Required features of CAD system as a real design tool

Kunwoo Lee
School of Mechanical & Aerospace Engineering, Seoul National University
kunwoo@cad.snu.ac.kr

Abstract: In this paper, the limitations of the current CAD systems are reviewed and the required features they should have to become a real design tool are identified. Current CAD systems are good enough to be used as a tool to manipulate three-dimensional shapes. This is a very important capability to be owned by a design tool because a major portion of designers' activities is spent on the shape manipulation in the design detailing process. However, the whole design process involves a lot more than the shape manipulation. Currently, these remaining tasks, mostly logical reasoning process, are processed in the designer's brain. This is one of the reasons why current CAD systems are not considered as the real design tool. The research activities being performed at Seoul National University to overcome this limitation are discussed in this presentation. Their goal is to provide the current CAD systems with the capabilities for design process modeling, evolutionary design using design intent and accumulated information, and case-based reasoning.

Key words: Design process modeling, evolutionary design, design information, design intent, case-based reasoning

1. INTRODUCTION

Traditional mechanical CAD systems are heavily used as a tool in the design process owing to their shape manipulation capabilities. This is a very important capability to be owned by a design tool because a major portion of designers' activities is spent for the shape manipulation in the design detailing process. However, the whole design process involves a lot more than the shape manipulation. It should be noted that the shape manipulation activity occupies only half the track of a design process. The remaining half

© Springer Science+Business Media New York 2001

of the track is the function decomposition process. When a designer is assigned a design task, it is usually specified by the desired functions of the final design output. Thus the designer's role is to come up with a product composed of parts with the necessary geometry such that the assembled product provides the desired functions. The designer usually can not come up with the detail geometry of the parts and the inter-part relationship directly from the desired functions unless a product with very similar functions already exists. Instead, the designer breaks down each desired function into the combination of smaller functions until he/she can match the functions with the geometry of parts or their relationship. When the necessary geometry or part relationship is determined, the designer details the design by using the shape manipulation capability of current CAD systems. The function decomposition process that occurs simultaneously with the geometry detailing process is completely handled in the designer's brain. Thus the whole design process or the design history is not stored and can not be reused by other designers. Only part of it can be preserved in the form of the technical reports.

Even though the design process modeler supporting the function decomposition process as well as the geometry detailing process became available, it only could help the designer's function decomposition process in a systematic way as a book keeping software helps the accounting task. That is, the intelligent task would be still assigned on the human. Even with the design process modeler, the design process would be completely guided by the designer. The only difference would be that the resulting design process itself is stored and can be referred later. In the conventional design process, the previous design process can be retrieved partially by counseling the designer who performed the design process or referring the technical document. For the current CAD systems to have some intelligence, they are desired to have a synthesis capability especially for conceptual design. If this capability could be realized, the conceptual design process would be guided by a computer instead by the designer. Of course, the computer cannot perform the synthesis process without any human intervention, but it can be possible with the minimum human intervention if the computer is provided with many design cases and the reasoning capability.

When a designer carries out a design task, he/she usually does not finish the design in one attempt, but iterates the design process to evolve into an improved design. Thus it would be nice if a CAD system could simulate and help this iterative process. For this purpose, the designer should be able to create or select the design parameters and define the objective function with the design parameters while he carries out the design process in a CAD system.

As an effort to provide the capabilities listed above, the following features are proposed as the key elements to realize a CAD system of a new concept. The necessary framework and the implementation ideas of the these features are described in the remaining sections.

- Design process modeling embodying function decomposition and shape manipulation
- Case-based reasoning for conceptual design
- Evolutionary design with defining design parameters and objective functions

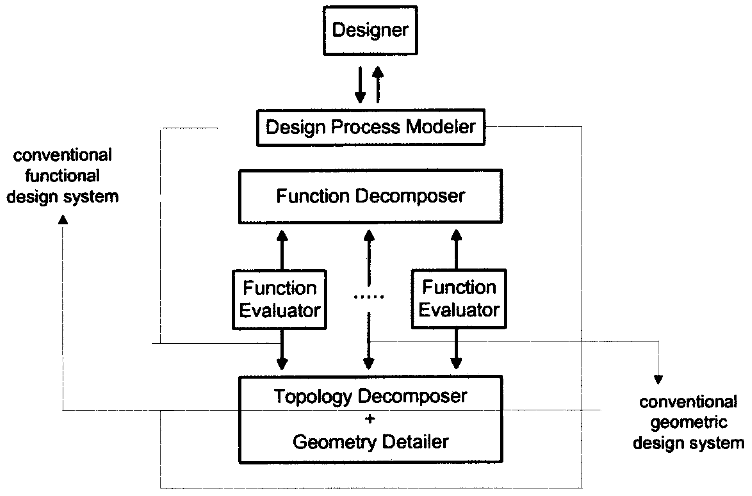


Figure 1. Architecture of the proposed design process modeller

2. DESIGN PROCESS MODELING EMBODYING FUNCTION DECOMPOSITION AND SHAPE MANIPULATION

When we review the design process, we can notice that the designer decomposes the given required functions into sub-functions and the structure of the whole assembly is decomposed into subassemblies in accordance with the decomposed function[1-6]. Thus the earlier stage of design activity is dominated by the function decomposition activity while the later stage is occupied by the shape manipulation activity. To support these two activities in a unified environment, the structure of the system, called design process modeler, is proposed as shown in Figure 1. The design process modeler is composed of three processors: function decomposer, shape detailer (topology decomposer + geometry detailer), and function evaluator. Function decomposer aids the function decomposition process of the designer while the shape detailer aids the decomposition of the assembly structure as he develops the design. Function evaluator verifies the detail geometry against the corresponding decomposed function and thus guides the designer to the right design.

