Artificial Intelligence for Engineering Design, Analysis and Manufacturing (1996), *10*, 313–331. Printed in the USA.
Copyright © 1996 Cambridge University Press 0890-0604/96 $11.00 + .10

An approach to functional synthesis of mechanical design concepts: Theory, applications, and emerging research issues

AMARESH CHAKRABARTI AND THOMAS P. BLIGH
Engineering Design Centre, Cambridge University Engineering Department, Trumpington Street, Cambridge CB2 1PZ, United Kingdom

(RECEIVED July 31, 1995; ACCEPTED January 15, 1996)

Abstract

Conceptual design is an early stage in the design process that involves the generation of solution concepts to satisfy the functional requirements of a design problem. Usually, there are many solutions to a design problem; therefore, there is scope for producing improved designs if one could explore a solution space larger than is presently possible. An approach would be to use the computer to synthesize a wide variety of concepts for a given problem, and allow designers to explore these before developing the most promising ones. Adopting a research approach based on developing basic representations, knowledge base, and reasoning procedures adequate for synthesizing concepts of existing devices and mechanisms, a computer program for synthesis of solutions to a class of mechanical design problems has been developed. For a given design problem, the program can produce an exhaustive set of solution concepts, in terms of their topological and spatial configurations, which can then be explored by designers. The program has been tested in two ways: (1) by comparing the candidate solutions produced by the program with those produced by designers in two real design case studies, and (2) by using three experienced designers to evaluate the solutions, generated by the program, for their novelty and usefulness. This paper presents the theoretical basis, research method, the theory and implementation of the synthesis approach. Also, the results of the above case studies and evaluations, and a discussion of further issues highlighted by the evaluations are presented.

Keywords: Conceptual Design; Functional Modelling; Functional Reasoning; Functional Synthesis; Mechanical Design.

1. OBJECTIVE

The objective of this paper is to establish the importance of the conceptual design stage in the design process, identify relevant areas of research within this stage, propose one method to do research in these areas, present an overview of an approach developed to support concept generation, present some results of testing this approach, and indicate further research directions, especially for supporting exploration of concepts generated using this approach.

Conceptual design is that stage of the design process where concepts of solutions are developed to meet the functional requirements of the design problem (Pahl & Beitz, 1984). Conceptual design, being one of the early stages of design, is characterized by information that is often imprecise, inadequate, and unreliable. Apart from the observation that the recognition and generation of functional requirements and the generation of solutions are highly coupled in this stage, there is little understanding as to how this is done, and consequently little support is available. Notwithstanding these difficulties, it is at this early stage that substantial costs are committed. As pointed out by Berliner and Brimson (1988), on average as high as 80% of the cost of a product over its total life cycle is committed by the conceptual stage of the design process.

The challenge is, how can we support designers so as to increase their chances of producing the best possible concepts? The key to answering this question is that there is often not one, but a multitude of possible alternative solu-

Reprint requests to: Dr. Amaresh Chakrabarti, Engineering Design Centre, Cambridge University Engineering Department, Trumpington Street, Cambridge CB2 1PZ, United Kingdom. Phone +44-1223-332828; Fax: +44-1223-332662.

tions to a given design problem. Therefore, if a designer can be supported and encouraged to generate and explore a wider range of solution alternatives using a wider range of evaluation criteria, this would increase their chances of producing better designs.

Usually, however, only a few design alternatives are considered, and using only a few evaluation criteria. There can be several reasons for this. Often designers are not aware of the existence of potential solutions in a different domain. This is particularly true of novice designers whose repertoire of designs is limited due to lack of experience. Designers can be biased toward using certain kinds of solutions, perhaps because they have used them before. Evaluation criteria often emerge from the observation of key positive and negative features of the design alternatives at hand. This means that the more and widely varied the considered solution alternatives are, the wider and richer the criteria for their evaluation should be. However, if none of these difficulties existed, designers would still not be able to consider more than a few alternatives without being supported by an enhanced information processing capability. This is because as each design is detailed, the information generated around it grows quickly, making it quite impossible to explore more than but a few alternatives. This is where computers, with their potential for information processing, could make a difference.

One possible route is to devise a computational framework where the designer could be presented with a wider range of ideas than is possible at present, and could be supported to evaluate and modify them before homing in on the most promising ones for further development. Moreover, conceptual design, as mentioned earlier, is when information is most imprecise, and often just is not available, leaving the designer with little choice but to make assumptions to proceed. These assumptions permeate into the design, and are often accepted by subteams who are not necessarily in a position to judge their accuracy or validity, or indeed they may not realize that these are no more than *best guesses*, and consequently the basis of each decision may be lost. In complex designs, the task of keeping track of all aspects that are influenced by each assumption is a formidable one. When the assumptions are improved or changed, by more detailed and improved knowledge, every part of the design that is influenced should be changed in an intelligent way. Unless we have an overall computational framework that supports design right through all the stages of the design process, from requirement identification through conceptua' and embodiment design to the detailing, such evaluations and back-propagations cannot be supported. Hence, the need for an overall support framework, within which conceptual design support is a part.

So there are two central objectives: (1) to support the generation of a wider range of ideas than is possible at present, and (2) to support exploration (evaluation and modification) of these ideas so as to fulfill the emerging functional requirements.

