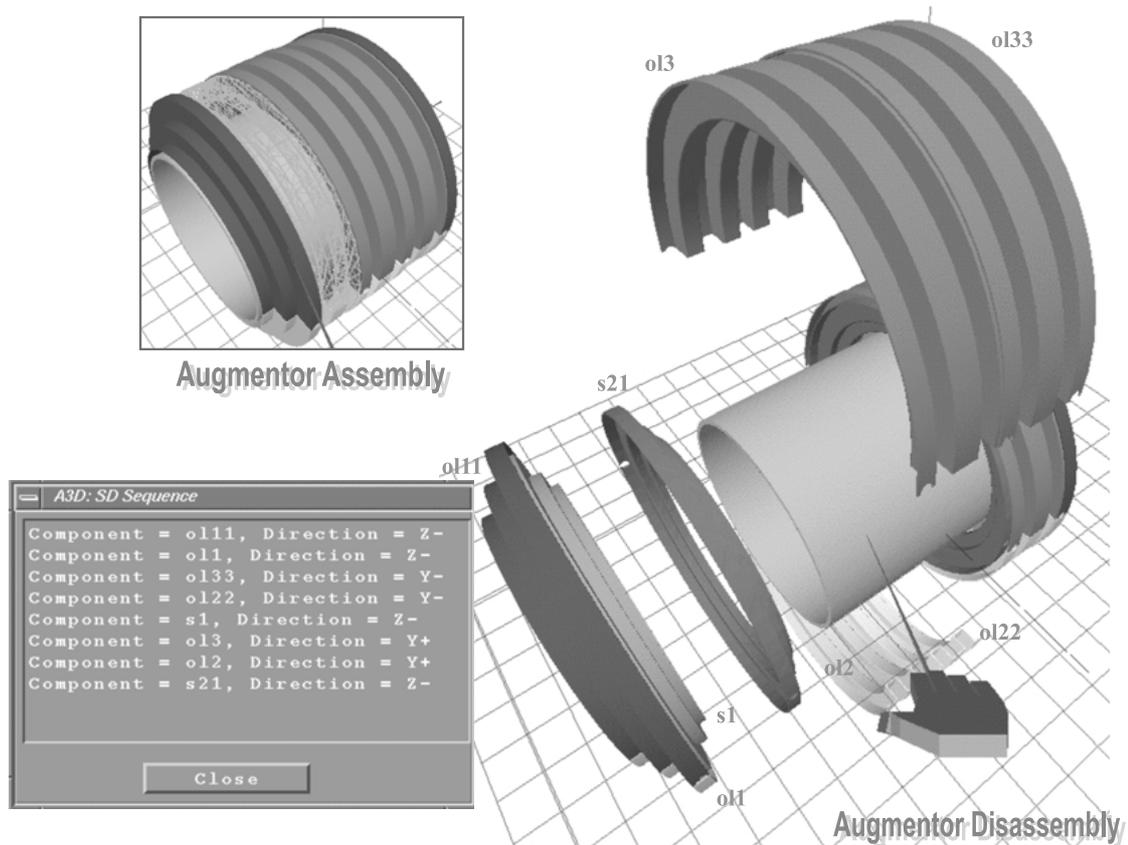


Fig. 6. A3D: assembly and disassembly in three dimensions.

(a)



(b)

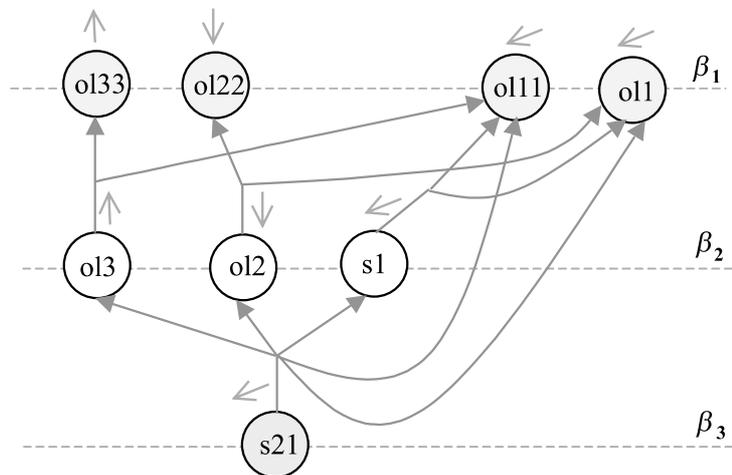


Fig. 7. (a) Augmentor assembly; $C = \{s21\}$, $OS = \{ol11, ol1, ol33, ol22, s1, ol3, ol2, s21\}$; and (b) the RG for $C = \{s21\}$.

neous assembly groups of outer-liners and stiffeners for the Augmentor assembly based on the above sequence are $(s21)$, $(ol2, ol3, s1)$ and $(ol22, ol33, ol1, ol11)$.