The features of the proposed system will be described with following a design case study as follows. We will consider a design of a gearbox. Figure 2 shows the GUI of the proposed system. The left-hand side viewport is for the shape detailer, the upper right-hand side viewport is for the function decomposer, and the lower right-hand side viewport is for the structural model for the design history. The designer starts the design by modeling the abstract shape of the gearbox as shown in the shape detailer. Its abstract shape is usually determined by the spatial restrictions. At the same time, the designer models the functional aspect of the gearbox as shown in the function decomposer. In this example, the designer assumes the the gearbox is largely composed of *housing*, *power transmission system*, and their relationship, *rotational support*, in terms of functions.

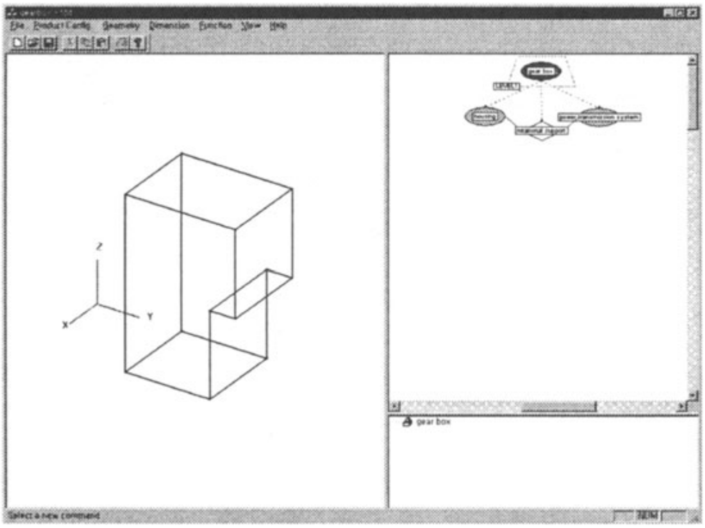

File Product Config Geometry Dimension Function View Help
gear box
housing
power transmission system
rotational support
Z
X
Y
gear box
Select a new command

Figure 2. GUI of the proposed design process modeler

Then, the abstract geometry of the power transmission system is specified in the shape detailer while the rotational support is divided into the related joints as shown in Figure 3. This decomposition and detailing process goes on resulting in the design process shown in Figure 4.

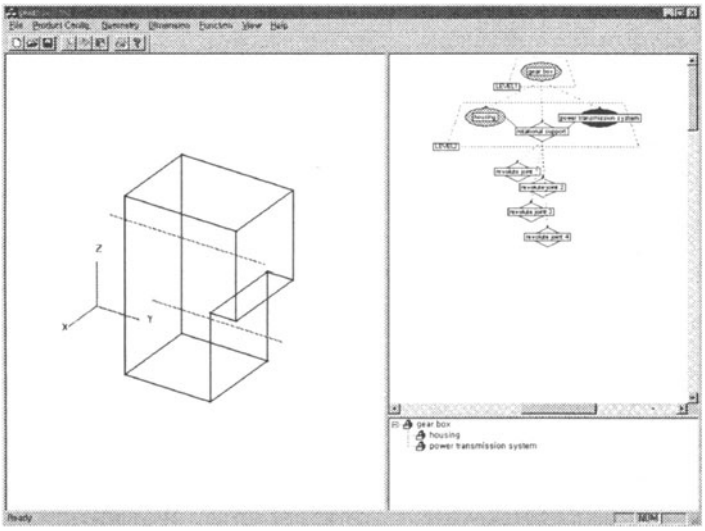

File Product Config. Geometry Dimensions Functions View Help
Z
Y
X
gear box
LEVEL1
housing
power transmission system
rotational support
LEVEL2
gear box
housing
power transmission system
Ready

Figure 3. Specification of abstract geometry of power transmission system and decomposition of the relationship

