Product Redesign for Robotic Assembly

Wynne Hsu, Lecturer
Department of Information Systems and Computer Science
National University of Singapore
Singapore 119260
whsu@iscs.nus.sg

Andrew Lim, Senior Technical Staff
Information Technology Institute
National Computer Board
Singapore
alim@iti.gov.sg

C. S. G. Lee, Professor
Electrical Engineering
Purdue University
West Lafayette, Indiana
U.S.A
csglee@ecn.purdue.edu

ABSTRACT

It is an established fact that majority of a product's cost is determined by its design. Hence, effort should be directed to achieve a lower cost design without sacrificing the original functionality. One way to achieve this is to reduce the cost of assembly (more than 50% of the total cost of product) by redesigning the product so that it is better for assembly. A computer-aided product redesign system is proposed to provide help in generating assembly-oriented redesign by taking the robotic assembly constraints into consideration. The system operates in two phases. In the first phase, an objective evaluation of the input design is performed to determine whether there is a need for redesign. During the second phase, the focus is to aid the designer in searching for feasible design alternatives. Two quantitative measures have been proposed that will evaluate the conformity of an input design to the design for robotic assembly guidelines. After the analysis, redesign suggestions are generated to improve the design from the perspective of robotic assembly. The redesign suggestions are classified into: global redesign suggestions (changes involve two or more components) and local redesign suggestions (changes involve only one component at a time). A prototype system written in C++ with X11/Motif interface was developed on the Sun Sparcstation with graphic interface to an Iris workstation. A real-life example, the tilt-mechanism example, is used as an illustrative example.

[0] This work was supported in part by the National University of Singapore Research Project Grant RP940644

1 Introduction

The aim of manufacturing industry is to create products which sell profitably. To achieve this aim, a product must constantly evolve to meet the changing needs. In this paper, a systematic approach to design evolution is proposed. The main motivation for this approach comes from the realization that a small difference in the design can have a major effect downstream. A chamfer can ease assembly, and a slight repositioning of a screw can greatly enhance reliability. Boothyrod, recognizing the significant impact a design activity has on the downstream activities(assembly), proposed a quantitative design-for-assembly approach in the 1980s [5]. His approach has been adopted in industry to this day and provides a good source of feedback to the designer concerning the assemblability of his/her design. At about the same time, another group of proponents proposed a more qualitative approach to help the designer. In this approach, high-level design principles are established which serve as guidelines to the designer on how to produce a more assembly-/manufacturing-oriented design [1]. Recently, the concurrent engineering advocates [17] also favor using a multi-disciplinary team during the design phase so as to bring later life-cycle issues into consideration in the design phase. These approaches all focus on the derivation of an 'optimal' initial design solution. However, in real-life situation, no design is perfect in the first attempt. Redesign is, and always will be, an essential part of a product's life-cycle. In this paper, we will like to discuss a systematic redesign approach which integrates the assembly phase and the design phase with the goal of providing dynamic, product-specific redesign suggestions to the designer. The fundamental issues addressed in our redesign approach are twofold: 1) to provide objective evaluation functions that will measure the robotic assemblability of a design, and 2) to provide suggestions on how the product should be redesigned so that it is better for assembly. The goal of this research is not to replace human designers but rather, to provide redesign support to the human designers in the form of two important tools: an evaluation tool and a redesign suggestion tool. The evaluation tool allows the designer to evaluate his/her design based on some objective, mathematically-based evaluation criteria. By providing such objective evaluation criteria, an unbiased comparison between different design alternatives is made possible. The redesign suggestion tool aids the designer by performing analysis on large amount of data (generated automatically from the computer-aided design tools and computer-aided assembly planners) and summarizing the results of analysis to the designer. In addition, a collective database of previous design knowledge is kept to aid in the search for feasible redesign alternatives.

Outline of this paper follows. Section 2 gives some background information and related work. The formalism of the proposed evaluation functions are given in Section 3. In Section 4, the process of generating suitable redesign suggestions is described. An illustrative example, the tilt-mechanism example, is given in Section 5. Finally, Section 6 summarizes the conclusions.