4.3. Grouping components and sequence determination

Subassemblies (group of components) may change S and disassembly directions for components. The current re-

search determines S for subassemblies by evaluating it as a single unit (single node in RG).

For example, Fig. 8a shows the CAD model of an Aero-engine subassembly and an $OS = \{rjf18, rjf16, rjf4, rjf5, rjf9, rjf8, rjf7, rjf6, rjf21\}$, generated for $C = \{rjf21\}$ with no components grouped. Figure 8b shows the corresponding RG. The simultaneous component removal groups are $(rjf18, rjf16, rjf4)$, $(rjf5)$, $(rjf9, rjf8, rjf7, rjf6)$ and $(rjf21)$

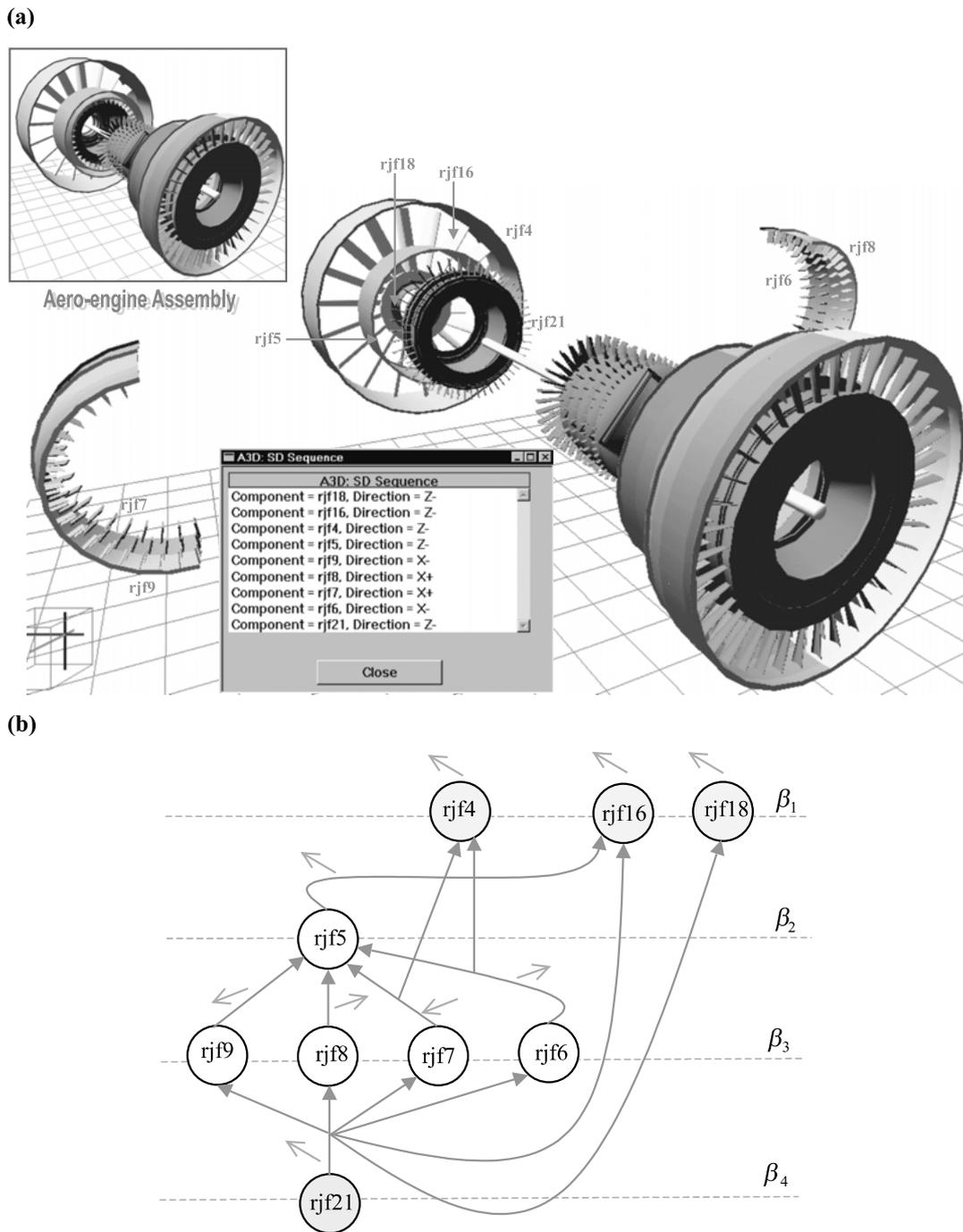


Fig. 8. (a) Aero-Engine; $C = \{rjf21\}$, $OS = \{rjf18, rjf16, rjf4, rjf5, rjf9, rjf8, rjf7, rjf6, rjf21\}$; and (b) RG for $C = \{rjf21\}$.