The main bulk of the results presented in this paper will deal with the first objective. The approach will be explained using a number of generic examples. The evaluation of the approach is done using in-house case studies and hands-on experiments with the program by experienced designers, the results of which are used to identify key issues for further research. Supporting exploration is part of further research and will be discussed in Sections 6 and 7.

2. RESEARCH APPROACH

The research approach is based on the assumption that new concepts can be generated by combining, and/or adapting, existing elements so as to satisfy new functionality. The reason is that a wide variety of designs do seem to have common elements, and therefore, if these elements could be distilled, and combined in different ways, this should generate new designs. We start our research by looking into a range of known design problems and their known design solutions (see steps in Fig. 1) for commonalities among them. What things are common across the given set of design problems, and what things are common across these design solutions? Are there some general ways of representing them? Can we perhaps identify the common elements across these designs that could form a collective knowledge base? Can we then, develop reasoning procedures that would use this knowledge and representation to generate solutions to these known problems? If this could be done, we could then compare the outcome of the procedures with the solutions already known to us. At the very least, do they generate the solutions that we already know? But, more importantly, do they generate solutions that are new? If this could be done, we would be able to provide designers with a wider range of ideas than is now possible, and allow them to manipulate and evaluate these.

3. RESULTS

In this section, each of the above steps, shown in Figure 1, will be illustrated with results.

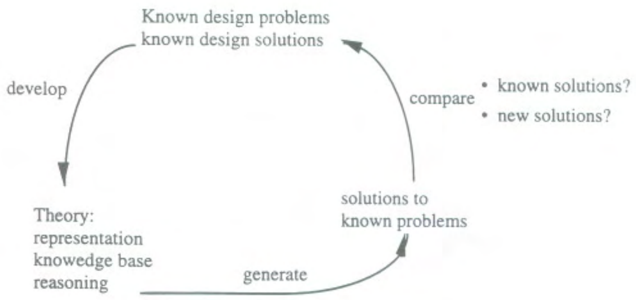


Fig. 1. Research approach.

3.1. Develop theory from known design problems and solutions

We start off with a set of existing devices and mechanisms, of which three are shown in Figure 2 for illustration. One is a door latch, one a paper punch, and the other a scotch-yoke mechanism. What is common among these devices? In terms of their functions, each of them requires some input, and produces some output. The number of inputs or outputs can be more than one (e.g., the two outputs of the paper punch). Each of these inputs and outputs has some characteristics that may change with time. For instance, as the handle of the door latch is pressed down, it moves down, causing the output point (the wedge) to retract into the casing. If this is continued, at some point the handle stops moving down, and with it the retraction motion of the wedge also stops. If we now let the handle go, it moves in the opposite direction to the original position, and simultaneously the wedge moves back to its original position. Now, this is a temporal sequence of activities defined by changes in the characteristics of the inputs and outputs of the device with time. The characteristics of the inputs and outputs, in this case, include kinds such as force or torque, or linear and angular motion, the directions of their action (e.g., up, down, or sideways), their magnitudes, and their positions. For further details, see Chakrabarti (1991) and Chakrabarti and Bligh (1994*a*).

Let us, for the discussion that follows, not consider the temporal characteristics of the inputs and outputs, and focus on them at an instant of time. What parameters are sufficient to describe the functionality of each of the above problems? We could have an adequate representation of the function of a design in terms of a set of inputs and outputs, each of which has a kind, direction (positive or negative directions along the *i*, *j*, *k* axes), magnitude, and position.

Now the question is, is there anything common between the structures of these designs? The first thing we notice is that each design is composed of a number of elements, which are connected in certain ways. We also notice that some of these elements function in the same way, although their embodiment might look quite different. For instance, the handle of the door latch is such that when pushed on the input end, it rotates at the other. In other words, it has a single input and output, one of which is a force (or translation) and the other is a torque (or rotation). Although visually different, functionally this element is similar to the top part of the paper punch which, when pushed down on its input edge produces a rotation at its pivot end. One could also notice that if rotation is taken as a pseudo-vector (using the right-hand rule for instance, where curled fingers point in the direction of the rotation and the thumb points in the vector direction), the spatial configurations of the input and output of this element have a definite relationship: they are orthogonal and nonintersecting to each other. Interestingly, the crank of the scotch-yoke mechanism is not dissimilar either, where the input and output kinds are just reversed. Another interesting feature is that the directions of the inputs and outputs (while always being orthogonal and nonintersecting to each other) depend on the direction in which the element is laid out in space. We could observe similar functional similarities between the wedge assembly (which can move back and forth), the two punches, and the output ele-