2 Related Work

Much work has been done in the area of design, ranging from the axiomatic approach proposed by Suh [24] to the application-oriented research such as case study of successful designs [9]. Many researchers have also proposed different design approaches to give better understanding of the design process itself. Design synthetic reasoning was proposed by Kannapan and Marshek to derive the structure of a product from functional specifications. Their focus is in deriving a conceptual design from specifications. On the other hand, Jack Mostow [16] focused on the issues of design by derivational analogy. Various issues involved in design such as adapting the old design to fit a new design and partial reuse of an old design were discussed. Functional design [3] emphasizes the role of functional specification to aid in the selection of engineering features. It is a process of function-to-feature matching which forms the basis of a part taxonomy. In Kota and Ward's work [13], an interesting view was presented on the relationship among functions, structures and constraints in a conceptual design. The study by Watton and Rinderle [26] provided an interesting approach to enhance a given design. Their approach is to find alternative parameters to give a more direct correspondence to functional behavior.

Beside looking at design/redesign from a functional perspective, researchers also began to look at redesign from the assembly or manufacturing perspective. A wealth of research has been developed on design for assembly [5, 1, 15, 10, 4], design for manufacturing [8, 19] and concurrent engineering [17, 23, 6, 25]. Recently, design for disassembly approach [18] has also been proposed which aims to derive designs that are easily decomposable. This helps to increase the feasibility of reusing old components and enable easy recycling. In addition, with the introduction of robotic assembly, substantial changes are required in the assembly methods, assembly tools, assembly system designs and product design. Rathmill [22], Rampersad [21] investigated various issues specific to robotic assembly.

3 Quantitative Design Measures for Robotic Assembly

There exists many high-level design guidelines for robotic assembly [22, 21]. Based on these design guidelines for robotic assembly, the authors have defined two quantitative measures to evaluate a design's conformity to these design guidelines. The two measures are: (1) degree of standardization, and (2) degree of uniformity.

3.1 Degree of Standardization

In robotic assembly, the degree of standardization of mating features plays an important role in determining the success or failure of an assembly task. A mating feature is considered standard if it conforms to the design principles of: maximize part symmetry; improve assembly access; maximize part compliance; and provide parts with integral self-locking features. A design that uses standard mating features has a higher chance of being assembled correctly.

To measure the degree of standardization of the mating features in a given design, we have adopted a stored-and-compared approach. A group of human designers are asked to come up with a list of mating features that are considered standard. This list is translated into an internal representation and stored in a generic library of standard mating features. A comparison procedure is then invoked to find the best match between the mating features of a given design and the standard mating features in the library. A high degree of match indicates a high degree of standardization.

The first issue to tackle using this approach is: what is the internal representation for storing the list of standard mating features? This internal representation must be flexible enough to allow storing of generic mating feature class characteristics; yet it must be specific enough to make comparison among different mating feature classes possible. A parametric constructive solid geometry representation is proposed. In the parametric CSG representation, the orientation matrix and the parameters of the primitive solids are variables. Example of a parametric representation is shown in Figure 1. The parametric CSG representation can then be instantiated to become a normal CSG tree representation. Figure 2 shows the results of different instantiation of the same parametric CSG representation.

With this parametric CSG representation, the computation of the similarity between two mating features can be carried out as follows. First, a graph matching

