Proceedings of
The 1996 ASME Design Engineering Technical Conference and
Computers in Engineering Conference
August 18-22, 1996, Irvine, California

96-DETC/DAC-1482

COMBINING INTERACTIVE EXPLORATION AND OPTIMIZATION FOR ASSEMBLY DESIGN

Gerard Jounghyun Kim and Simon Szykman
A-127, B-220, Engineering Design Technology Group
Manufacturing System Integration Division
National Institute of Standards and Technology
Gaithersburg, MD 20899, USA
e-mail: gkim@cme.nist.gov or szykman@cme.nist.gov
Fax: (301)-258-9749

ABSTRACT

This paper presents an integrated framework for assembly design. The framework allows the designer to represent knowledge about the design process and constraints, as well as information about the artifact being designed, design history and rationale. Because the complexity of assembly design leads to extremely large design spaces, adequately supporting design space exploration is a key issue that must be addressed. This is achieved in part by allowing the designer to use both top-down and bottom-up approaches to assembly design. Exploration of the design space is further enabled by incorporating a simulated annealing-based optimization tool that allows the designer to rapidly complete partial designs, refine complete designs, and generate multiple design alternatives.

INTRODUCTION

In order to design and optimize a product, designers must be able to consider different alternatives, perform analysis to guide their own design process and focus in on a "good", if not optimal, design. It is difficult to accomplish such a practice using most current computer-aided design (CAD) systems because their implementations are geared toward supporting only a single level of design abstraction, that is, detailed geometry.

Most engineering design processes proceed in series of stages, such as a functional design stage, a conceptual design stage and a detailed design stage [4]. During the functional design stage, functional requirements of a prospective product are identified. During the conceptual design stage, mappings from the required functional entities of the product to their physical forms occur. Preliminary shapes, sizes, orientations, materials, features, and locations of the physical forms may be determined. Then, the product design is refined to its final form during the detailed design stage through, for example, dimensioning, adding cosmetic features, surface modelling, etc.

One conventional method of handling design alternatives is to use version control. Version control of design files alone can not sufficiently support design space exploration, because each version of a design file simply represents a design snapshot in time. Organizing and managing design alternatives according to different design stages of a given design process provides a much clearer picture of the expanding design space. Version control is suited for exploration within a design abstraction (see Figure 1).

In this paper, we present an interactive tool for assembly design and optimization. In particular, our focus is on design exploration for assembly configurations and its impact on design-for-assembly (DFA) [1]. Most CAD systems model assemblies using geometric relationships quantitatively with geometric constraints such as surface-to-surface, point-to-point, axis-to-axis constraints, etc. Frequently, this necessitates each individual part to be designed to fair amounts of geometric detail, while other assembly-related design alternatives, such as selection of mating types and function-to-form mappings, may be explored at a higher level. If better design decisions can be

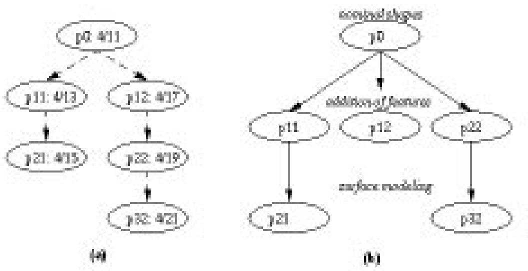

Figure 1. (a) Version control vs. (b) design abstraction for design exploration. With version control, design files are organized only by date, and it is up to the designer to keep track of the flow of design evolution (e.g. link between files do not exist physically). Organizing designs by certain abstraction levels allows for more efficient exploration of design. For example, instead of designing part-22 from part-12, it might be more natural to design part-22 from part-0.

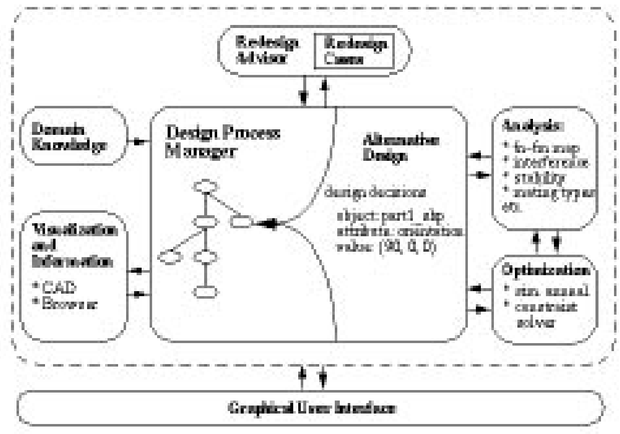

Figure 2. Architecture of the CAMF assembly design framework.

made upstream through exploration, designers can reduce unnecessary backtracking during detailed design stage.

The interactive tool, called CAMF (from Conceptual Assembly Modelling Framework) allows, 1) an ability to create and maintain evolving assembly designs, 2) mixture of top-down and bottom-up assembly modelling, and 3) seamless incorporation of several tools that support exploration of the design space such as analysis programs, optimization techniques and design case bases.

Within the framework, the designer can specify both domain knowledge and problem-specific knowledge relating to the assembly. A designer would begin by creating a representation of the multiple stages in the design process along with constraints that relate design decisions at various stages, and would then proceed to instantiate the design at these stages by adding additional detail.

Although the framework allows the designer to impose a structure on an ill-defined problem, it does not prescribe the manner in which the designer approaches the assembly design. Thus, designers are able to move back and forth between design stages, which represent different views and/or levels of abstraction, incrementally modifying or augmenting different aspects of the current design. This design strategy, which is commonly used by designers, is called the alternate use of abstraction and refinement [8].

As the designer moves back to earlier stages in the assembly process and changes previously-made design decisions, it is not uncommon for other existing parts of the design to become less useful or even rendered infeasible. Therefore, the refinement process is an integral part of the exploration of the design space. It is through refinement that these difficulties are resolved by either by applying local "patches" to the design, or by doing more substantial redesign at various design stages.