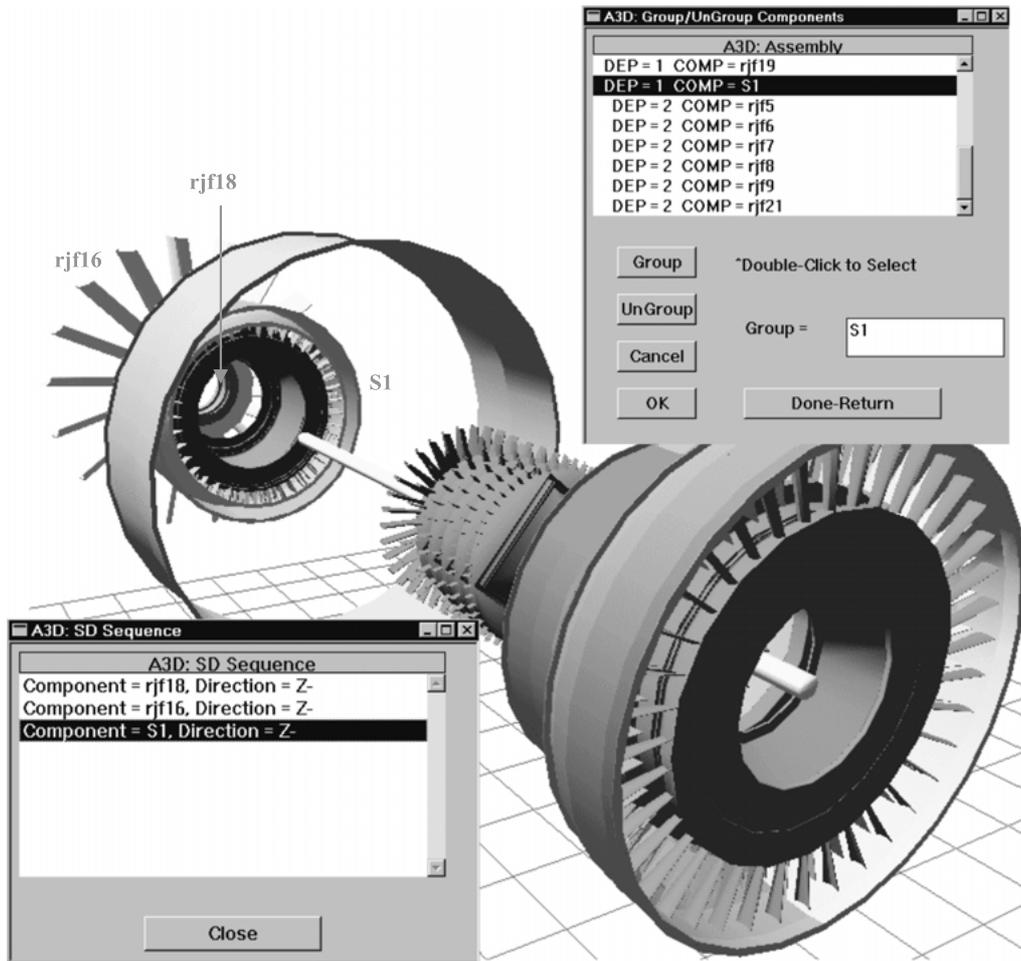
with $n_r = 9$ and $n_s = 4$. However, with components grouped as a subassembly $S1 = \{rjf5, rjf6, rjf7, rjf8, rjf9, rjf21\}$, Figure 9a shows an $OS = \{rjf18, rjf16, S1\}$ generated for $C = \{S1\}$. Figure 9b shows the corresponding RG. The simultaneous component removal groups are $(rjf18, rjf16)$ and $(S1)$ with $n_r = 3$ and $n_s = 2$. The specification of the subassemblies can be done either in the CAD system where the models were created or inside the A3D system using the interface provided (the user-interface for defining subassemblies is also shown in Fig. 9a).