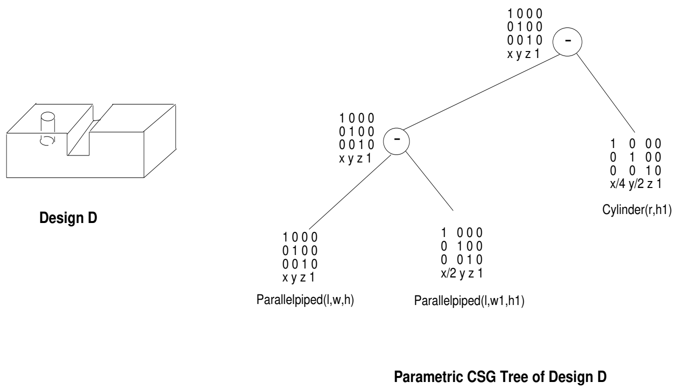

Figure 1: The CSG Tree Example

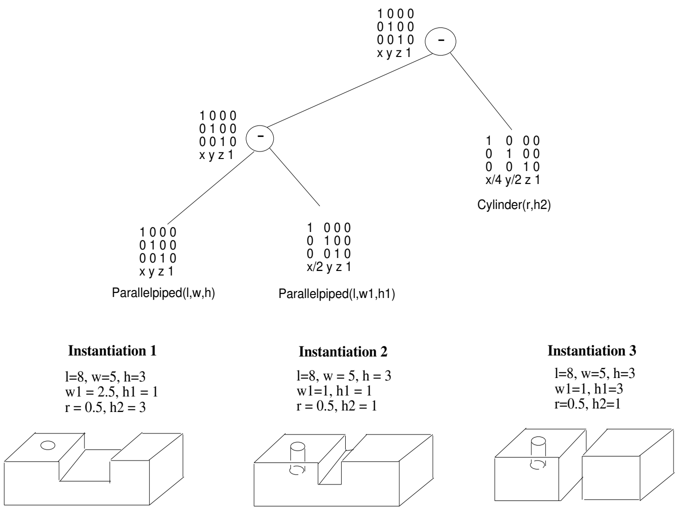

Figure 2: The Instantiations of a Parametric CSG Tree

algorithm is used to find all standard mating features whose graph structure is the same as the mating feature to be compared. A unification process is then carried out to instantiate the variables to values. The instantiated CSG tree represents a solid. Standardization measure is then computed based on the similarity between the two solids.

Two solids are considered similar if the ratio of intersection volume to the union volume is close to 1. Let $S1$ denote the instantiated CSG description of the mating feature f. Let $S2_i$ denotes the instantiated CSG description of the ith standard mating feature in the generic library L. The geometric similarity measure of mating feature f, gs(f), is

$$gs(f) = \max_i\left(\frac{volume(S1 * S2_i)}{volume(S1 + S2_i)}\right)$$

where $S1 * S2_i$ denotes the intersection between $S1$ and $S2_i$, and $S1 + S2_i$ denotes the union between $S1$ and $S2_i$. gs(f) takes values between 0 and 1. If gs(f) is close to 1, it indicates f is very similar to a standard mating feature. On the other hand, if gs(f) is close to 0, then perhaps a redesign is needed to transform f into a standard mating feature. The standard mating feature that is geometrically the closest to f is denoted by f^*.

For a given design with N mating features, the degree of standardization is

$$\text{Degree of Standardization} = \frac{1}{N}\sum_{i=1}^{N} gs(f_i)$$

3.2 Degree of Uniformity

Uniformity implies a same and unchanging set of conditions. This is particular important in the context of robotic assembly where the changes in the direction of assembly and the numbers of assembly tools used should be kept to a minimum. Such a design takes less time to assemble, has a higher chance of being assembled successfully, and hence better for assembly.

This measure focuses on the impact of a design on the assembly process. With the advancement in automatic assembly plan generator, feasible assembly plans can now be generated automatically [7, 12, 14, 20]. An assembly plan is represented as a partial order graph $G = (V, E)$. G is a directed graph in which the vertices are partitioned into $k \geq 2$ disjoint sets $V_i, 1 \leq i \leq k$. In addition, if $< u, v >$ is an edge in E, then $u \in V_i$ and $v \in V_{i+1}$ for some $i, 1 \leq i < k$. We transform this graph into a multistage graph as follows: Two artificial nodes are introduced, $s \in V_0$ and $t \in V_{k+1}$, $|V_0| = |V_{k+1}| = 1$. s is the source and t is the sink. For each vertex $v \in V_1$, we add an edge from s to v. Similarly, for each vertex $w \in V_k$, we add an edge from w to t. The set of added edges is denoted by $E^{'}$. A new graph results. This new graph, $G^{'} = (V \cup \{V_0, V_{k+1}\}, E \cup E^{'})$, is a multistage graph. Let $c(i, j)$ be the cost of edge $< i, j >$. We define