The interactive tool can be also used to generate multiple design alternatives, adding to partial design solutions through selection of values for certain types of variables, and optimally-directed refinement of designs created by the user. The aim of this research is not to automate any part of the assembly design process, but rather to provide a human designer with a tool that is used interactively to promote exploration of the design space. As a natural extension, we also discuss different ways of assisting the exploration of a potentially very large design space using CAMF and illustrate it by applying simulated annealing to optimize certain appropriate design variables and automatically generating plausible design alternatives.

RELATED WORK: CAD FRAMEWORKS FOR ASSEMBLY MODELLING

Relatively little attention has been paid to design or modelling methodologies for assemblies. Computer-aided design of assemblies can be categorized largely into two groups, top-down [4][10][11][12] and bottom-up approaches. Top-down assembly modelling is based on first generating a functional or symbolic description of a prospective design, and performing a stepwise refinement of component geometries. The functional model should be validated to some extent before moving into individual part design. A bottom-up approach starts with component design (with a *mental* model of the design) followed by continuous revision of the mental models and part design [13]. Designers, in applying the alternate use of abstraction and refinement, mix both top-down and bottom-up approaches; thus, the CAMF framework supports both approaches to assembly design.

Wolf has recognized a need for creating a new type of CAD framework which he defines as "a software infrastructure that provides a common operating environment for CAD tools" to address the increasingly complex design activities in the context of concurrent engineering [14]. Key features of such CAD frameworks include an extended data management scheme for different types of design objects such as versions, a design process-driven user interface, and an open and flexible architecture - principles that are shared by CAMF. However, most work to date relating to CAD frameworks is oriented toward electronic circuit design [2][15]. Mechanical assembly design has different requirements in terms of user interaction and design exploration. Peplinski et al. [9] describe a system for evaluating design-for-manufacturability at different abstraction levels and proposing a design based on the result.

CAMF

Overview

Figure 2 shows an overview of CAMF. The central component of CAMF is the design process manager (DPM). The DPM provides a pictorial view of design evolution and alternatives using a tree-like representation where levels in the tree represent stages in the design process and each path to a node corresponds to a design alternative. Through the DPM, the user is able to traverse freely around the design space and create new alternatives (see Figure 4). Each node represents a subspace of the entire design space and is defined by series of design decisions and supporting design rationale supplied by the designer. The DPM allows the user to specify a generic design process model and then to design within that model in a structured manner. Therefore, only certain types of design decisions may be applicable at any given design stage.

The DPM is integrated with several other design tools, namely, a geometric modeller (e.g. Spatial Technologies' ACIS Solid Modeller, SDRC's I-DEAS Master's Series[1]), a knowledge browser (for visualization), a number of design analysis tools (e.g. interference checking and evaluation of mating types), a simulated annealing optimization tool, and a case-based redesign advisor [6] that uses output from the analysis tool to generate new design alternatives. The next section presents a more detailed explanation of the data structures used in CAMF.

[1] Use of these commercial products does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the products identified are necessarily the best available for the purpose.

Representation of Designs

A design can be viewed as a sequence of state transitions, where at each design state, design decisions are applied, incrementally satisfying various types of constraining design information such as geometric constraints, engineering specifications, and design rationale. Design decisions are continuously applied until an acceptable design state is reached. Let us first define a notion of a state-based and a decision-based representation of *Design Space*.

Definition 1 (State-based Design Space). *A State-based Design Space is defined as a 5 tuple, $D_s = (\mathcal{S}, \mathcal{A}, \mathcal{T}, S_0, \mathcal{S}_\mathcal{F})$, where,*
(1) $\mathcal{S}$ is a nonempty finite set of design states,
(2) $\mathcal{A}$ is a nonempty finite set of design decisions,
(3) $\mathcal{T}$ is a function which maps $\mathcal{S} \times \mathcal{A}$ into $\mathcal{S}$,
(4) S_0 is the start state (null state), and,
(5) $\mathcal{S}_\mathcal{F}$ is a set of final states (or acceptable design instances).

Figure 3(a) illustrates a state-based representation of a design space. Each design state contains its own design description (e.g. a geometric model, objects, etc.); a design decision is applied to produce the next design state. Alternatively, the same design space may be described using a decision-based representation in which each *design step* contains a set of design decisions (see Figure 3(b)).

Definition 2 (Decision-based Design Space). *A Decision-based Design Space is defined as a 4 tuple, $D_D = (\mathcal{S}, \mathcal{T}, S_0, \mathcal{S}_\mathcal{F})$, where,*
(1) $\mathcal{S}$ is a nonempty finite set of set of design steps (i.e. set of design states as defined in Definition 1),
(2) $\mathcal{T}$ is a function which maps $\mathcal{S}$ into $\mathcal{S}$,
(3) S_0 is the start state (null state), and
(4) $\mathcal{S}_\mathcal{F}$ is a set of final states (or acceptable design instances).

Figure 4 shows a decision-based design space for the assembly design problem used as an example in this paper (the design process of a television remote control) where nodes represent design steps. Each set of design steps connected by arrows represents a distinct design context, i.e., a design alternative. The highlighted design context is the current design thread, with each row representing a particular design stage.