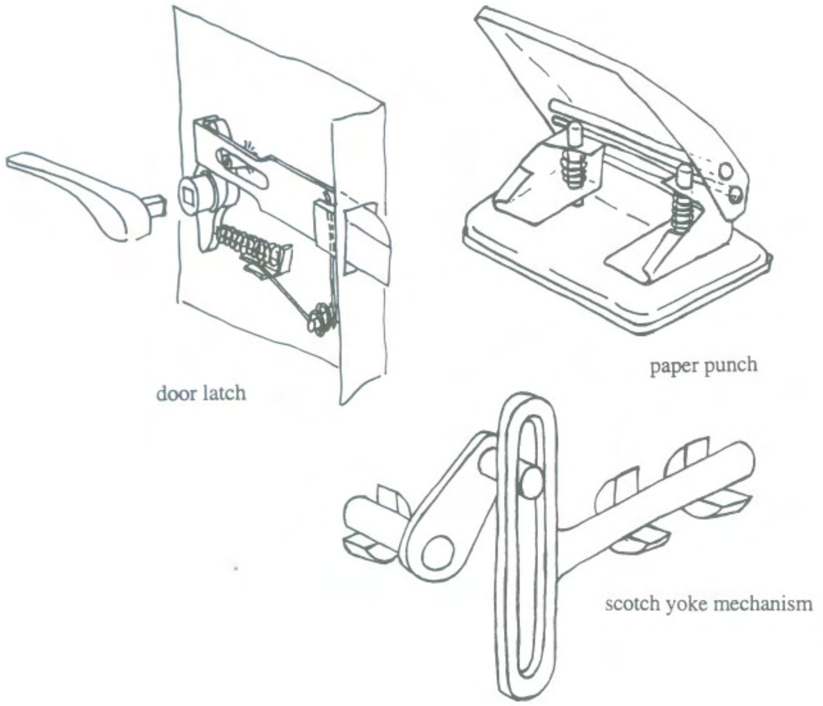


Fig. 2. Devices and mechanisms for analysis.

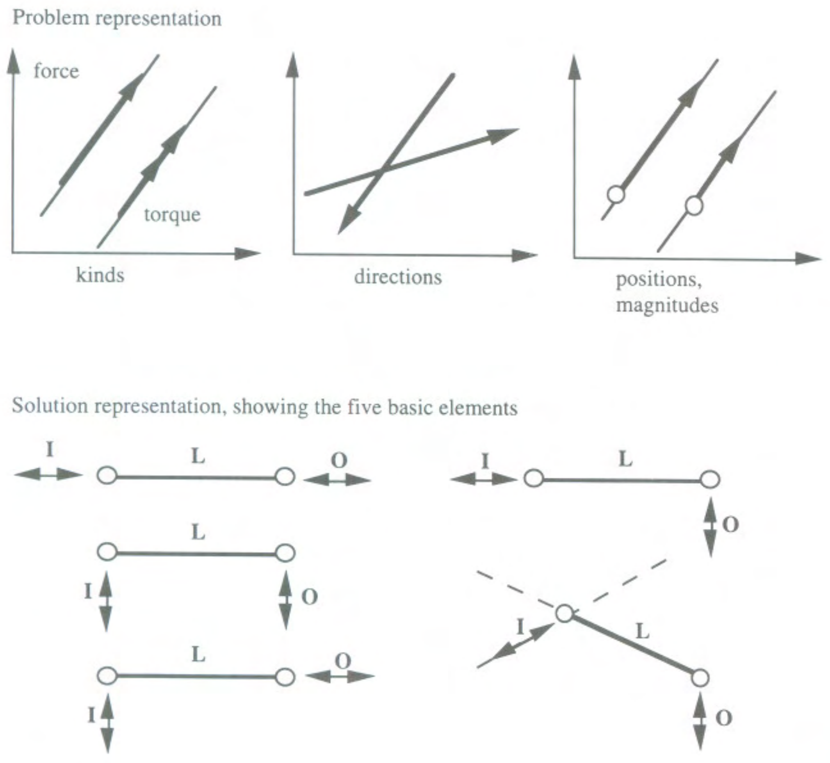


Fig. 3. Representation for design problems and solutions.

ment of the scotch-yoke mechanism. Also, their inputs and outputs are of the same kinds (translation) and parallel to each other. What we also notice is that the way these elements are connected to form the device allows them to transfer their inputs and outputs. In the door latch, for instance, the output rotation from the handle is transferred to the input of the cam, which produces a translation at its output. This translation is taken at the top portion of the wedge assembly, which is transferred to its output to produce the desired effect. In other words, if the function of each element is known in terms of its inputs and outputs, and if they are appropriately connected so that the inputs and outputs match at the connections, then they together provide a causal account of the internal functioning of the device.[1]

Could we now develop a representation from all these observations? Figure 3 presents such a representation. A design problem is represented by its functions, where a function is represented by a number of inputs and outputs, each having a kind, direction, and magnitude. Design solutions are described as combinations of a set of functional elements (such as a lever, of which the door latch handle is an example embodiment). Each such element is defined as one of five basic element types,[2] or combinations thereof. These elements are distinguished by the spatial relationships between their inputs, outputs, and the spatial separation between them.

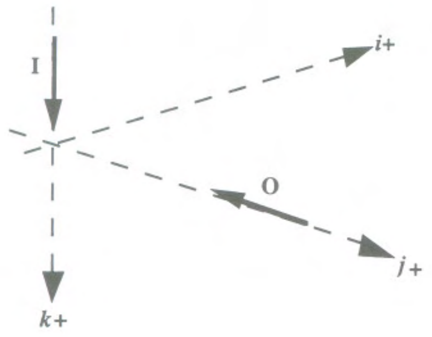
Using this representation, for example, we could now present the function of the door latch as a vertical downward force input to be transformed into a horizontal leftward translational output, as shown in Figure 4a. The door latch solution, shown before, could be represented using the functional elements as (see Fig. 4b) a combination of a lever taking the input, transferring its output, a torque, to a cam that in turn produces a translational output, to be transferred by two tie-rods to the desired output point.