$$c(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = \text{k} + 1; \\ -3 & \text{if the assembly direction and} \\ & \text{assembly tool of node } i \text{ is} \\ & \text{different from that of node } j; \\ -1 & \text{if either the assembly direction or} \\ & \text{assembly tool of node } i \text{ is} \\ & \text{different from that of node } j; \\ 0 & \text{otherwise} \end{cases}$$

The cost of a path from s to t is the sum of the costs of the edges on the path. Then, our problem of finding the degree of uniformity is reduced to the multistage graph problem of finding a minimum cost path from s to t.

An algorithm for solving the multistage graph problem is given in Figure 3. It uses dynamic programming technique [11]. For a graph with n vertices and e edges, the complexity of the algorithm is $O(n+e)$.

```
    procedure Uniformity(E, k, n, P)
    The input is a k stage graph with n
    vertices indexed in order of stages.
    stages. E is a set of edges and
    c(i, j) is the cost of edge < i, j >.
    P is a minimum cost path.
1.  Cost(n) ← 0
2.  for j ← n - 1 to 0 step -1
3.      let r be a vertex such that < j, r > ∈ E and
    c(j, r) + COST(r) is minimum
4.      Cost(j) ← c(j,r) + COST(r)
5.      D(j) ← r
6.  endfor
7.  P(1) ← 1, P(k) ← n
8.  for j ← 2 to k - 1
9.      P(j) ← D(P(j-1))
10. endfor
11. end Uniformity
```

Figure 3: Algorithm for Multistage Graph Problem

At the completion of the procedure, we obtain the degree of uniformity

$$\text{degree of uniformity} = -\text{Cost}(0)$$

4 The Redesign Suggestions Generator

Having performed an objective analysis on a design, we need to help the designer to search for feasible redesign alternatives. There are two types of redesign: local redesign (changes made to only one components) and global redesign (changes made to two or more components).

4.1 Global Redesign Suggestions Generation

An interactive approach is adopted where the computer interacts with the human designer to search for feasible redesign suggestions. A design concepts library is developed. This design concepts library contains all previous design solutions that has been captured. A graph coverage algorithm is used to select a subset of the design concepts that satisfy the functional requirements. The main idea is as follows. The designer will input the functional requirements of the product in the form of a state transitional graph. A X11/Motif graphical user interface is built to support this input task. This state transitional graph, denoted by G, represents the functional behavior of the product to be designed. A design concepts library, consisting of realizable design concepts for different state transitions, is constructed. Associated with each design concept in the library is a cost denoting the assemblability of that particular design concept. During the conceptual design phase, a systematic search is conducted. The goal is to find a subset of the design concepts stored in the design concepts library that realizes all the state transitions in G such that the cost of this subset is minimum. This problem is then reduced to the set covering problem so that we can apply the known solutions for the set covering problem to solve our design concepts selection problem. The subset of solutions selected then forms the global redesign suggestions that will be output to the designer for further consideration.

The set-cover problem is defined as:

In the set-covering problem, the data consists of finite sets $P_1, P_2, \cdots, P_n$ and positive numbers $c_1, c_2, \cdots, c_n$. We denote $\cup(P_j : 1 \leq j \leq n)$ by I and write $I = \{1, 2, \cdots, m\}$, $J = \{1, 2, \cdots, n\}$. A subset J^* of J is called a cover if $\cup(P_j : j \in J^*) = I$; the cost of this cover is $\sum(c_j : j \in J^*)$. The problem is to find a cover of minimum cost.