Definition 3 (Design Step). *A Design Step with type i, s_i, is a set of design decisions. Only design decisions with type i may belong to s_i.*

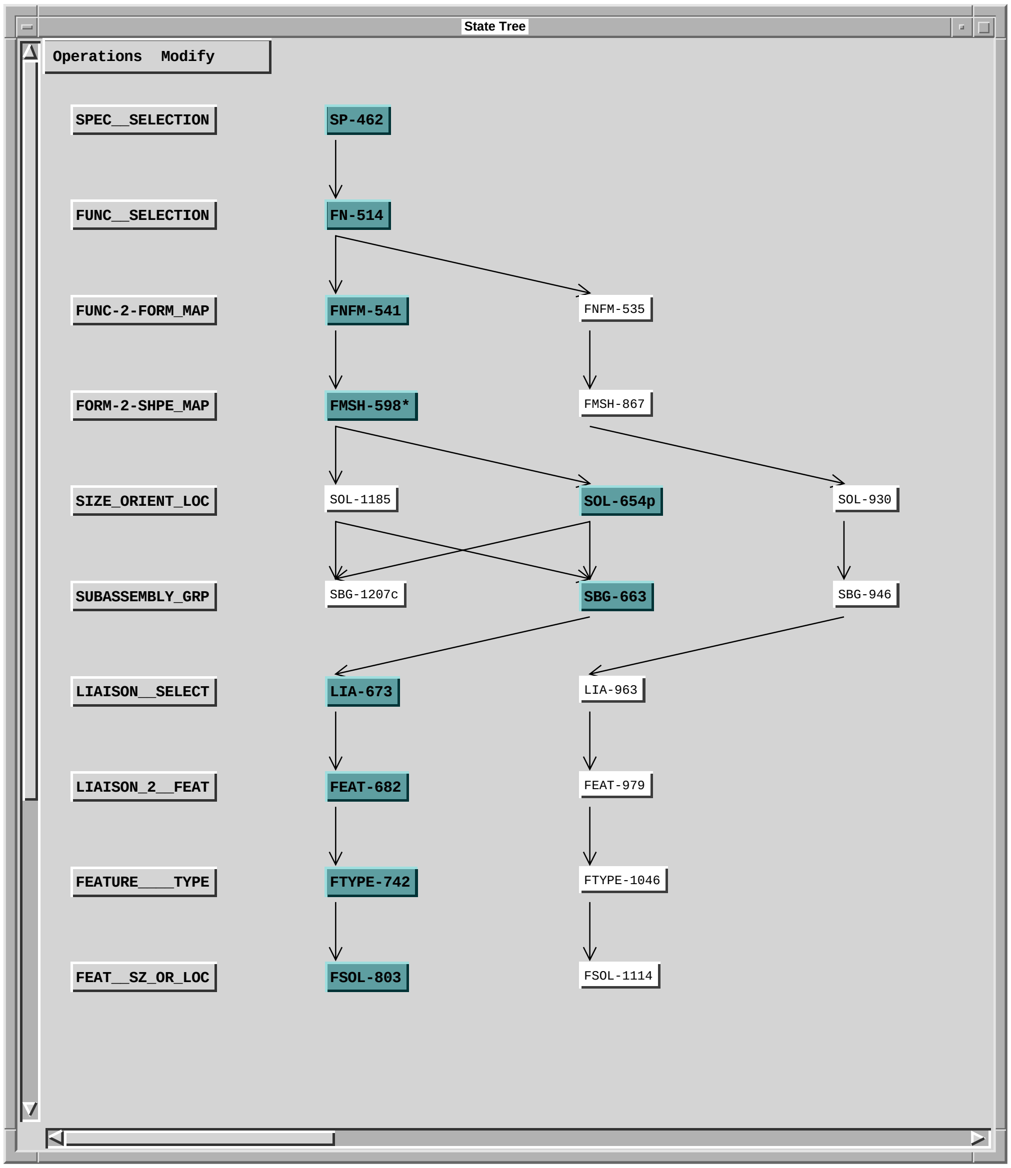

Figure 4. A design space for a television remote control assembly design task.

A design decision either creates a new design object or determines values for a design attribute of an existing design object. Table 1 shows examples of different types of design objects and attributes that are represented in CAMF.

Definition 4 (Design Decision). *A Design Decision with type i, a_i, is,*
(1) a 2 tuple, [O, T], where O is a new object and T is its type, or,
(2) a 3 tuple, [O, A, V], where O is a design object, A is a design attribute (or relation), and V is a set of values (or objects).

Definition 5 (Design Context). *A Design Context, C, is defined as a set of design steps, such that if s_i and s_j belong to C, then $i \neq j$ (i.e. a path to a node in Figure 4).*

Definition 6 (Consistent Design Context). *A Consistent Design Context is defined as a design context in which none of its design step invalidate each other.*

Each design context represents a decision-based representation of a design alternative. Its corresponding design

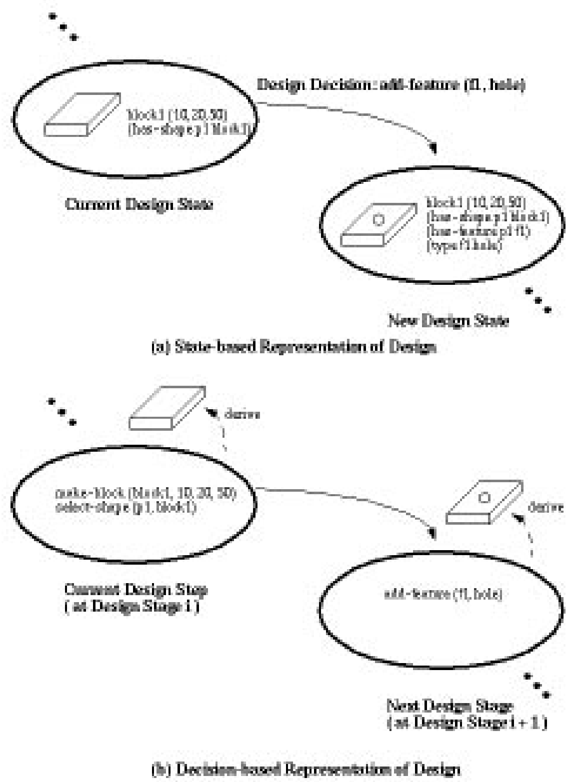

Figure 3. (a) A state-based design representation vs. (b) a decision-based design representation.

descriptions can be recreated by sequentially replaying all the design actions contained in a design context.