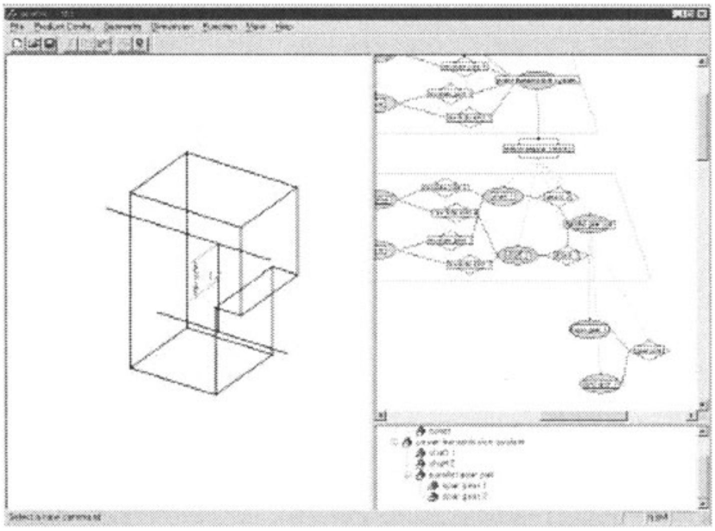


Figure 4. Design process of gearbox

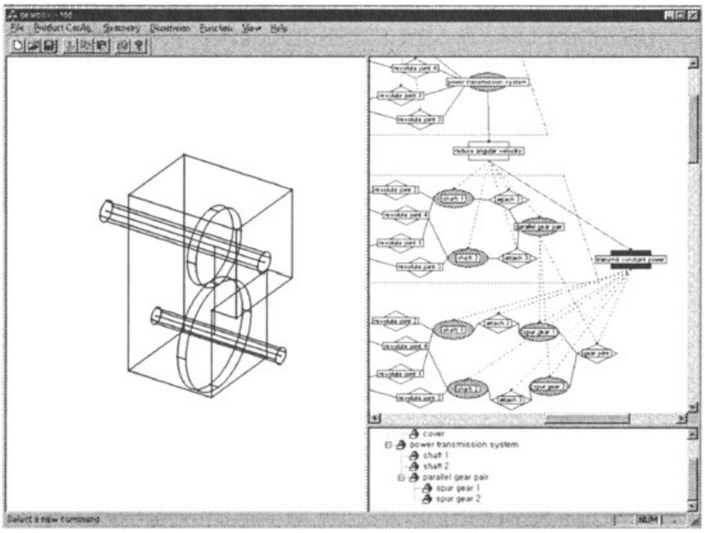


Figure 5. Evaluation of transmission constant power function

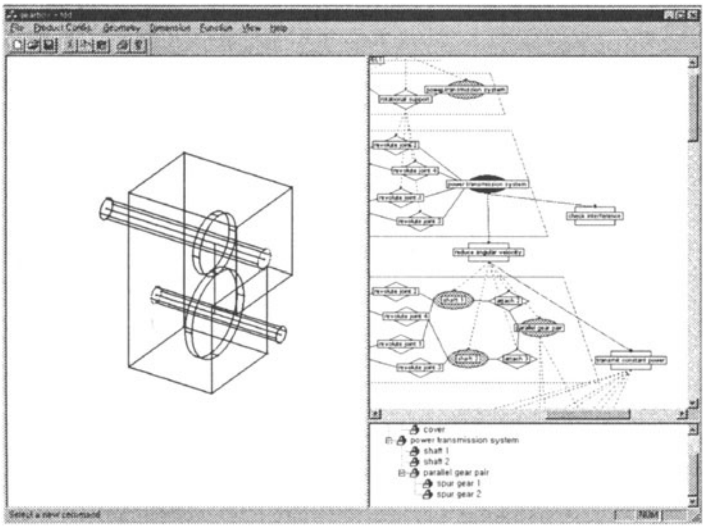


Figure 6. Redesign of gearbox by activating the design process

As mentioned earlier, when a function is defined in the proposed system, the evaluator for the function is also defined. Thus we can invoke the evaluator of the function *power transmission system* to check whether the involved parts can endure the loads corresponding to the desired transmission power. According to the evaluation result, the related parts are converted into solids as shown in Figure 5. Another function evaluator *check interference* can be invoked to check the interference between parts in this stage. Then we can also activate the whole design process from the beginning and get an updated design as shown in Figure 6. This is possible because all the steps taken in the design process for designing the gearbox are stored by the proposed design process modeler.

3. CASE-BASED REASONING FOR CONCEPTUAL DESIGN

When human designers are assigned a design task, they always review the existing designs for the similar tasks and generate many design alternatives by combining the relevant elements of the existing design cases. As we know, there is no general software system which recommends the possible complete designs for given desired tasks or functions by reviewing the existing designs for the similar tasks. However, there have been many research efforts to provide the computer with the case-based reasoning[7-8] capability for the limited scope of the design problem. Here we will concentrate on a case-based approach to the conceptual design of mechanisms[9-13] for function generation and motion transmission only in the functional point of view. Simply speaking, the system for the conceptual design system in this work recommends all the possible mechanisms for a given task by combining the component mechanisms existing in the existing mechanisms in a systematic way. The key idea in the proposed approach is that the whole design concept (complete mechanism) or sub-concepts (component mechanisms) originated from different design cases can be imported and merged to generate a various new design concepts that satisfy the given functional requirement. Thus the essential task would be to store the every sub-concepts as well as the whole concept of the design cases in the computer and to search for their right combinations to result the given input and output motion of the desired mechanism.