To reduce the design concepts selection problem to the set-cover problem, we first construct a bipartite graph in the following manner: A bipartite graph is one whose vertex set can be partitioned into two subsets X and Y so that each edge has one end in X and one end in Y. For each state transition edge in graph G, a vertex v is created and placed in the partition X. For each d_i in set D', a vertex w is created and placed in partition Y. An edge connects the node x in partition X to a node y in partition Y if and only if the state transition associated with node x is isomorphic to an edge in the state transitional graph of the design concept associated with node y. To reduce subproblem (2) to the set-covering problem, we let the nodes in partition Y to be the finite sets $P_1, P_2, \cdots, P_n$ in the set-covering problem. $c_1, c_2, \cdots, c_n$ in set-covering problem corresponds to the assembly costs of the design concept associated with each node in partition Y. Finally, each node in partition X corresponds to an element in the set I. With this mapping, we see that a solution to the set-covering problem implies a solution to the design concepts selection problem and vice versa. Now, the set-covering problem is known to be NP-complete. A heuristics for solving the set-covering problem is used to find a near optimal solution to this problem [2]. By using this heuristic algorithm to arrive at the solution for the set-covering problem, we can solve the design concepts selection problem. The algorithm begins with an empty set as the initial solution and successively add to the solution the subset P_j which covers previously uncovered elements of I at a minimum average cost per new cover. The process continues until all the elements of I are covered by the solution set J^*. Figure 4 shows the algorithm. The time complexity of the heuristic algorithm is $O(m^3 N)$.

Step 1. Initialize: The solution set is empty.
No rows are covered.
Step 2. Select an augmenting Column:
Choose the column that has the minimum average cost per uncovered row.
Call this column j^*.
Step 3. Update the solution:
Add column j^* to the solution.
Update row coverage.
If all rows are covered, STOP.
Otherwise Go To Step 2.

Figure 4: Heuristic Algorithm for Solving Set Covering Problem

The outputs of the algorithm is a set of design concepts that can realize the given functional requirements. This set of design concepts form the global redesign suggestions.

4.2 Local Redesign Suggestions Generation

Local redesign suggestions aim to introduce changes that are confined to design features within one physical component. The objective is to improve the product design locally so that the degree of standardization and the degree of uniformity are improved.

The input to the local suggestion generator includes: the CSG representation of the product design, the evaluation results as described in Section 3, the assembly plan, and the functional constraints to be satisfied by the product design. Generation of local redesign suggestions is performed by a rule-based inference engine. This inference engine interacts with a generic toolkit library to derive feasible redesign suggestions. First, the inference engine generates a list of possible redesign actions. The corresponding redesign operators (in the form of routines in the generic toolkit library) are invoked to simulate the suggested redesign operations. The resultant redesign is checked to ensure none of the functional constraints are violated. If a functional constraint is violated, the redesign action is terminated and an alternate redesign action is initiated. At the end of the redesign process, a list of possible redesigns are displayed on screen.

Figure 5 shows the list of local redesign operators available in the toolkit library.

5 Result

A prototype system is being implemented on the Sunsparc workstation with graphic interface to an Iris workstation. The overall view of the system is shown in Figure 6.

Redesign Operator	Description
Cpr(C,P)	Change the CSG primitive of component C to P.
Cor(C, O)	Change the orientation matrix of component C to O. Care must be taken to ensure that the change does not result in an invalid solid.
Cpa(C,P,p-name p-value)	Change the parameter p-name of CSG primitive P of component C p-value.
Cmb(C1, C2)	Combine two components to form one integrated component. Care is taken to ensure the union is a valid solid.
Spt(C, d)	Split the component C along the axis denoted by d.