What about the reasoning procedures? Assuming that we have a set of known functional elements that were distilled from a host of existing designs, how can we use these to generate these existing designs, and perhaps other new ones?

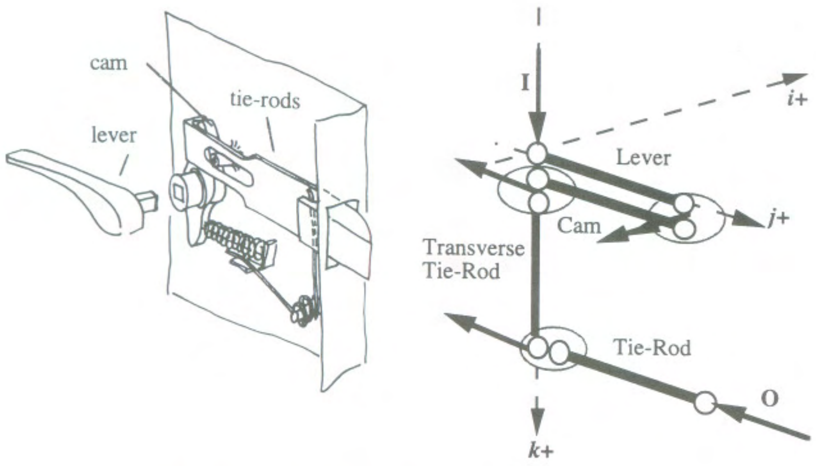
As an illustration of how this could be done, a single input output (SISO) synthesis algorithm, a simpler version of a reasoning procedure for the exhaustive generation of multiple input output designs, is presented in Figure 5. The general idea is that having arbitrarily chosen the maximum allowable number of elements to be used in a design solu-

[1] Even though we have chosen to illustrate these by elements having inputs and outputs with mechanical characteristics, this approach is not restricted only to such elements. Generally, an element could have inputs and outputs with any characteristics.

[2] These element types are basic in the sense that these are the only possible distinct spatial relationships that an input vector **I**, an output vector **O**, and the length vector **L**, which denotes the spatial separation between **I** and **O**, could have (they can be coaxial, parallel, intersecting with **I** being coaxial with **L**, intersecting with **O** being coaxial with **L**, and skew). These would give rise to more elements when the kind of input and output are specified. For more details, see Chakrabarti and Bligh (1994*a*).



4a: Door latch problem representation



4b: Door latch concept and its representation

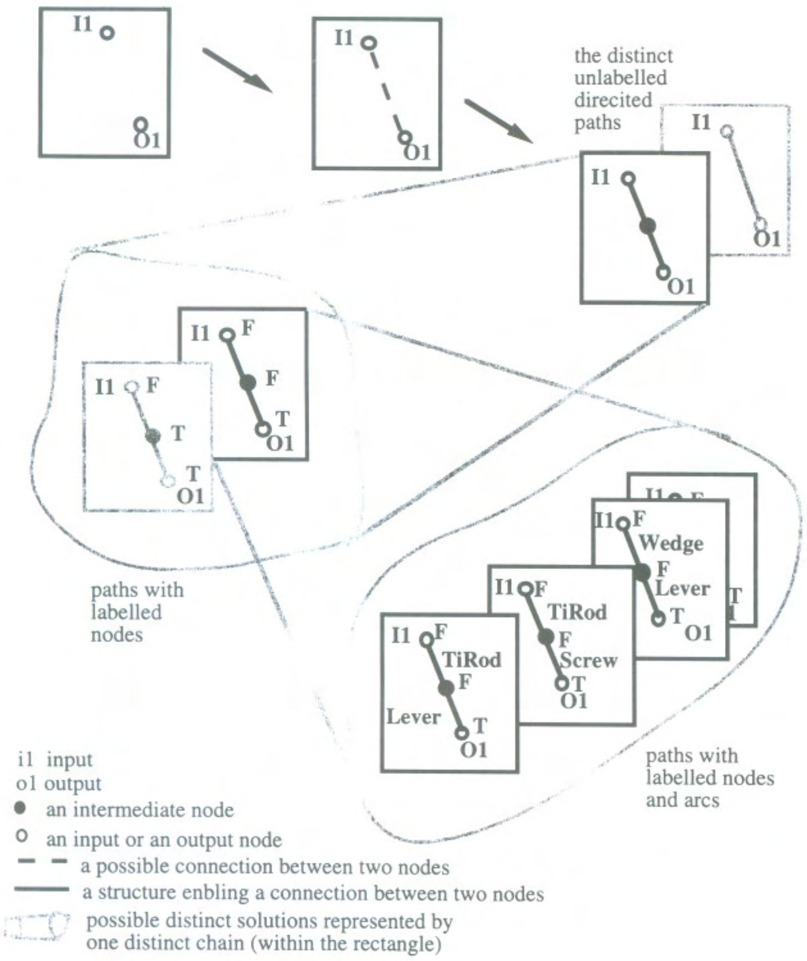
Fig. 4. Describing the door latch problem and an existing solution using the representation in Figure 3.