The design concept of a mechanism can be identified via individual primitive mechanisms constituting the mechanism. The individual functions of the primitive mechanisms account for the underlying design concept in the functional aspect of the mechanism. Not only the individual primitive mechanisms constituting the mechanism but also all possible concatenated chains can be a distinct function generator. Thus a new conceptual building block can be derived by the combination of two or more primitive mechanisms belonging to the mechanism.

For example, consider a film clawing mechanism illustrated in Figure 7(a). The mechanism transforms an input motion R_0 into two output motions: reciprocating translation (rT) and oscillating shaft rotation (oR_0), and transmit them to the output link (claw). Two cams transform the input rotation into two separate reciprocating motions. Then a slider-crank transforms reciprocating motion into oscillating motion. Thus the motion transmission of this mechanism can be illustrated in a directed graph as in Figure 7(b).

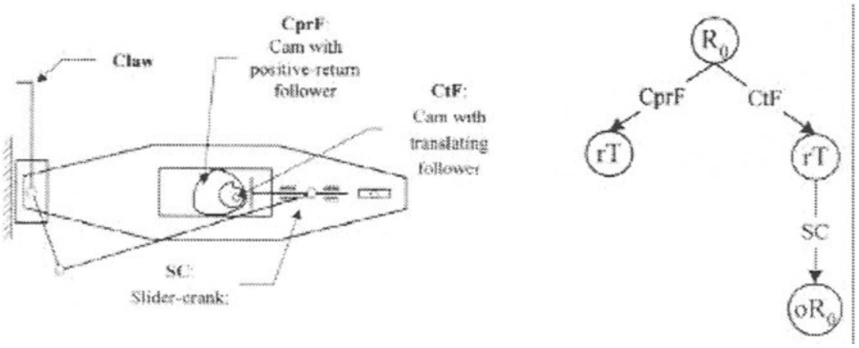

Claw
CprF Cam with positive return follower
CtF: Cam with translating follower
SC: Slider-crank
R0
CprF
CtF
rT
rT
SC
oR0

Figure 7. Kinematic diagram and abstract representation of a film clawing mechanism

From the design case illustrated in Figure 7(b), the whole design concepts and its sub-concepts can be derived as in Figure 8. As more design cases are available, the repertoire of the design concepts is expanded. Note that each concept has input and output motion(s). These sub-concepts are the building blocks to be used for synthesizing the new mechanism. Two concepts or mechanisms can be combined if one's output motion is the same as the other one's input motion, or vice versa. This combination is applied recursively until the right combinations of the mechanisms are derived. When the input motion of the first mechanism in the chain is the same as the desired input motion of the design task and the output motion of the last mechanism is the same as the output motion of the design task, the chain of the mechanisms is the right one. Figure 9 illustrates that a desired mechanism with input motion M_i and output motion M_o is obtained from two primitive mechanism p_i and p_o.

Among the right combination of the mechanisms, some of them may not satisfy geometric constraints involving the direction and position of the input and output of the desired mechanism. To generate only the spatially satisfactory mechanisms, the spatial knowledge of the primitive mechanisms should be prepared in advance because the overall spatial configuration of the whole mechanism is determined by the individual configurations of the constituent primitive mechanisms. The spatial configuration of a primitive mechanism is composed of orientation configuration, direction configuration, and position configuration. The direction configuration is the relative direction of the input and the output motions, the position direction configuration addresses the relative positional change between input and output point, and the orientation configuration refers to the layout of a primitive mechanism in a qualitative sense. From these configurations of the primitive mechanisms, the configuration of the chain of them can be derived and checked against the original requirement.