4.4. Directional constraints and sequence determination

The addition of directional constraints reduces an OS to lie in a smaller solution space. Moreover, the constraints may change S and disassembly directions of components.

For example, Fig. 10a shows the CAD model of an aero-engine subassembly and $OS = \{rjf11, rjf10, cy, rjf20\}$ generated for $C = \{rjf20\}$ without any directional constraints applied to any components. Figure 10b shows the corresponding RG. The simultaneous component

(a)



(b)

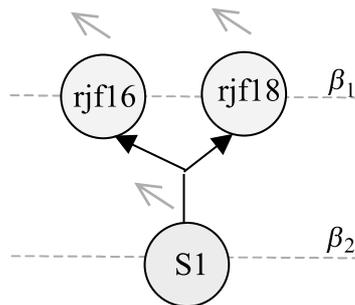


Fig. 9. (a) Aero-engine; $C = \{s1\}$, $OS = \{rjf18, rjf16, s1\}$; and (b) the RG for $C = \{s1\}$.

removal groups are (rjf11, rjf10, cy) and (rjf20) with $n_r = 4$ and $n_s = 2$.

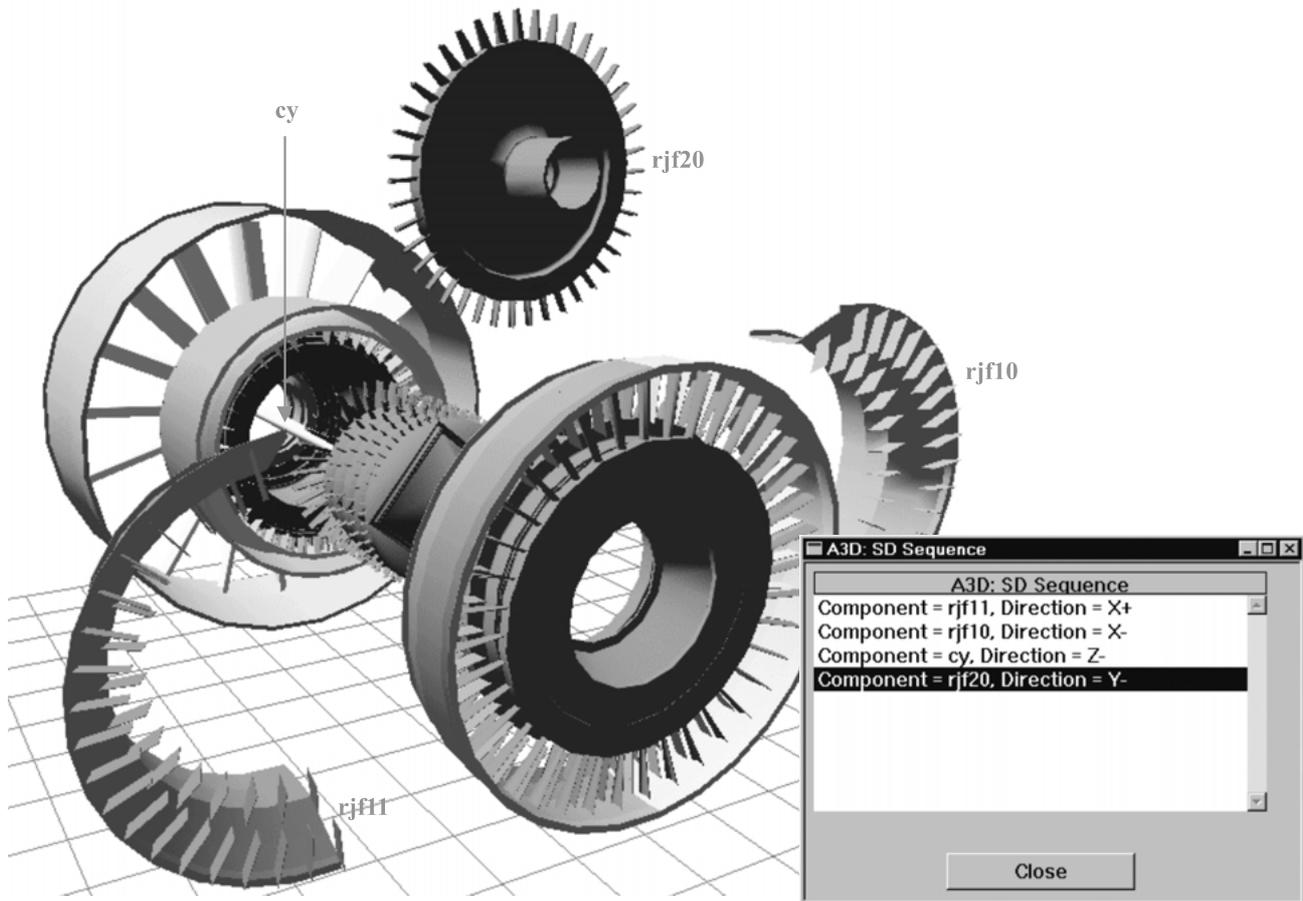
However, if the user constrains cy from moving, then the resultant $OS = \{rjf13, rjf12, rjf11, rjf10, rjf22, rjf20\}$ generated for $C = \{rjf20\}$, as shown in Fig. 11a. Fig. 11b shows the corresponding RG. The simultaneous component removal groups are (rjf13, rjf12, rjf11, rjf10), (rjf22) and (rjf20) with $n_r = 6$ and $n_s = 3$. The user-interface to