tion, the procedure generates each possible distinct unlabelled simple directed path.[3] For instance, if the maximum allowable number of elements is 2, the two possible paths are: with a single element connecting the input to the output, or with two elements in series. Each arc in such a graph is an element, and each node is an input output connection. If we now label each of the nodes in each path using the information about the functional requirements and the kinds of I/O that can be handled by the database of functional elements, we get a number of distinct paths with labelled nodes. For instance, if the system input should be a force *F*, and the system output a torque *T*, and if the elements in the database can only have forces and torques as inputs or outputs, the only two possible paths with labelled nodes are *F-F-T* and *F-T-T* (which means that this function could be achieved either by a force to force followed by a force to torque transformation, or by a force to torque followed by a torque to torque transformation). If we now label each arc with each possible alternative database element capable of doing the transformation indicated by the labelled nodes, the alternative solutions are found. For instance, in the *F-F-T* path, if the *F-F* transformation could be done in two ways, that is, using a wedge or using a tie-rod, and the *F-T* transformation could be done in two alternative ways, that is, by a screw or by a lever, then the overall function could be achieved by their possible combinations (see Fig. 5). For further details, see Chakrabarti (1991) and Chakrabarti and Bligh (1994*a*, 1996*a*, 1996*b*).

3.2. Generate solutions to known problems and compare with existing designs

What now is the result of applying the above theory (representation, knowledge base, and reasoning) to the problems with which we started? As an illustration let us generate solutions to the door latch problem. What happens when we give the single input–output door latch function described before as an input to a program based on the above theory?

[3] A graph in Graph theory (Reinschke, 1988) is a mathematical structure that consists of two kinds of sets that may be interpreted geometrically as nodes (or vertices) and as arcs (or edges) whose end points are nodes. A path is a sequence of arcs such that the initial node of the succeeding arc is the final node of the preceding arc. It is directed if each of its edges can be traversed in one direction only. A path is simple if one reaches going along the path from its initial to its final vertex no vertex more than once. It is unlabelled if its vertices and edges have not been labelled. See Reinschke (1988) for further details.



I1
O1
the distinct unlabelled direcited paths
F
T
paths with labelled nodes
Wedge
Lever
TiRod
Screw
paths with labelled nodes and arcs
i1 input
o1 output
● an intermediate node
o an input or an output node
a possible connection between two nodes
a structure enbling a connection between two nodes
possible distinct solutions represented by one distinct chain (within the rectangle)

Fig. 5. The SISO synthesis algorithm.

Does it generate at least the existing designs? Does it generate other novel, exciting ones?

This is what we did for each of the problems shown in Figure 2, and for other design problems. As an illustration of the results, using a maximum of 5 elements per solution and for a given database of 6 elements, the program generated 193 different designs[4] to the door latch problem, so as to have a tie-rod at the output end of each. It generated a number of alternative (stick-diagram like) spatial layouts for each of these solutions as well. One layout of one such design is shown in Figure 6. The left-hand diagram is computer generated, whereas the right-hand embodiment is conceived by us. It should be noted that each such spatial layout could, in principle, be developed into a number of alternative embodiments; that shown is just one, and not necessarily the best one at that. As we see, in this embodiment when the handle is pressed down, it presses down the wedge, which then moves down as well as to the left, thereby pushing the latch assembly leftward. The constituent elements are similar to those extracted from existing designs such as door latches and paper punches, but the combination is a different one, leading to a substantially different concept. This is precisely the assumption we mentioned in Section 2, that new designs are combinations or adaptations of existing designs.

[4] The exhaustive set of solutions, which can be synthesized by combining these 6 elements (a lever to transform a force into a torque, a lever to transform a torque into a force, a cam to transform a torque into a force, a wedge to transform a force into a force, a transverse tie-rod to transform a force into a force, and an axial tie-rod to transform a force into a force) so as to use at most 5 elements in each solution so generated, would contain 687 distinct solutions. This can be calculated using Eq. (5) given in Appendix A of Chakrabarti and Bligh (1996*a*). There would be 193 of these solutions, which would have a tie-rod as their output element.

4. IMPLEMENTATION OF THE SYNTHESIS APPROACH AND ITS EVALUATION

The synthesis approach described in Section 3 has been implemented using Common-LISP language on a LispWorks^R (Harlequin Limited, 1991) package, which is a LISP-based environment for developing knowledge-based tools. Our program is called FuncSION, which is an acronym for **Func**tional **S**ynthesizer for **I**nput **O**uput **N**etworks.

FuncSION has been evaluated in two ways. One was the use of an in-house project, called the Mobile Arm Support (MAS) project, carried out at the Cambridge University