Definition 7 (Valid Design Context). *A Valid Design Context is defined as a design context which is "feasible" with respect to design constraints (e.g. geometric and symbolic) known at the time of creation.*

CAMF currently does not ensure complete consistency nor validity of a design context. However, the refinement tool (described in greater detail in a later section) is able to generate design alternatives that are consistent subject to constraints on certain types of design attributes.

CAMF uses the decision-based representation because it is more natural to describe and organize the assembly design process using design actions. The transition from one state to another is defined in terms of a predefined design process that divides the design space into disjoint subspaces. Both symbolic and geometric descriptions of the evolving design are "derived" when needed, rather than storing all the alternative designs in accumulation. This promotes reuse of certain design steps and also helps designers in focusing on certain design stages.

Table 1. Examples of design objects and some of their attributes.

Object	Attributes (partial list)
function	:function-of
specification	:specification-of
subassembly	:has-function, :stability
part	:has-function, :size, :location
feature	:type, :location, :size
liaison	:type, :mating-parts
design rationale	:justifies, :type

Assembly Design Process

The generic assembly design process model employed in CAMF allows the designer to define different types of design stages and, in effect, the levels of design abstraction for a given design task. Table 2 illustrates the different design stages for the television remote control design example. Each design stage is associated with a particular set of objects and attributes that can be created or modified only during that stage. Such specification of design processes is input by the user to the system as domain or design knowledge. For example, during the function element selection stage, only functional elements can be created and they are associated with the assembly design through *:function-of* relation.

For the design process model shown in Table 2, design stages are explicitly defined for their contribution and impact to assembly cost. For instance, the function-to-form mapping stage determines the number of physical components in the assembly. The number of components is a very important assembly cost variable (because it has a direct relationship to required handling and feeding devices on the factory floor), and the designer might want to consider and maintain multiple alternatives. Different combinations of components' sizes, orientations, and locations (see the fifth design stage, *SOL*, in Table 2) may result in a stable or unstable assembly, which has a direct implication with regards to fixture requirements.

Table 2. CAMF Design Stages for the television remote control example.

Order	Design Stage	Mnemonic	Synopsis
1	Specification Selection	sp	create and select set of specifications (SP)
2	Functional Element Selection	fn	create and select set of functional elements (FE)
3	Function-to-Form Mapping	fnfm	create conceptual forms and map FE's, to conceptual forms
4	Shape Selection	fmsh	create and select appx. shapes for forms
5	Size, Orientation, and Location of Forms	sol	select relative sizes, orientations, and locations for shapes
6	Subassembly Grouping	sba	select subassembly groupings
7	Liaison Creation	lia	create liaisons among forms
8	Feature Selection	ft	create and select features for liaisons
9	Feature Shape Selection	ftsh	create and select shapes for features
10	Size, Orientation, and Location of Features	fsol	select relative sizes, orientations, and locations for features

DESIGN EXPLORATION

Manual Design Space Exploration

In this section, an example design session of creating different assembly alternatives for the television remote control with CAMF is illustrated. Due to lack of space, only a subset of the overall design process is presented in this section: the selection of batteries and their spatial configurations (design at other stages is illustrated in the next section).

A design space corresponding to the television remote control assembly was shown in Figure 4. Initially, the design space starts out empty. During the first two design stages, specifications, functions, and their inter-relationships are defined. At design stage 3 (function-to-form mapping), two alternatives are generated by the designer: *FNFM-535* and *FNFM-541* (see the third row of Figure 4). In the former alternative, a function, *generate-power-fn*, is mapped onto a configuration consisting of two parts: *bat1* and *bat2* (this mapping is shown in Figure 8). In the latter design step, *generate-power-fn* is mapped to a configuration which consists of only one conceptual part, *9v-bat*.

At the next design stage, the designer selects shapes for the two battery configurations. Design step *FMSH-598* corresponds to the selection of cylinders as generic shapes for the 2 AA batteries, while *FMSH-867* uses a rectangular block to represent the approximate shape of a 9 Volt battery. In the fifth design stage, *SOL*, different geometric configurations are explored using an interface to a geometric modeller and size, orientation and location operators; the corresponding geometric descriptions are shown in Figures 5, 6, and 7.

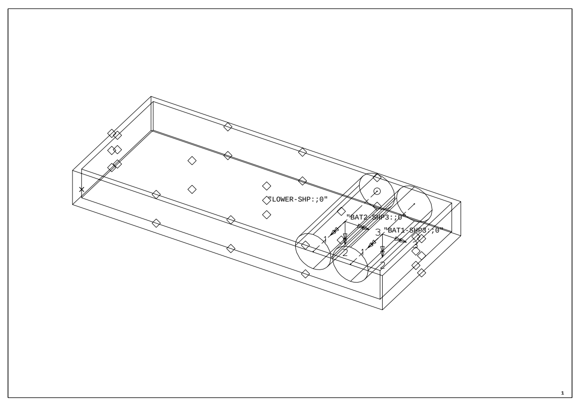

Figure 5. Battery configuration 1 in SOL-1185 in Figure 4.

Further design refinements continue, and among the three design threads, one alternative (*SBA-207*, which groups two batteries as a subassembly) is manually discarded by the user by marking this particular design path invalid. Features are created and placed on each design thread, and two assemblies designs are created.