Figure 5: Table of Available Local Redesign Operators

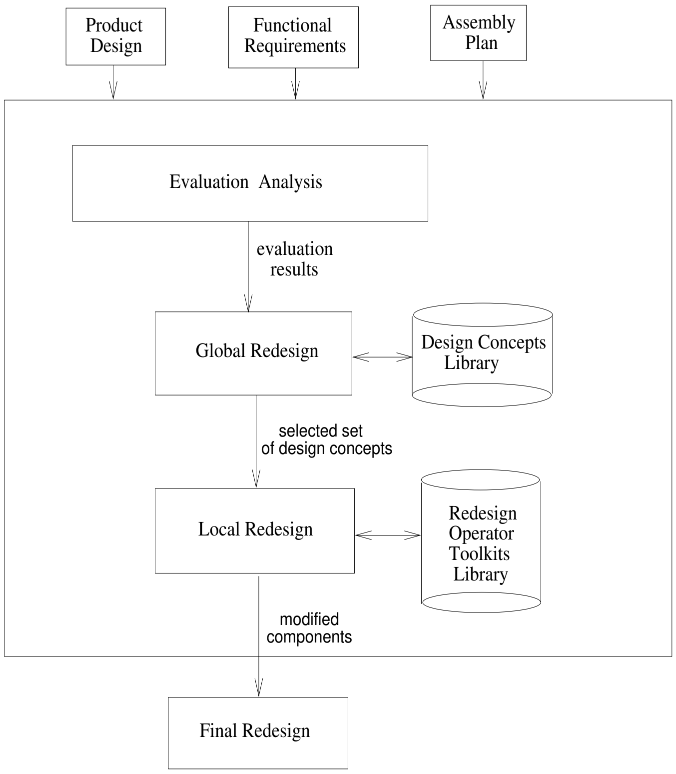

Figure 6: The Overall View of the Prototype System

A real-life example, the tilt-mechanism example, is used to illustrate how the evaluation process and the redesign process are performed. The objective of this design is to support a computer monitor such that it allows the monitor to tilt at a slight angle as determined by the user. Figure 7 shows the original design with eight components.

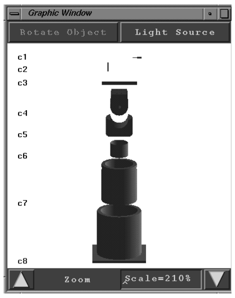

Figure 7: The Tilt Mechanism Example

Based on the design for assembly analysis, it was found that the product needs further improvement. The design concepts selection procedure is invoked to derive a global redesign suggestion. The suggested design is shown in Figure 8.

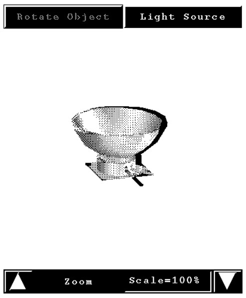

Figure 8: The Global Redesign for the Tilt Mechanism Example

Local redesign analysis is then carried out. It was found that the degree of uniformity can be improved by changing the horizontal insertion of the rod into a top-down insertion. The final design is shown in Figure 9.

With this redesign, we reduced the number of assembly steps from 7 to 3 and the tilt mechanism exhibits a better modular structure. This is certainly a significant improvement in terms of cost for assembly.

6 Conclusion

In this paper, we have discussed an approach to product redesign for robotic assembly. The redesign can be performed at two levels: the global redesign level and the local redesign level. Two quantitative measures are defined to evaluate a design for robotic assemblability. The two measures are: degree of standardization and degree of uniformity. Based on the evaluation results, product-specific redesign suggestions are generated. The analysis and redesign process have been implemented in a prototype system. The tilt-mechanism example is used to illustrate the promise of using the this approach for aiding in product redesign for robotic assembly.

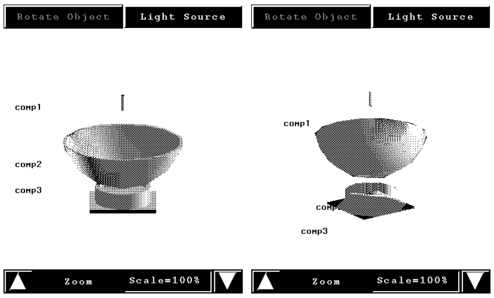

Figure 9: The Tilt Mechanism Redesign

References

[1] M. M. Andreasen, *Design for Assembly*, Springer-Verlag, 1988.

[2] E. K. Baker, "Efficient heuristic algorithms for the weighted set covering problem," *Comput. and Operations Research*, 8(4):303–310, 1981.

[3] J. K. Blundell, "Functional design," *Proc. of Manufacturing International '90*, 2:63–67, Mar. 1990.