add/remove directional constraints in the A3D system is also shown in Fig. 11a.

4.5. Summary of results and discussion

Some of the preliminary results of the GSD algorithm are summarized in Table 1. Applications for the GSD algorithm include maintenance and assembling (Srinivasan

(a)



(b)

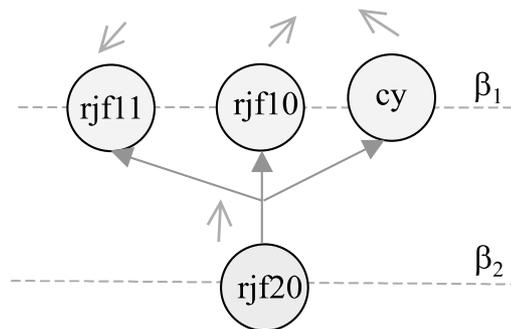


Fig. 10. (a) Aero-engine; $C = \{rjf20\}$, $OS = \{rjf11, rjf10, cy, rjf20\}$; and (b) the RG for $C = \{rjf20\}$.

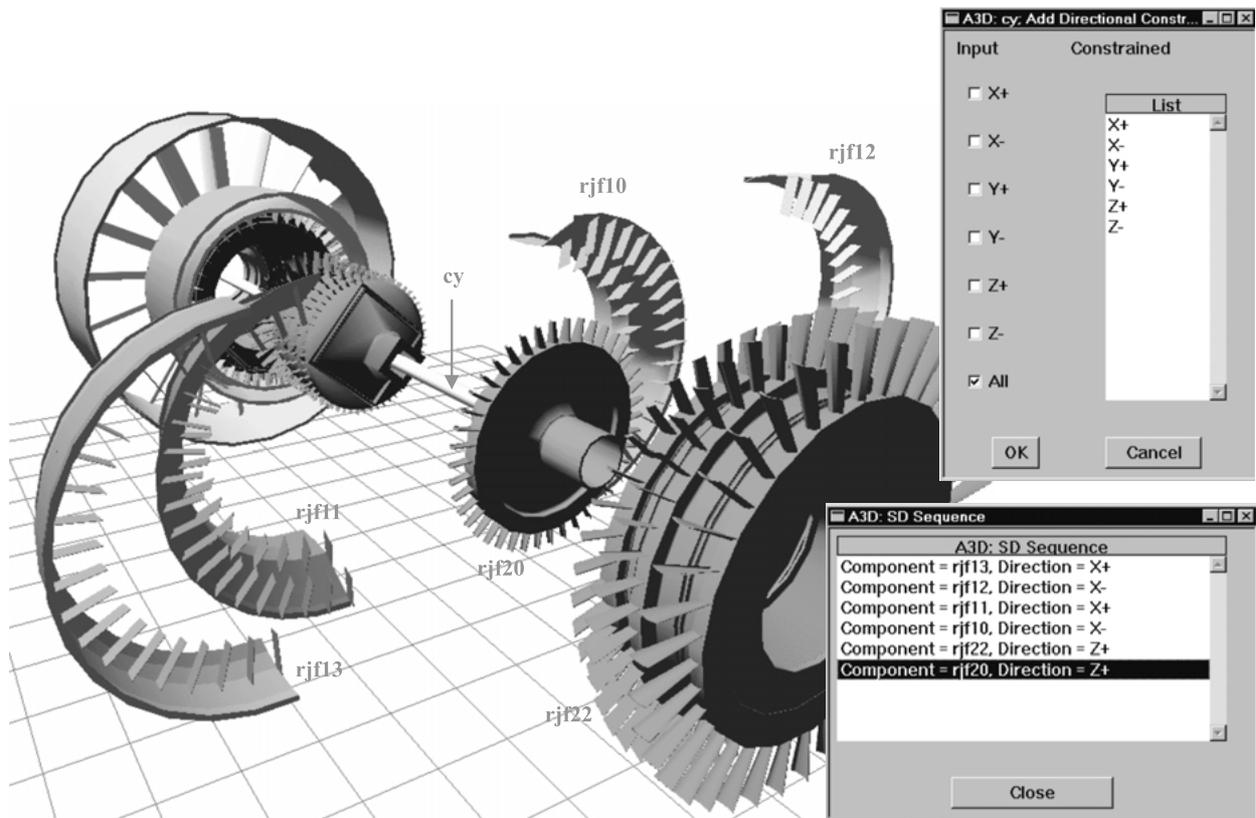
et al., 1999). However, while the GSD algorithm automatically determines a linear path for disassembly/assembly sometimes complex paths may be required for efficient assembling and disassembling.