At any design stage, the designer has the option of per-

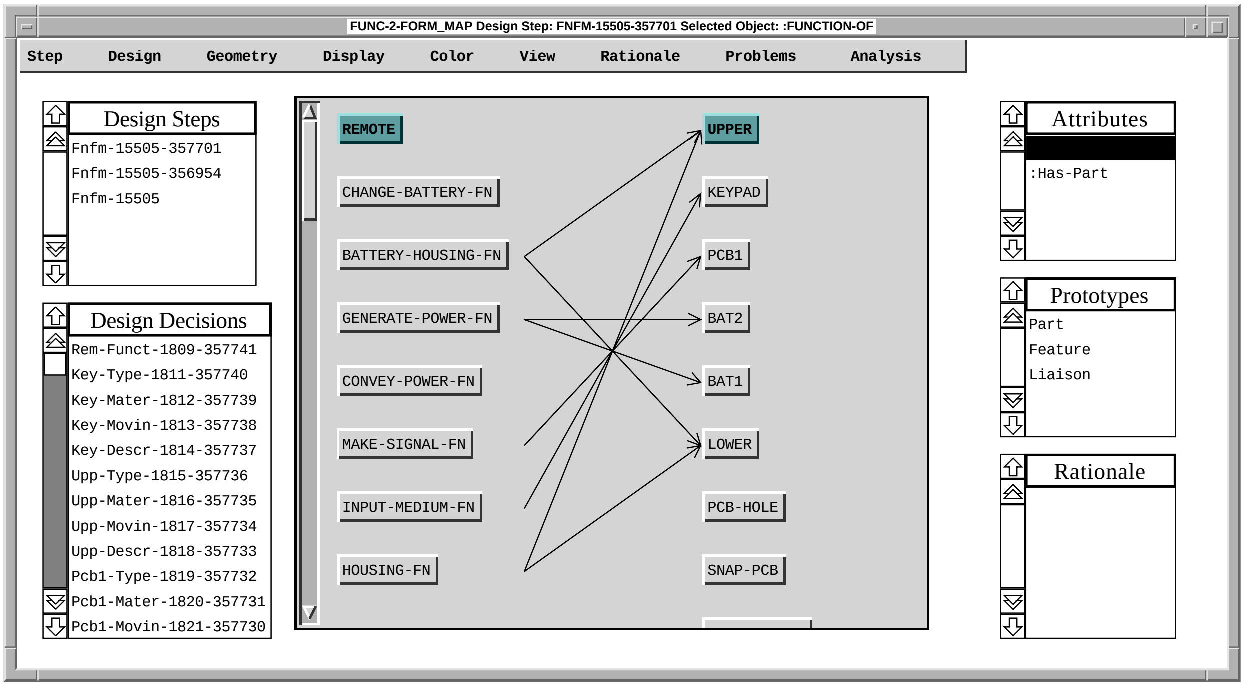

Figure 8. Design step, FNFM-535, in which *generate-power-fn* **is mapped to two physical components** *bat1* **and** *bat2***.**

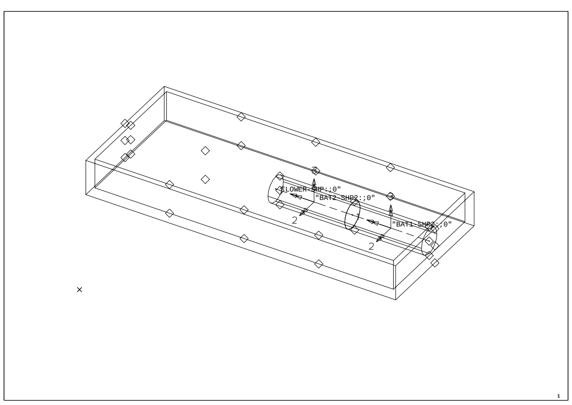

Figure 6. Battery configuration 2 in SOL-654 in Figure 4.

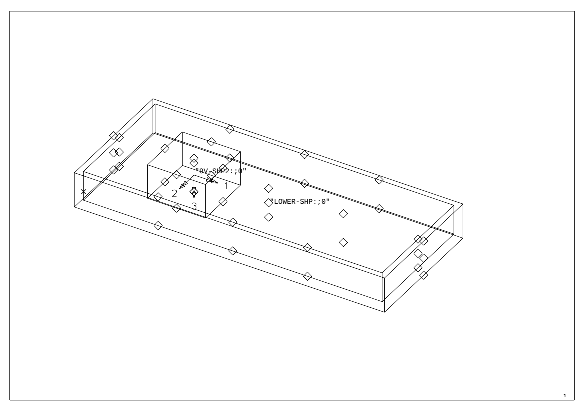

Figure 7. Battery configuration 3 in SOL-930 in Figure 4.

forming any applicable analysis (available through a menu seen in Figure 8) to help guide the design space exploration. Designers may focus on one part of the design, as is illustrated in this example, and then modify or complete unfinished parts of the design at preceding or subsequent design stages.

Computer Aided Design Exploration Using Simulated Annealing

Simulated Annealing Simulated annealing is a stochastic optimization technique that was introduced by Kirkpatrick et al. [7]. At the start of an optimization using simulated annealing, the algorithm begins at an initial design state. The algorithm then takes a step to a new design state by perturbating the current design. The objective function value of the new state is compared to that of the previous state. If the new state is better than the previous one, it is accepted; if it is worse, it is accepted or rejected with some probability.

The probability of accepting an inferior state (i.e. a step in a direction away from an optimum) is a function of a parameter called temperature. Initially, the temperature (and therefore the probability of accepting inferior steps) starts out high. Since many inferior steps are accepted, this results in near-random exploration to find promising regions of the design space. As the optimization proceeds, the temperature decreases and fewer inferior steps are accepted, making the exploration less random. As the temperature continues to decrease, the algorithm reaches a point where the search resembles a downhill search because virtually no inferior steps are accepted. This allows the algorithm to converge to local optima in the current region of the design space.