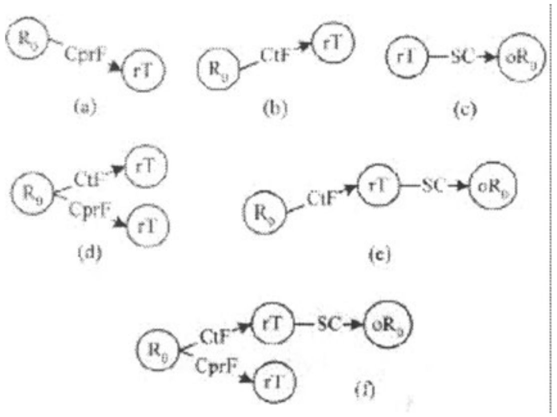


Figure 8. Extracted sub-concepts (sub-mechanisms) of a film clawing mechanism

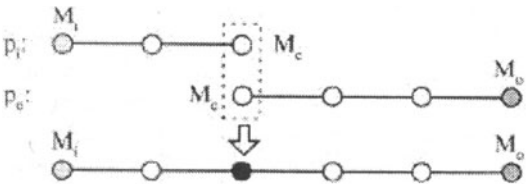
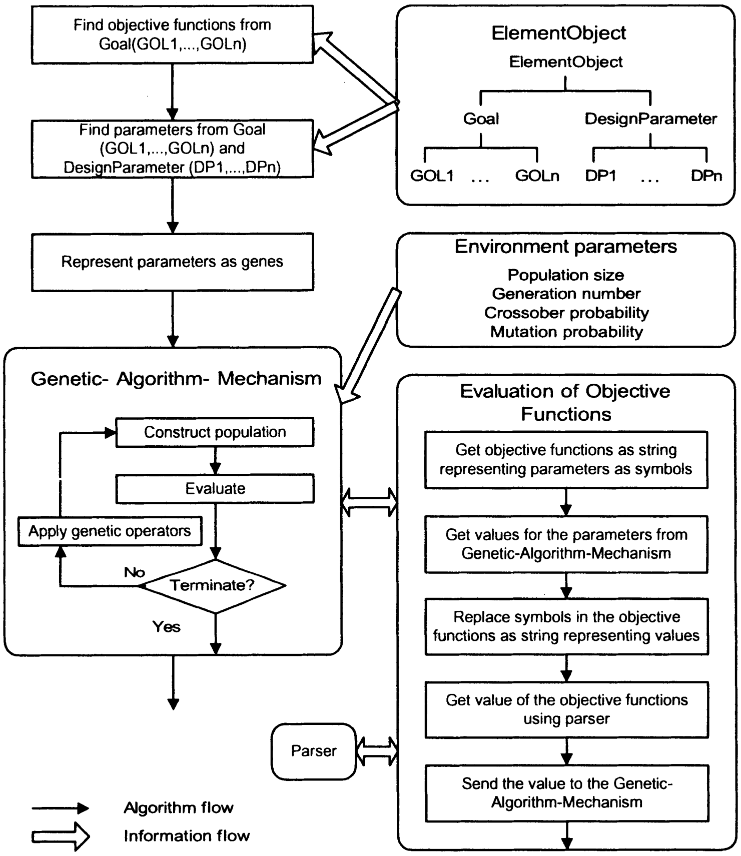


Figure 9. Combination of two compatible partial-matching cases

4. EVOLUTIONARY DESIGN WITH DEFINING DESIGN PARAMETERS AND OBJECTIVE FUNCTIONS

A design process is inherently iterative. In this approach, the iterative design process for improving a design is simulated using a genetic engine[14,15] as illustrated in Figure 10. As can be noted from Figure 10, the designers should be able to define the design parameters and express the goals (objective functions) in terms of the design parameters as they proceed along the design process.

Since a design parameter belongs to a part, it can be defined when the part has been or is being defined. In this approach, each part, each subassembly, and the whole assembly are stored in the entity called *Element Object* (EO). The element objects are created as the design proceeds or already exist for the standard parts such as a gear, shaft, and so on. Thus the design parameters are defined when the element object is created.



Find objective functions from Goal(GOL1,...,GOLn)
Find parameters from Goal (GOL1,...,GOLn) and DesignParameter (DP1,...,DPn)
Represent parameters as genes
ElementObject
ElementObject
Goal
DesignParameter
GOL1 ... GOLn
DP1 ... DPn
Environment parameters
Population size
Generation number
Crossober probability
Mutation probability
Genetic- Algorithm- Mechanism
Construct population
Evaluate
Apply genetic operators
No
Terminate?
Yes
Evaluation of Objective Functions
Get objective functions as string representing parameters as symbols
Get values for the parameters from Genetic-Algorithm-Mechanism
Replace symbols in the objective functions as string representing values
Get value of the objective functions using parser
Send the value to the Genetic-Algorithm-Mechanism
Parser
Algorithm flow
Information flow

Figure 10. Procedure for evolutionary design using genetic engine

We will illustrate how the proposed system in this work enables evolutionary design system using an example. Figure 11 shows how the element objects are declared as the designer carries out the design of a gearbox by the design information processor proposed in this work. In this example, the gearbox named reducer is divided into five EO's: *Input-Assy, Output-Assy, Housing, Requirement, and Team.* This division is completely up to the designer's decision. *Input-Assy*, *Output-Assy*, and *Housing* represent the input shaft subassembly, output shaft subassembly, and the casing of the gearbox respectively. Requirement and team represent the information on the design requirement and the members in the team respectively. The information on the team members are used to notify the design change to the relevant members for the collaborative design. The input shaft subassembly, *Input-Assy*, is again divided into *Input-shaft, Input-Gear, Bearing 1, and Bearing 2* by the designer. Similarly, *Output-Assy* is divided into *Output-Shaft, Out-Put-Gear, Bearing 3, and Bearing 4.* When EO's for these component are created (or modified from the existing EO's for the standard parts), design parameters are defined as in Figure 11. Figure 12 shows the skeleton model of the reducer generated by the designer using a geometry processor. The geometry processor runs simultaneously with the design information processor shown in Figure 11.