One potential solution to incorporate complex paths is to have the universe of disassembly directions to include complex motion/paths. Another solution that is being researched is to allow the user to modify/edit the gener-

ated sequence and paths and validate the resulting solution. For example, Fig. 12 shows the result of a sequence editing procedure performed on a crankshaft assembly. Component 4 is edited to disassemble along direction Y^- . However, as Y^- for component 4 results in a collision, an error is reported as a feedback text (shown in Fig. 12).

Performing SD analysis based on geometry determines whether a solution is geometrically feasible. However,

(a)



(b)

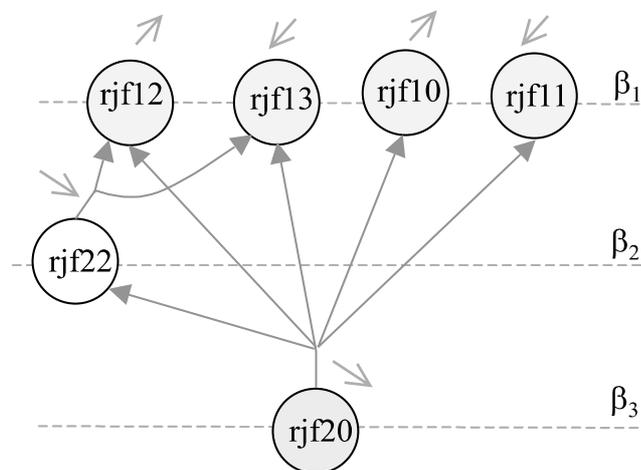


Fig. 11. (a) $C = \{rjf20\}$, cy is constrained, $OS = \{rjf13, rjf12, rjf11, rjf10, rjf22, rjf20\}$; (b) the RG for $C = \{rjf20\}$.

incorporating application-domain knowledge in SD analysis has to be researched. For example, in specifying subassemblies while disassembling, modeling fixture elements as constraints and disassembly precedence relation of components. Moreover, while the current research performs SD analysis with an objective of minimal simultaneous removals, other objectives such as minimal cost, motions, etc. need to be researched.

4.7. Contributions of the current research

The main contributions of this paper are:

1. A new *Global Selective Disassembly* algorithm for automatic determination of a non-interfering sequence for a selected component by analyzing the spatial constraints of the components in the assembly.

Table 1. Results for the GSD algorithm; optimal sequences

A	n	C	Sequence			
			S	n _r	n _s	Optimal
Figure 7a	15	s21	{ol11, ol1, ol33, ol22, s1, ol3, ol2, s21}	8	3	Yes
Figure 8a	17	rjf21	{rjf18, rjf16, rjf4, rjf5, rjf9, rjf8, rjf7, rjf6, rjf21}	9	4	Yes
Figure 9a	17	S1	{rjf18, rjf16, S1}	3	2	Yes
Figure 10a	17	rjf20	{rjf11, rjf10, cy, rjf20}	4	2	Yes
Figure 11a	17	rjf20	{rjf13, rjf12, rjf11, rjf10, rjf22, rjf20}	6	3	Yes