The control of the temperature parameter is done using an annealing schedule, which is a critical part of a simulated annealing algorithm. The annealing schedule used for this research is an adaptive annealing schedule which, after the optimization begins, calculates an initial temperature according to a scheme proposed by White [16] and calculates temperature reductions using a method described by Huang et al. [5]. The advantage of this approach is that the annealing schedule is tailored to a particular problem during the optimization in contrast to a fixed annealing schedule where the schedule parameters are selected ahead of time and do not change during the optimization. The adaptive annealing schedule results in improved efficiency and convergence characteristics.

Formulation of the Optimization Problem The previous section described simulated annealing at a generic level. This section describes the problem-specific aspects of the algorithm formulation - the design variables, the method of perturbating designs and the objective function.

Because the assembly design process involves a great deal of knowledge, it is not possible to completely automate it. In practice, automated search or design generation can not be achieved at all design stages, nor over all design variables. For the television remote design example, refinement takes place over four variables through the processes of material selection for components, mating type selection for liaisons among parts, feature selection to realize the matings and selection of generic shapes for those features. These variables will be referred to as *refinement variables*.

For this example, each of the refinement variables is changed at a different design stage (stages 3, 7, 8 and 9, respectively) as shown in Table 3. However, this is not a general rule and it is possible to have more than one refinement variable at each design stage. It should also be noted that design stages may contain more than one type of variable. For instance, while material is the only refinement variable at the function-to-form mapping stage, the designer can make other design decisions that affect other design attributes at that stage.

The input to the refinement phase is a design produced by the designer. This design may be a complete design or a partial design having values for one or more refinement variables left unassigned. At each iteration in the simulated annealing algorithm, a design stage that contains one or more refinement variables is randomly selected to be perturbed. Next, a design decision is chosen at random from that level, and one of the refinement variables is selected to be modified. The current value for that refinement variable is then changed to one selected from a list of feasible new values. This list is generated by taking the list of possible values for that variable and removing the current value as well as any values that would violate constraints which have been previously entered as part of the domain knowledge.

Depending on the combination of values of refinement variables for other design decisions at various design stages, it is possible for the list of feasible new values to be empty because of constraints that affect allowable values (e.g. the mating type, or liason, affects allowable materials and vice versa). When this occurs, a penalty is assigned to the objective function. As the optimization proceeds, these penalties are eliminated by resolving the source of the violation; that is, in a subsequent iteration, one of the constraints that caused variable values to be removed from the feasible list is rendered inactive by changing a different variable elsewhere. This results in a feasible list which is no longer empty. The next time that design decision is selected for a design perturbation, a feasible value can be found, eliminating the violation penalty.

Results We now return to the television remote control design example and illustrate the use of the simulated annealing refinement tool as an aid for the exploration of the design space. The designer begins by generating an initial attempt at a design and by using the refinement tool interactively, is able to rapidly guide design space exploration towards promising designs.

In order to preserve a record of the interactive exploration, a copy of the current design context is made at the stages which the simulated annealing refinement occurs.

The user interface provides ways to easily represent design constraints and specify the design attributes to be affected by the simulated annealing algorithm. The designer inputs constraints relating refinement variables at different stages (as described in the previous section) and specifies the design stages over which refinement is to take place. In this case, the designer selects all four stages that include refinement variables, though fewer could have been selected. The alternative design context is illustrated by the highlighted path in Figure 9.

Table 3 shows the results of applying the simulated annealing algorithm to optimizing the four refinement variables. In the first run, the refinement was used to generate a design that improved the designer's initial attempt, subject to any relevant design constraints. In other words, the algorithm optimizes the total cost[2] without regards to the functional aspect of the design.

The designer realizes that the battery material should have been fixed; because it was allowed to vary, the refinement selected wood, a low cost material but one that is inappropriate for batteries. The designer manually fixes the material to *MAT-12*, which corresponds to the material cost of batteries in the CAMF materials library. The designer then makes a similar modification to the keypad material (rubber for human factor reasons) and fixes that variable as well by setting tags through the user interface.

The algorithm is run the second time with these new constraints (see Table 3). Note that the changes in material have caused to changes in liaisons (mating types), which have in turn affected the feature and feature shape selection at other design stages. After this refinement, the current design uses wood for the upper and lower container of the remote control (fifth column of Table 3) due to its low material cost. In practice, wood is not used despite a low material cost because it is more costly to manufacture and assemble wooden parts.

From this result, the designer realizes that material cost should not be weighted as heavily relative to the other costs and adjusts the weights in the evaluation function, again through the user interface. The algorithm is run a third time, and plastic materials are selected for the two container components, and again liaisons have changed accordingly as shown in Table 3. Under the new weighting, more expensive materials were selected in order to allow less expensive mating types, which lead to lower assembly costs.

DISCUSSION

The assembly design tool developed through this work makes it easier for a designer to explore the design space associated with assembly design tasks. Through the use of this framework, a designer can represent a design process, and both represent and evaluate design artifacts. By using CAMF interactively, the designer can perform design refinement and redesign in a variety of ways - either by locally patching a design (extending a design at one design stage) or by moving back to previous levels of the hierarchy.

At some stages, such as specification selection and function-to-form mapping, the exploration of the design space and the evaluation of designs is left to the user. At other stages, the designer can generate an initial attempt at an design, which may be partial or complete, and then call on a simulated annealing refinement tool for further interactive design space exploration.

The intent of this work is not to automate the design process, since developing a system that captures all relevant domain and design knowledge is in practice extremely difficult (if not impossible) for non-trivial domains. Rather, the aim is to create an interactive design tool that is able to aid the designer in generating alternatives, directing search towards good or optimal designs, and evaluating designs.