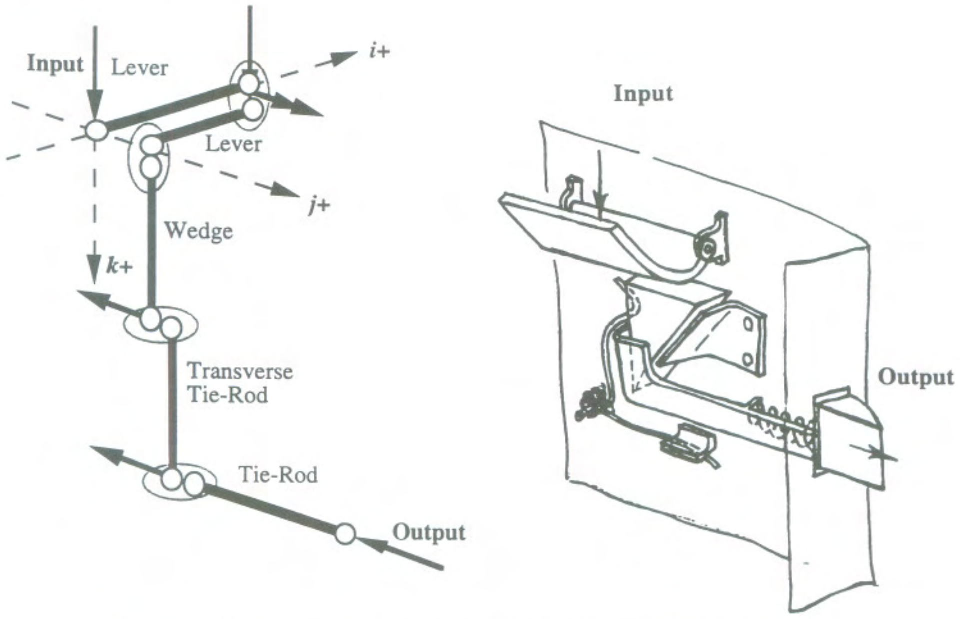


Fig. 6. One of the solutions generated by the program for the door latch problem.

Engineering Design Centre, as a case study. The other test was the evaluation of the concepts generated by FuncSION, by three experienced designers, for their novelty and usefulness.

4.1. MAS project case studies

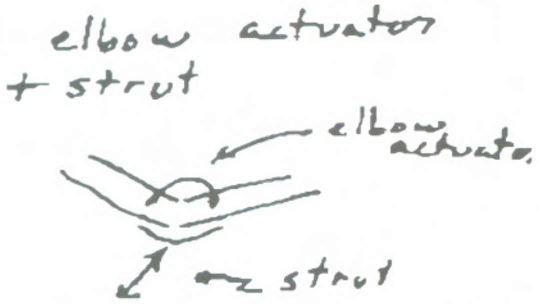
The MAS project (Chakrabarti & Abel, 1994) was to design a means for enhancing the mobility of Muscular Dystrophy (MD) sufferers. People having this disorder gradually lose strength in their muscles, and soon have little or no lifting strength in their arms, although they do not lose any of the finer controls. They become wheelchair-bound, and almost totally dependent on caretakers, even for their daily activities. In the task clarification phase of the project, it was found that the MD sufferers are capable of using their bodies to move their arms in horizontal planes in the absence of significant resistance. It was decided that an arm support with powered vertical motion would be designed as a means of enhancing mobility. There should be enough freedom in the horizontal plane for the users to use their own limited strength to move their arms. The project ran in two phases for over 3 years, which led to the development of two prototypes.

4.1.1. Comparison with designs generated in phase I of the MAS project

The MAS in phase I was designed by two designers who worked closely with each other. The designers were assisted by a brainstorming session involving eight people, which gave them an initial pool of ideas. They explored these ideas, and eventually came up with three concept variants, from which one was selected for embodiment. All these ideas are documented in Bauert (1993). Two groups of ideas were mentioned in the designers' documents. One group of about 30 contained abstract/incomplete ideas for which no sketches existed. The other had some sketches, sometimes incomplete or incomprehensible scribbles, to accompany them. There were 43 ideas, not necessarily distinct.

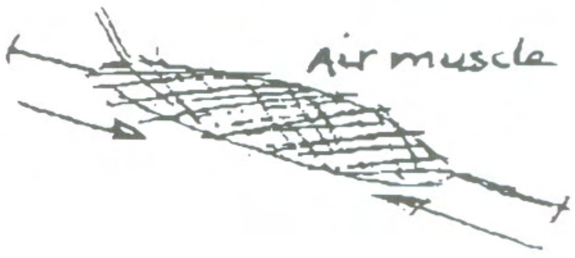
As a retrospective study, an input–output requirement, which describes the intended instantaneous function of the arm support, was given to FuncSION, for it to generate solutions and their spatial configurations. As vertical motion was the main requirement, the input was either a translation or a rotation, which could be in any of the three reference directions, and the output was specified as a vertical translation. For the two specifications given to it (one is a torque to force transformation, and the other, a force to force transformation) a total of 162 solutions were generated. These were compared with the ideas that the designers generated.