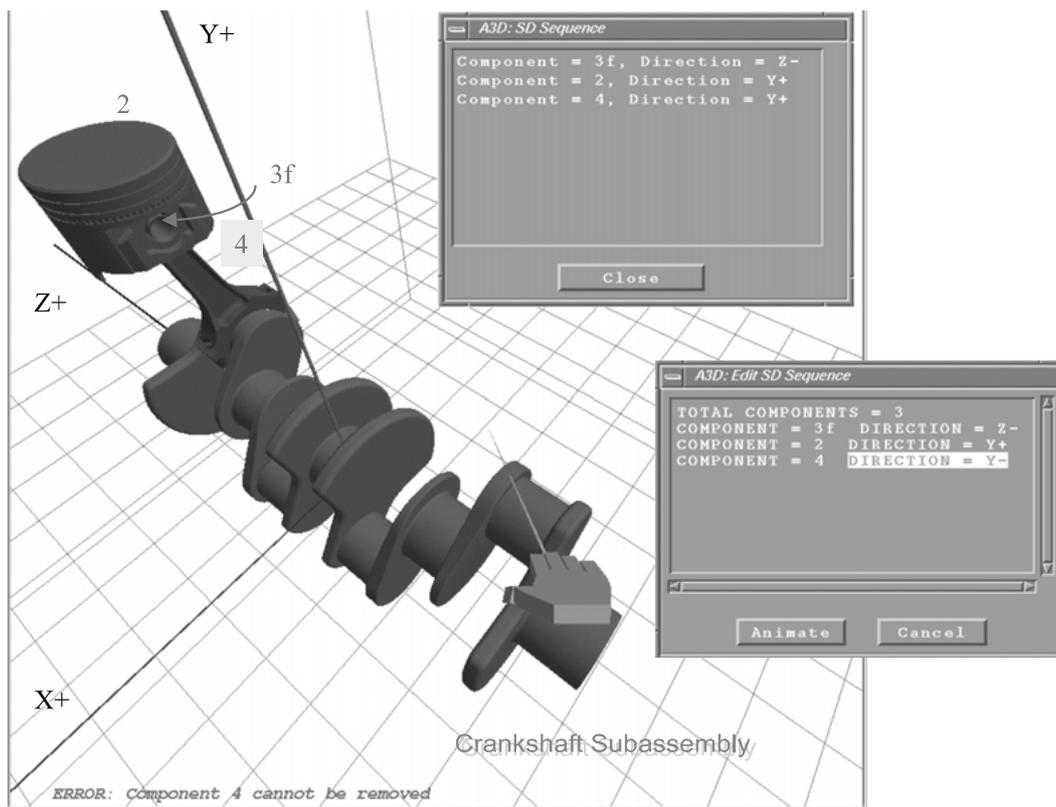


Fig. 12. Crankshaft subassembly; sequence, editing and validation.

2. Incorporation of component grouping constraints and directional constraints analysis along with the spatial constraint analysis in evaluating an optimal sequence.
3. Implementation of the *Global Selective Disassembly* algorithm and the application of the algorithm for maintenance and assembling applications.

5. Summary

This paper analyzes the problem of global disassembly of a selected component by evaluating the spatial and user-defined constraints imposed on the geometry of the

components. A new algorithm to automatically determine a non-interfering disassembly sequence with minimum simultaneous removals with polynomial computational complexity is presented. Some preliminary implementation results of the algorithm and its applicability to maintenance and assembling are discussed.

Acknowledgments

We thank Pratt and Whitney for providing us some CAD test-models for selective disassembly algorithm evaluation. This work was funded in part by the following grants: NSF research equipment grants (DMI 9622665) and CenCITT grant (MTU 960238Z1).