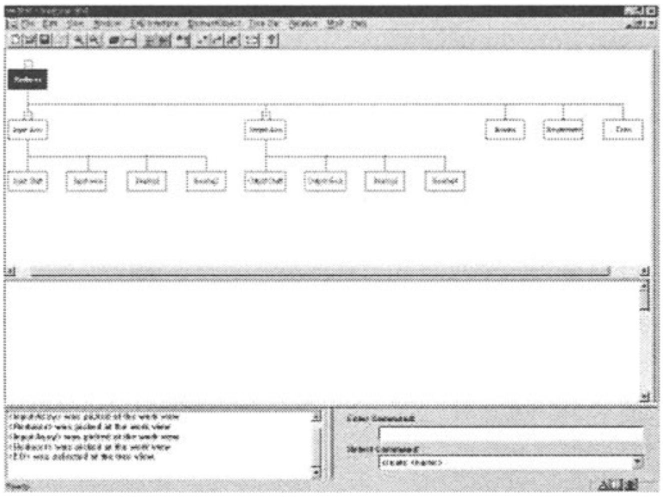


Figure 11. Process of designing a gearbox in the proposed design information processor

Figure 13 shows the screen display when the design parameters of a ball bearing is defined by retrieving its EO (standard EO in this case). Note that the relationship of the design parameters are also specified when they are defined. Figure 14 shows the design parameters of the EO of the *Reducer.* The design parameters are chosen by the designer based on his design intent; *a* is the number of the tooth of the input gear, *m* is its module, *b* is the number of the tooth of the output gear, *i* is the reduction ratio, and *l* is the offset distance between input and output shaft. It can be noted from the column *InducedEO* that *a* was already defined when *Input-Gear* was defined. Similarly, *b* was also defined when *Output-Gear* was defined. Figure 14 also shows that *i*, *l*, and *m* are specified as fixed values while *a* and *b* are the variables to be optimized in the evolutionary design to be followed. The choice of the design variables is also up to the designer and represents his design intent.

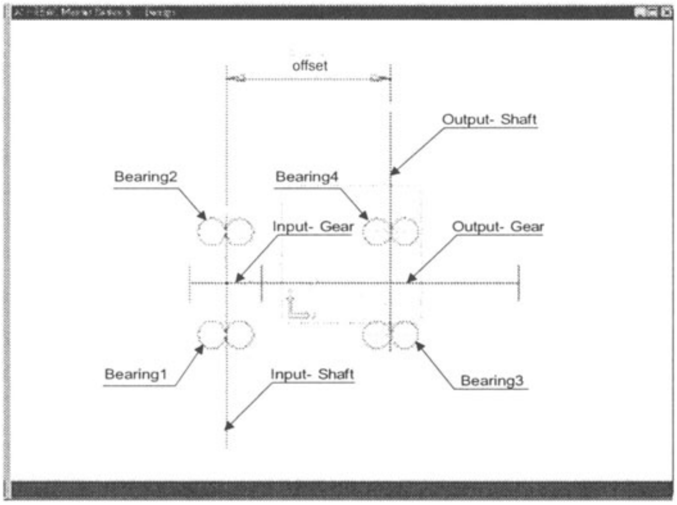


Figure 12. Skeleton model of a gearbox generated by geometry processor

EO	Parameter	Value	Description	*InducedEO*	Parameter2 [1)]
Requirement	*power*	10	power[kW]	*Requirement*	*power*
	rpm	500	rpm [rpm]	*Requirement*	*rpm*
	ratio	3.5	reduction ratio	*Requirement*	*ratio*
	offset	237.5	offset	*Requirement*	*offset*
Reducer	*a*	(21)	Teeth number	*Input-Gear*	*Teeth*
	b	(74)	Teeth number	*Output-Gear*	*Teeth*
	m	5	module	*Input-Gear*	*Module*
	i	3.5	reduction ratio	*Requirement*	*ratio*
	l	237.5	offset	*Requirement*	*offset*
Input-Gear	*Module*	5	module	*Input-Gear*	*Module*
	power	10	power[kW]	*Requirement*	*power*
	rpm	500	rpm [rpm]	*Requirement*	*rpm*
	Teeth	(21)	Teeth number	*Input-Gear*	*Teeth*
	pitch	(105)	ref. pitch dia.	*Input-Gear*	*pitch*
	width	(55)	teeth width	*Input-Gear*	*width*
	Torque	(191)	torque[N.m]	*Input-Gear*	*Torque*