[4] J. Boorsma, "Design for assembly," *Proc. of 8th Int'l Conf. Assembly Automation*, pages 141–148, Mar. 1987.

[5] G. Boothroyd, *Design for Assembly Handbook*, University of Massachusetts, Amherst, 1980.

[6] M. R. Cutkosky and D. R. Brown, "Extending concurrent product and process design toward earlier design stages," *Concurrent Product and Process Design*, pages 65–72, Dec 1989.

[7] L. S. Homem de Mello and A. C. Sanderson, "And/or graph representation of assembly plans," *IEEE Trans. on Robotics and Automation*, 6(2):188–199, Apr. 1990.

[8] J. R. Dixon, "Designing with features: Building manufacturing knowledge into more intelligent cad systems," *Manufacturing*, pages 51–57, 1988.

[9] S. Frederic Guggenheim, editor, *Designers in Action 1985-1991*, ASME, 1992.

[10] A.E.K. Holbrook and P. J. Sackett, "Positive design advice for high precision, robotically assembled product," *Developments in Assembly Automation*, pages 181–190, Mar. 1988.

[11] Ellis Horowitz and Sartaj Sahni, *Fundamentals of Computer Algorithms*, Computer Science Press, 1984.

[12] Y. F. Huang and C. S. G. Lee, "A framework of knowledge-based assembly planning," *Proc. of 1991 IEEE Int'l Conf. on Robotics and Automation*, pages 599–604, Apr. 1991, Sacramento, CA.

[13] S. Kota and A. C. Ward, "Functions, structures, and constraints in conceptual design," *The 1990 ASME Design Technical Conferences - 2nd International Conference on Design Theory and Methodology*, pages 239–250, Sept 1990, Chicago, Illinois.

[14] S. Lee and Y. G. Shin, "Assembly planning based on subassembly extraction," *Proc. of 1990 IEEE Int'l Conf. on Robotics and Automation*, pages 1606–1611, 1990, Cincinnati, OH.

[15] B. L. Miles, "Design for assembly - a key element within design for manufacture," *Proc. of the Institution of Mechanical Engineers Part D: Journal of Automobile Engineering*, 203:29–38, 1989.

[16] Jack Mostow, "Design by derivational analogy: Issues in the automated replay of design plans," *Artificial Intelligence*, 40:119–184, 1989.

[17] J. L. Nevins, D. E. Whitney, and T. L. De Fazio, *Concurrent Design of Products and Processes: A Strategy for the Next Generation in Manufacturing*, McGraw-Hill Inc., 1989.

[18] R. M. Noller, "Design for disassembly tactics," *Assembly*, pages 24–26, January 1992.

[19] C. Poli, R. J. Graves, and J. E. Sunderland, "Computer-aided product design for economical manufacture," *Manufacturing*, pages 23–25, 1988.

[20] P. Pu and L. Purvis, "Assembly planning using case adaptation methods," *Proc. of 1995 IEEE Int'l Conf. on Robotics and Automation*, pages 982–983, 1995.

[21] Hubert K. Rampersad, editor, *Integrated and simultaneous Design for Robotic Assembly*, Wiley and Sons Inc., 1994.

[22] Keith Rathmill, editor, *Robotic Assembly*, IFS Ltd, 1985.

[23] J. R. Rinderle and V. Krishnan, "Constraint reasoning in concurrent design," *The 1990 ASME Design Technical Conferences - 2nd International Conference on Design Theory and Methodology*, pages 53–62, Sept 1990, Chicago, IL.

[24] Nam Suh, *The Principles of Design*, Oxford University Press, New York, 1990,

[25] S. N. Talukdar and S. J. Fenves, "Towards a framework for concurrent design," *Concurrent Product and Process Design*, pages 35–40, Dec 1989.

[26] J. D. Watton and J. R. Rinderle, "A method to identify reformulations of mechanical parametric constraints to enhance design," *Design Theory and Methodology DTM '90*, pages 77–84, Sept 1990, Chicago IL.