There are a number of motivations for the use of simulated annealing as a design space exploration technique for this work. Due to the combinatorial nature of the problem addressed in this research, a complete search of the design space is infeasible. The number of design alternatives (each corresponding to a path through the hierarchy of design stages) is equal to the number of stages over which the refinement occurs, multiplied by the number of design decisions at each design stage, multiplied by the number of refinement variables for each design decision, multiplied by the number of allowable values for each refinement variable. This number can rapidly grow to an unmanageable size. Simulated annealing is able to find optimal or near-optimal designs without performing an exhaustive enumeration of all possible alternatives, and is therefore well-suited for a problem such as this one, as well as other classical combinatorial optimization problems such as the traveling salesman and graph partitioning problems [3].

A more traditional alternative to exhaustive enumeration would be to use a best-first search technique starting at the first design stage and moving through subsequent stages. The difficulty with this approach is that the ability to generate a given solution is dependent on the sequence that the design stages are visited, due to constraints between design attributes at different stages. For instance, consider a design problem where the globally optimal solution uses a particular liason. If assignment of materials is done before assignment of liaisons, a decision that selects the best (e.g. lowest cost) material for a pair of parts may preclude the selection of the best liason for the parts at some later stage because not all liaisons are possible with all materials. This would prevent a best-first search from finding the overall best solution.

The simulated annealing algorithm is able to avoid this difficulty because the design refinement task has been for-

[2]Currently, a weighted sum of machining cost, assembly cost and material cost is used as a simplified measure of the overall cost.

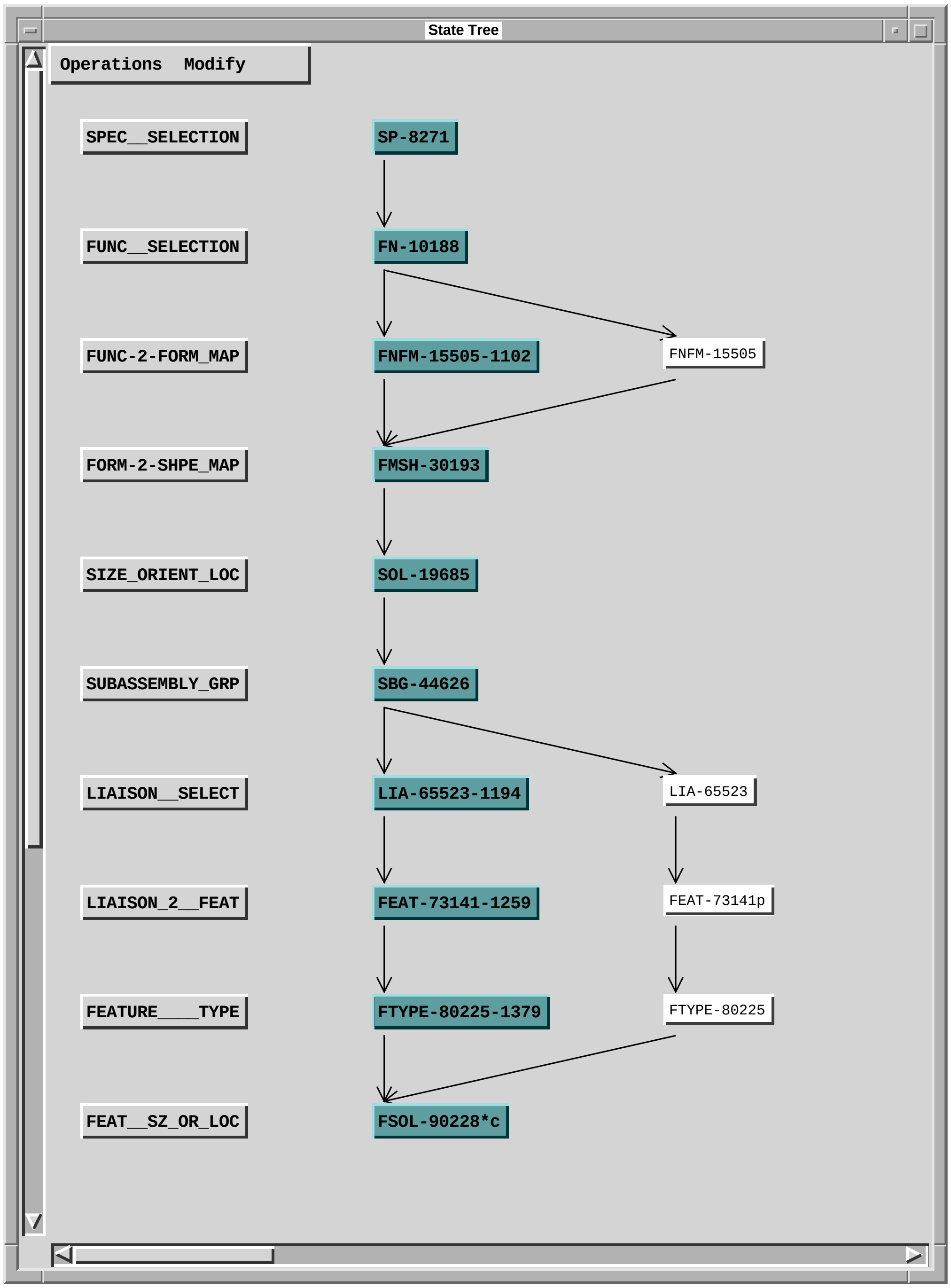

Figure 9. Generating an alternative design context.

mulated as a global optimization problem. Rather than refining the design sequentially by moving down the design stage hierarchy, each iteration the algorithm can make a change at any of the applicable design stages. Thus, this approach does not lead to design decisions which result in committing to a path that would make finding the global optimum impossible.

Table 3. Interactive design using the simulated annealing refinement tool.