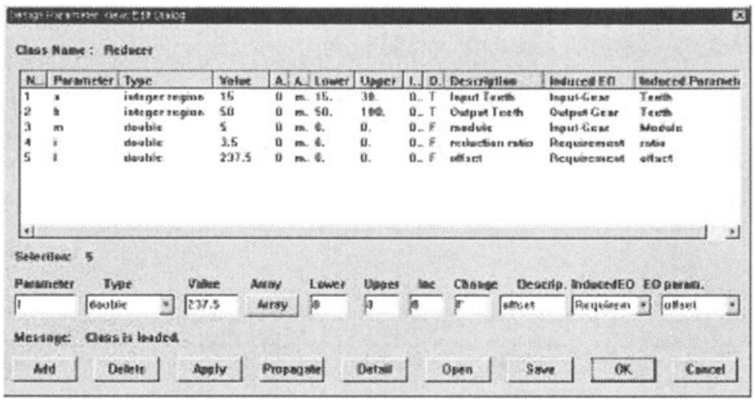
Input-Shaft	*Torque*	(191)	torque [N.m]	*Input-Gear*	*Torque*
	mindia	(40)	min. diameter	*Input-Shaft*	*mindia*
Output-Gear	*Module*	5	module	*Output-Gear*	*Module*
	power	10	power[kW]	*Requirement*	*power*
	rpm	500	rpm [rpm]	*Requirement*	*rpm*
	Teeth	(21)	Teeth number	*Output-Gear*	*Teeth*
	pitch	(105)	ref. pitch dia.	*Output-Gear*	*pitch*
	width	(45)	teeth width	*Output-Gear*	*width*
	Torque	(673)	torque [N.m]	*Output-Gear*	*Torque*
Output-Shaft	*Torque*	(673)	torque [N.m]	*Output-Gear*	*Torque*
	mindia	(55)	min. diameter	*Output-Shaft*	*mindia*

Note:

1) Parameter2 is the parameter that is used in *InducedEO*

2) The values in the parenthesis at the *Value* column may be determined during design process.

Figure 13. Definition of design parameters of ball bearing



Class Name : Reducer

N.	Parameter	Type	Value	A.	A.	Lower	Upper	I.	D.	Description	Induced EO	Induced Paramet
1	a	integer region	15	0	m.	15.	30.	0..	T	Input Teeth	Input-Gear	Teeth
2	b	integer region	50	0	m.	50.	100.	0..	T	Output Teeth	Output Gear	Teeth
3	m	double	5	0	m.	0.	0.	0..	F	module	Input-Gear	Module
4	i	double	3.5	0	m.	0.	0.	0..	F	reduction ratio	Requirement	ratio
5	l	double	237.5	0	m.	0.	0.	0..	F	offset	Requirement	offset

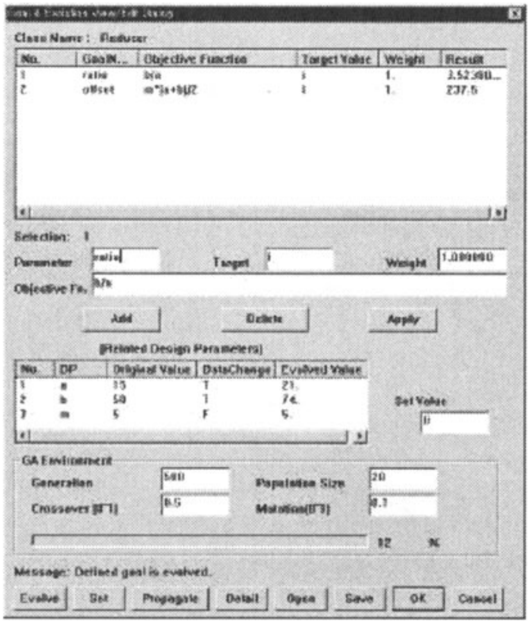
Selection: 5

Parameter	Type	Value	Array	Lower	Upper	Inc	Change	Descrip.	InducedEO	EO param.
l	double	237.5	Array	0	0	0	F	offset	Requirem	offset

Message: Class is loaded.

Figure 14. Screen display of design parameters of *Reducer*

Figure 15 shows how the objective functions are defined for the evolutionary design. Each objective function has the target value and the weight factor to be used. Figure 16 shows the resulting design generated by the geometry processor from the design parameters determined by the design information processor.



Class Name : Reducer

No.	GoalN...	Objective Function	Target Value	Weight	Result
1	ratio	b/a	i	1.	3.52380...
2	offset	m*(a+b)/2	i	1.	237.5

Selection: 1

Parameter: ratio Target: i Weight: 1.000000

Objective Fn.: b/a

[Related Design Parameters]

No.	DP	Original Value	DataChange	Evolved Value
1	a	15	T	21.
2	b	50	T	74.
3	m	5	F	5.

Set Value: 0

GA Environment

Generation: 500 Population Size: 20

Crossover(0~1): 0.5 Mutation(0~1): 0.1

12 %

Message: Defined goal is evolved.