References

- Anderson, D.M. (1990) *Design for Manufacturability—Optimizing Cost, Quality and Time-to-Market*, CIM Press, Lafayette, CA.
- Arkin, E.M., Connelly, R. and Mitchell, J.S.B. (1989) On monotone paths among obstacles, with applications to planning assemblies, in *Proceedings of the ACM Symposium on Computational Geometry*, ACM Press, Saabrucken, Germany, pp. 334–343.
- Beasley, D. and Martin, R.R. (1993) Disassembly sequences for objects built from unit cubes. *Computer Aided Design*, **25**(12), 751–761.
- De-Fazio, T.L. and Whitney, D.E. (1987) Simplified generation of all mechanical assembly sequences. *IEEE Transactions on Robotics and Automation*, **3**(6), 640–658.
- Goldwasser, M., Latombe, J.C. and Motwani, R. (1996) Complexity measures for assembly sequences, in *Proceedings of the International Conference on Robotics and Automation*, IEEE Press, Minneapolis, MN, pp. 1581–1587.
- Halperin, D., Latombe, J.C. and Wilson, R.H. (1998) A general framework for assembly planning—the motion space approach. *Proceedings of the ACM Comp. Geometry*, ACM Press, Minneapolis, MN, 9–15.
- Homem de Mello, L.S. and Lee, S. (1991) *Computer Aided Mechanical Assembly Planning*, Kluwer Academic Publishers, Boston, MA.
- Homem de Mello, L.S. and Sanderson, A.C. (1991) Representations of mechanical assembly sequences. *IEEE Transactions on Robotics and Automation*, **7**(2), 211–227.
- Jones, R.E., Wilson, R.H. and Calton, T.L. (1997) Constraint based interactive assembly planning, in *Proceedings of the International Conference on Robotics and Automation*, IEEE Press, Albuquerque, NM, pp. 913–920.
- Kaufmann, S.G., Wilson, R.H., Jones, R.E., Calton, T.L. and Ames, A.L. (1996) The Archimedes 2 mechanical assembly planning system, in *Proceedings of the International Conference on Robotics and Automation*, IEEE Press, Minneapolis, MN, pp. 3361–3368.
- Lee, S. and Shin, Y.G. (1990) Assembly planning based on geometric reasoning. *Computation and Graphics*, **14**(2), 237–250.
- Lozano-Perez, T. (1983) Spatial planning: a configuration space approach. *IEEE Transactions on Computers*, **32**(2), 108–120.
- Mattikalli, R.S. and Khosla, P. (1992) Motion constraints from contact geometry: representation and analysis, in *Proceedings of the International Conference on Robotics and Automation*, IEEE Press, Nice, France, pp. 2178–2185.
- Srinivasan, H., Figueroa, R. and Gadh, R. (1999) Selective disassembly for virtual prototyping as applied to de-manufacturing. *Journal of Robotics and Computer Integrated Manufacturing*, **15**(3), 231–245.
- Srinivasan, H. and Gadh, R. (1998) A geometric algorithm for single selective disassembly using the wave propagation abstraction. *Computer Aided Design*, **30**(8), 603–613.
- Srinivasan, H. and Gadh, R. (2000) Efficient disassembly of multiple components from an assembly using wave propagation. *ASME Journal of Mechanical Design*, **122**(2), 179–184.
- Srinivasan, H., Shyamsundar, N. and Gadh, R. (1997) A framework for virtual disassembly analysis. *Journal of Intelligent Manufacturing*, **8**(4), 277–295.
- Wilson, R.H., Kavradi, L., Lozano-Perez, T. and Latombe, J.C. (1995) Two-handed assembly sequencing. *International Journal of Robotics Research*, **14**(4), 335–350.
- Woo, T.C. and Dutta, D. (1991) Automatic disassembly and total ordering in three dimensions. *ASME Journal of Engineering for Industry*, **113**(1), 207–213.
- Xu, Y., Mattikalli, R. and Khosla, P. (1995) Generation of partial medial axis for disassembly motion planning. *Journal of Design and Manufacturing*, **5**(2), 89–102.

Biographies

Hari Srinivasan is currently a research engineer at the United Technologies Corporation. He has 8 years of work experience, working for different companies, in the areas of CAD, geometric modeling, virtual prototyping, Internet, virtual reality, design automation and knowledge engineering. He received his Ph.D. from University of Wisconsin-Madison researching in the area of virtual assembly and disassembly. He received his Bachelor's degree in Mechanical Engineering from Anna University, Madras, India and Master's degree from Indian Institute of Science (IISc, Bangalore), India, researching in feature extraction and virtual prototyping. He has authored 20+ technical publications and has also filed a Patent for a new software system developed. He has delivered 30+ industrial seminars and reviewed 30+ international journal/conference/book publications. He served as a session co-chair (1999–2001) and review co-chair (2000–2001) for the ASME International Design Conferences. He has received several awards and certificates, including the University Gold Medal for Overall Excellency, Anna University, Madras.

Rajit Gadh is a Professor at the University of Wisconsin-Madison. His research interest is virtual design/prototyping. He received his B.S. from Indian Institute of Technology, Kanpur, M.S. from Cornell University and Ph.D. from Carnegie Mellon University. He has been a visiting researcher at University of California, Berkeley and has spent roughly 5 years in industry working for companies such as Ford, GM and DEC. Professor Rajit Gadh's research interests are in the modeling of fundamentally new computer-aided design approaches. The first theme, virtual shape design involves development of a fundamentally new interface and geometric representation to support designing shapes within a virtual reality system. The second theme, shape abstraction definition and extraction to support design, involves determination of shape features from component shapes and disassembly of assemblies in the context of computer-aided design. He received the 1996 AT&T–Lucent, Industrial Ecology Fellowship, the NSF Career Development Award and Grant in 1995, in 1994 the Engineering Foundation, Research Initiation Award, and the Ralph Teetor Research and Educational Award, from SAE in 1993.

Contributed by the Manufacturing Process Planning Department