Design Stage (Order)	Object	Attribute	Run 1	Run 2	Run 3
Function-to-Form	upper	material	STEEL	WOOD	PLASTIC
Mapping (3)	lower	material	PLASTIC	WOOD	PLASTIC
	keypad	material	STEEL	RUBBER	RUBBER
	pcb	material	PLASTIC	PLASTIC	PLASTIC
	battery1	material	WOOD	MAT-12	MAT-12
	battery2	material	WOOD	MAT-12	MAT-12
Liaison	lia-upper-keypad	type	SOLDER	INSERTING	PRESS-FIT
Selection (7)	lia-lower-pcb	type	PRESS-FIT	INSERTING	PRESS-FIT
	lia-bat1-lower	type	INSERTING	INSERTING	INSERTING
Feature	pcb-feat	type	BOSS	PAD	PAD
Selection (8)	lower-feat1	type	GROOVE	SLOT	SLOT
	pcb-support	type	FEATURE-4	FEATURE-4	FEATURE-4
	button1	type	PAD	PAD	PAD
	upper-feat	type	GROOVE	GROOVE	GROOVE
Feature Shape	pcb-feat	has-shape	CIRC-PROT	RECT-PROT	RECT-PROT
Selection (9)	lower-feat1	has-shape	CIRC-HOLE	RECT-HOLE	RECT-PROT
	pcb-support	has-shape	FEAT-SH-8	FEAT-SH-8	FEAT-SH-8
	button1	has-shape	RECT-PROT	RECT-PROT	RECT-PROT
	upper-feat	has-shape	RECT-HOLE	RECT-HOLE	RECT-HOLE

CONCLUSION

This paper presents an assembly design framework that creates and manages multiple design alternatives at different levels of abstraction according to a generic design process model in conjunction with domain-specific knowledge. Within the CAMF framework, design process management is achieved using a decision-based representation which allows the designer to represent knowledge about the design process and constraints, as well as information about the artifact being designed, design history, and design rationale.

Because the complexity of many assembly design problems can lead to extremely large design spaces, enabling adequate support of design space exploration is a key issue that must be addressed in a CAD tool. CAMF enables both top-down and bottom-up approaches to assembly design by allowing the designer to freely move back and forth between design stages. The exploration of the design space is further supported by incorporating a simulated annealing-based optimization tool, that allows the designer to rapidly complete partial designs, refine complete designs, and generate multiple design alternatives.

The current framework provides the user with substantial flexibility to represent knowledge about the design process and artifact. One drawback to this approach is the difficulty that can arise with explicitly representing this knowledge within CAMF. While designs generated by the refinement/optimization tool are consistent subject to user-specified constraints that relate refinement variables to one another, knowledge about interactions between non-refinement variables resides largely with the designer. Thus, consistency of certain aspects of the design must be maintained by the user. Future work in expanding the scope of the refinement tool past the four design stages currently implemented will require addressing these issues.

An area of future work that will affect the capabilities of CAMF is improvement of the design evaluation. Because of the difficulty in automating the evaluation of some assembly-related attributes such as symmetry or tangling of parts, CAMF will remain an interactive design tool. However, as analysis and evaluation improve, search through the design space will be better guided toward good solutions, whether the exploration be manual or computer-aided using the refinement tool.

The development and implementation of CAMF is still a continuing effort. A need for an assembly design framework arose from earlier efforts to build a DFA redesign system [6]. Another future interest is to compare design processes along different dimensions such as number of backtrackings, number of user interactions, number of design alternatives generated, and number of abstraction levels in order to quantitatively evaluate the effectiveness of tools like CAMF.

ACKNOWLEDGMENTS

This research was conducted at the National Institute of Standards and Technology and was supported by the National Research Council Postdoctoral Research Associateship program.

REFERENCES

G. Boothroyd and P. Dewhurst. *Product Design for Assembly Handbook*. Boothroyd and Dewhurst, Inc., 1989.

J. Brockman and S. Director. The Hercules CAD task management system. *Proceedings of IEEE ICCAD-91*, pages 254–257, 1991.

N. Collins, R. Eglese, and B. Golden. Simulated annealing - an annotated bibliography. *American Journal of Mathematical and Management Sciences*, 1988.

J. Gui and M. Mantyla. Functional understanding of assembly modeling. *CAD*, 26:435–451, June, 1994.

M. Huang, F. Romeo, and A. Sangiovanni-Vincentelli. An efficient general cooling schedule for simulated annealing. *ICCAD-86: IEEE International Conference on Computer-Aided Design - Digest of Technical Papers*, pages 381–384, 1986.

G. Kim and G. Bekey. Design-for-assembly by reverse engineering. *Artificial Intelligence in Design, 94*, pages 717–734, 1994.

S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science, 220(4598)*, pages 671–679, 1983.

J. Paz-Soldan and J. Rinderle. The alternate use of abstraction and refinement in conceptual mechanical design. *Proceedings of the ASME Winter Annual Meeting*, 1989.

J. Peplinski, P. Koch, J. Allen, and F. Mistree. Flame: A manufacturability evaluator for use at different levels of abstraction. *Proceedings of 1995 ASME Design Engineering Technical Conference*, pages 893–900, 1995.

R. Popplestone. Edinburgh designer system as a framework for robotics. *Proceedings of IEEE International Conference on Robotics and Automation*, pages 1972–1977, 1987.

D. Rocheleau and K. Lee. System for interactive assembly modelling. *Computer-aided Design*, 19:65–72, 1987.

R. Sodhi and J. Turner. Representing tolerance and assembly information in a feature based design environment. *Proc. of ASME Design Automation Conference*, pages 101–108, 1991.

R. Sodhi and J. Turner. Towards modelling of assemblies for product design. *CAD*, 26:85–97, June, 1994.

P. van der Wolf. *CAD Frameworks: Principles and Architecture*. Kluwer Academic Publishers, 1994.

P. van der Wolf, G. W. Sloof, P. Bingley, and P. Dewilde. On the architecture of a cad framework: The nelsis approach. *Proceedings of European Design Automation Conference*, pages 29–33, 1990.

S. White. Concepts of scale in simulated annealing. *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers*, pages 646–651, 1984.