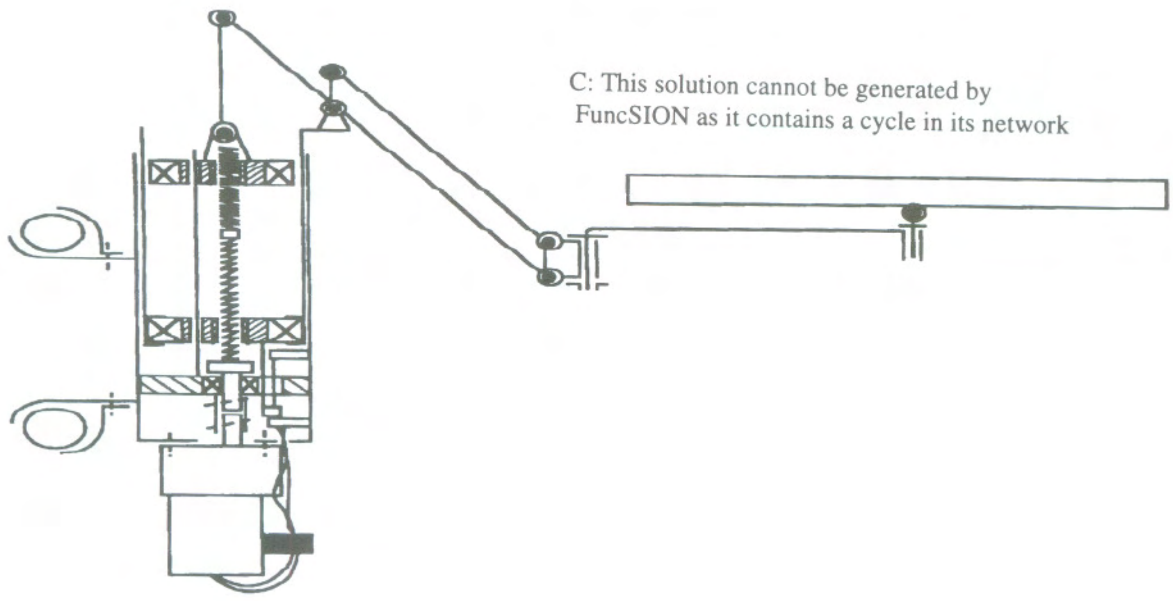
The first group of designers' ideas are not within the realm of FuncSION. These are physical effect-like solutions, or very early impressions of solutions (such as electromagnetism, or using bimetallic expansion). The designers' 43 ideas that have sketches could be classified into four categories: incomplete/incomprehensible ideas, ideas from a different domain of knowledge, ideas that FuncSION should be able to generate but at present is unable to generate (see Section 6), and ideas that can be reasoned about by FuncSION in its present form. An example of each of these four categories of ideas from the MAS project is given in Figure 7. The statistics of how designers' ideas with sketches relate to the solutions generated by FuncSION is as follows. Of the 43 ideas, 18 are either incomplete, abstract, infeasible, incomprehensible, or based on a different domain of knowledge, and therefore could not have been generated by FuncSION.



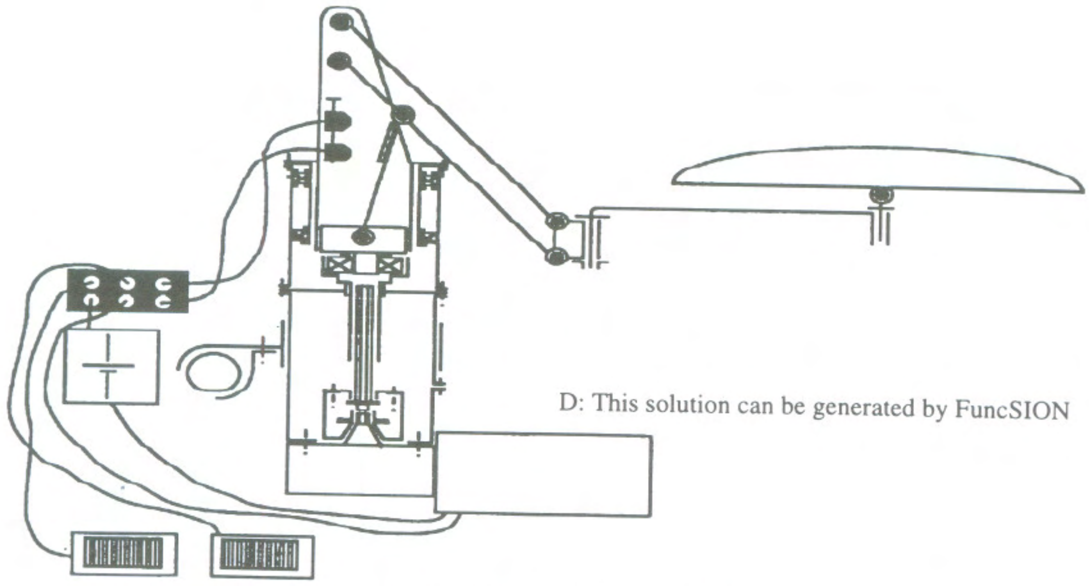
A: This solution is not sufficiently detailed to be discussed within the realm of FuncSION

B: This solution cannot be generated at present by FuncSION as it is in a different domain





C: This solution cannot be generated by FuncSION as it contains a cycle in its network



D: This solution can be generated by FuncSION

Fig. 7. Examples of various categories of ideas found in the MAS project documents.

Of the other 27 (not necessarily distinct) ideas, 22 were generated by FuncSION, and there were 5 ideas that it could not have generated, as discussed later. This comparison was done abstracting the designers' concepts to the level of FuncSION's solutions. See Figure 8 for a list of the database of elements used by FuncSION for generating solutions.