Figure 15. Design goals constructed for *Reducer*

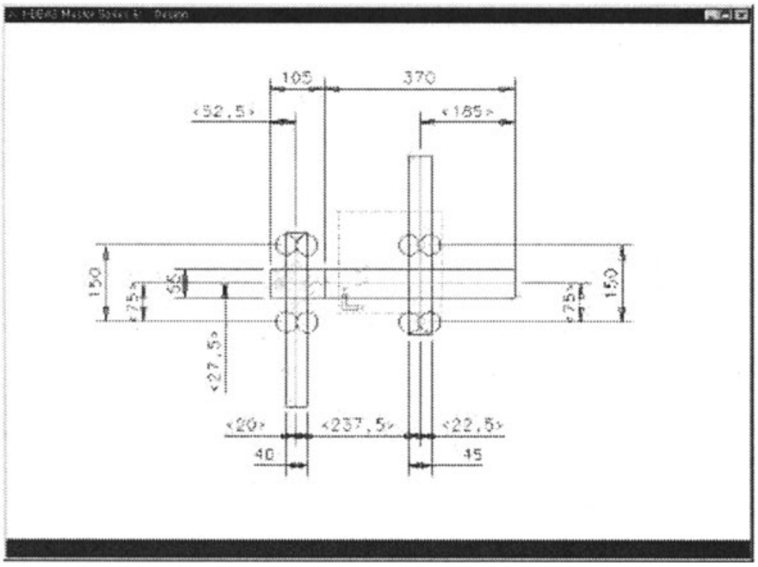


Figure 16. Reducer modeled by geometry processor

5. CONCLUSION

Three capabilities of a CAD system to make it a real design tool are proposed and their implementation is described. For the proposed capabilities are to be used in the real design practice, they should be improved as follows. First, the repertoire of functions in the function decomposer should be expanded and should have a structure to be expanded easily in the course of design practice. Second, the case-based reasoning should have more applications other than kinematic linkage synthesis. Third, the repertoire of the element objects used in the evolutionary design should be expanded and have a structure to be expanded easily in the course of design practice.

Three capabilities mentioned above are developed separately even though their goal is identical, i.e. the whole design process is modeled and accumulated with the design tasks, and a large portion of the design process is guided by a computer if possible. It is desirable to embody them in a CAD system such that the capabilities are used in collaboration. This is left until each capability is completely understood and implemented.

Reference

[1]. Mantyla, M., "Modeling system for top-down design of assembled products", IBM Journal of Research and Development

[2]. Gui, J.-K. and Mantyla, M., "Functional understanding of assembly modelling," Computer-Aided Design, Vol. 26, No. 6, pp.435-451, 1994.

[3]. Horvath, I., Themesz, V. and Bagoly, Z., "Conceptual design with functionally and morphologically parametrized feature objects," Proceedings of the Computers in Engineering Conference and the Engineering Database Symposium ASME, pp. 507-516, 1995

[4]. Gortie, S. R. and Sriram, R. D., "From symbol to form: a framework for conceptual design," Computer-Aided Design, Vol. 28, No. 11, pp. 853-870, 1996.

[5]. Szykman, S. and Kim, J. H., "Interactive Exploration and Optimization in Assembly Design," Proceedings of 1996 ASME Int. Conf. On Design Automation, Irvine, CA, USA, 1996.

[6]. Pahng, F., Senin, N. and Wallace, D., "Distribution modeling and evaluation of product design problems," Computer-Aided Design, Vol. 30, No.6, pp.411-423, 1998.

[7]. Kolodner, J. L., Case-Based Reasoning, 2ed., Morgan Kaufmann, Inc., 1993

[8]. Gebhardt, Friedrich, Voß, Angi, Gräther, Wolfgang and Schmidt-Belz, Barbara, Reasoning with Complex Cases, Kluwer Academic Publishers, Nowell, Massacusetts, 1997

[9]. Kota, Sridhar and Chiou, Shean-Juinn, "Conceptual Design of Mechanisms Based on Computational Synthesis and Simulation of Kinematic Building Blocks," Research in Engineering Design, Vol.4, No.2, pp.75-87, 1992.

[10]. Chakrabarti, A and Bligh, T. P., "An Approach to Functional Synthesis of Solutions in Mechanical Conceptual Design. Part I: Introduction and Knowledge Representation," Research in Engineering Design, Vol.6, No.3, pp.127-141, 1994.

[11]. Chakrabarti, A and Bligh, T. P., "An Approach to Functional Synthesis of Solutions in Mechanical Conceptual Design. Part II: Kind Synthesis," Research in Engineering Design, Vol.8, No.1, pp.52-62, 1996.

[12]. Issa, Ghassan, Shen, Stewart and Chew, Meng Sang, "Using Analogical Reasoning for Mechanism Design," IEEE Expert, Vol.9, pp.60-69, June 1994.

[13]. Narasimhan, S., Sycara, Katia P. and Navin-Chandra, D., "Representation and synthesis of non-monotonic mechanical devices," in Issues and Applications of Case-Based Reasoning in Design, eds., Mary Lou Maher and Pearl Pu, Lawrence Erlbaum Associates, Inc., Publishers, pp.187-219, 1997.

[14]. Gero, J. S. and Kazakov, V., "Evolving design genes in space layout problems," Artificial Intelligence in Engineering, Vol.12, No.3, pp.163-176, 1998.

[15]. Gero, J. S., Kazakov, V. and Schnier, T., "Genetic engineering and design problems," in Evolutionary Algorithms in Engineering Applications, eds., D.Dasgupta and Z.Michalewicz, Springer Verlag, Berlin, pp.47-68, 1997