Of the 162 solutions that FuncSION generated, 13 (the number of distinct ideas of the 22) were generated by the designers. However, given that FuncSION allows the same element to be used more than once in a solution, it often generates a number of solutions that might be considered as variants, all of which would not be recorded by a designer. To be fair, we need to turn the solutions generated by FuncSION into a set that is more comparable to that produced by the designers. One method might be to group solutions generated by FuncSION into a number of clusters of similar designs, and then, if an idea exists in the designers' documents that can be abstracted as one of the solutions in a cluster from FuncSION, to assume that this cluster has been considered by the designers. There are two problems with this method. The first is the issue of what criteria should be used to group solutions as similar. The second is that the ideas generated by the designers are often at a different level of abstraction than those generated by FuncSION. If these designs are at a higher level of abstraction, they cannot be discussed within the realm of FuncSION (e.g., those without sketches). If they are at a lower level of abstraction, we would need to abstract them to the right level before they can be compared with solutions generated by FuncSION; this then has two consequent difficulties.

Take the instance of the "shaft, rack, and pinion" idea (Bauert, 1993). This should be abstracted as a "shaft, lever, and tie-rod" solution before it is compared with FuncSION (because a pinion and a rack could be abstracted as a lever and an axial tie-rod, respectively). If we assume, after having spotted an instance of the "shaft, rack, and pinion" idea, that the designers have considered the whole cluster that represents "shaft, lever, and tie-rod" then the first difficulty is whether or not the designers did consider "shaft, lever, and tie-rod" solution class as a whole, and the second is whether they did consider the whole cluster of "shaft, lever, and tie-rod"-type solutions.[5]

We have tried to deal with the first problem, finding criteria for clustering solutions, by carrying out a set of further experiments with experienced designers, and identifying their common notions of similarity for use as the clustering heuristic. The other problem, comparison of designers' idea-instances with FuncSION's solution clusters, was dealt with by this assumption: if the designers' ideas can be abstracted into more than one solution in a cluster, then they have considered the solution types represented by that whole cluster; however, if there is just a single or no idea-instance that could be abstracted as a solution in a cluster from FuncSION, then the designers did not consider this solution cluster.

There were interesting solutions that were suggested by FuncSION, which designers did not conceive. One example concept of this is given in Figure 10, which is a single link lever connecting the input rotation to a tie-rod to provide the output hand motion. It was interesting to note that some concepts, which were regarded by the designers as distinct solutions, were regarded by FuncSION as topologically the same solutions (e.g., the final two solutions in MAS I, see Fig. 11). This signifies the importance of considering spatial configurations as distinct solutions.

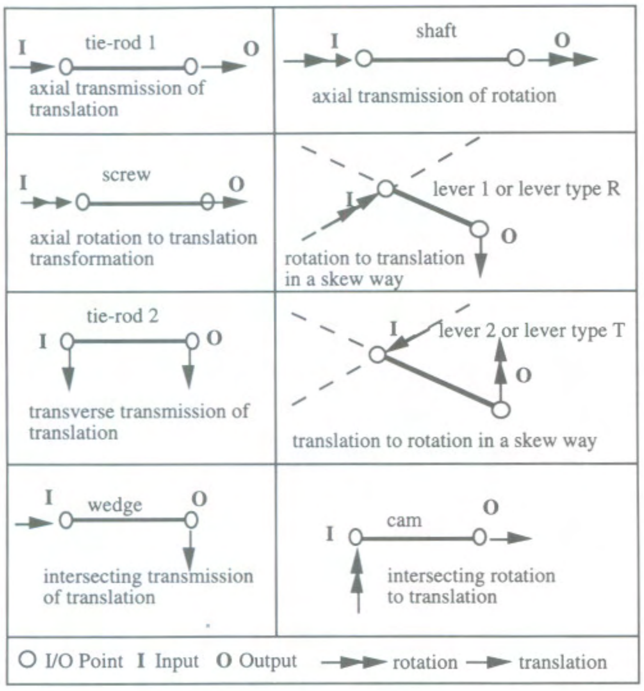

I
tie-rod 1
O
axial transmission of translation
I
shaft
O
axial transmission of rotation
I
screw
O
axial rotation to translation transformation
I
lever 1 or lever type R
O
rotation to translation in a skew way
tie-rod 2
I
O
transverse transmission of translation
I
lever 2 or lever type T
O
translation to rotation in a skew way
I
wedge
O
intersecting transmission of translation
O
cam
I
intersecting rotation to translation
○ I/O Point I Input O Output
rotation
translation

Fig. 8. The entire set of elements used by FuncSION in synthesizing concepts.

4.1.2. Comparisons with designs generated in phase II of the MAS project

In the conceptual stage of phase II of the MAS project, the designers were given the solutions generated by FuncSION in phase I, along with other existing ones, for consideration. They went through these as an exercise, hardly taking note of them as serious solutions, and got on with designing as they otherwise would, and yet did not generate all of the feasible designs which FuncSION generated in phase I. Possible rea-

[5] Each solution generated by FuncSION represents a class of lower level concepts (see Fig. 9). For instance, the "shaft-lever-tie-rod" solution could represent a shaft connected to a belt-pulley combination, a shaft-pinion-rack combination, a shaft-link-lever-connecting rod combination, etc. The solutions generated by FuncSION can be grouped into a set of clusters, each of which contains solutions that are similar. For instance, a "shaft-lever-tie-rod" solution is similar to a "shaft-lever" solution in the sense that they have similar I–O, and that only the final translating element is missing in the latter one. Again, these two solutions are similar to a "lever-tie-rod" and therefore could form a cluster as shown in Figure 9.

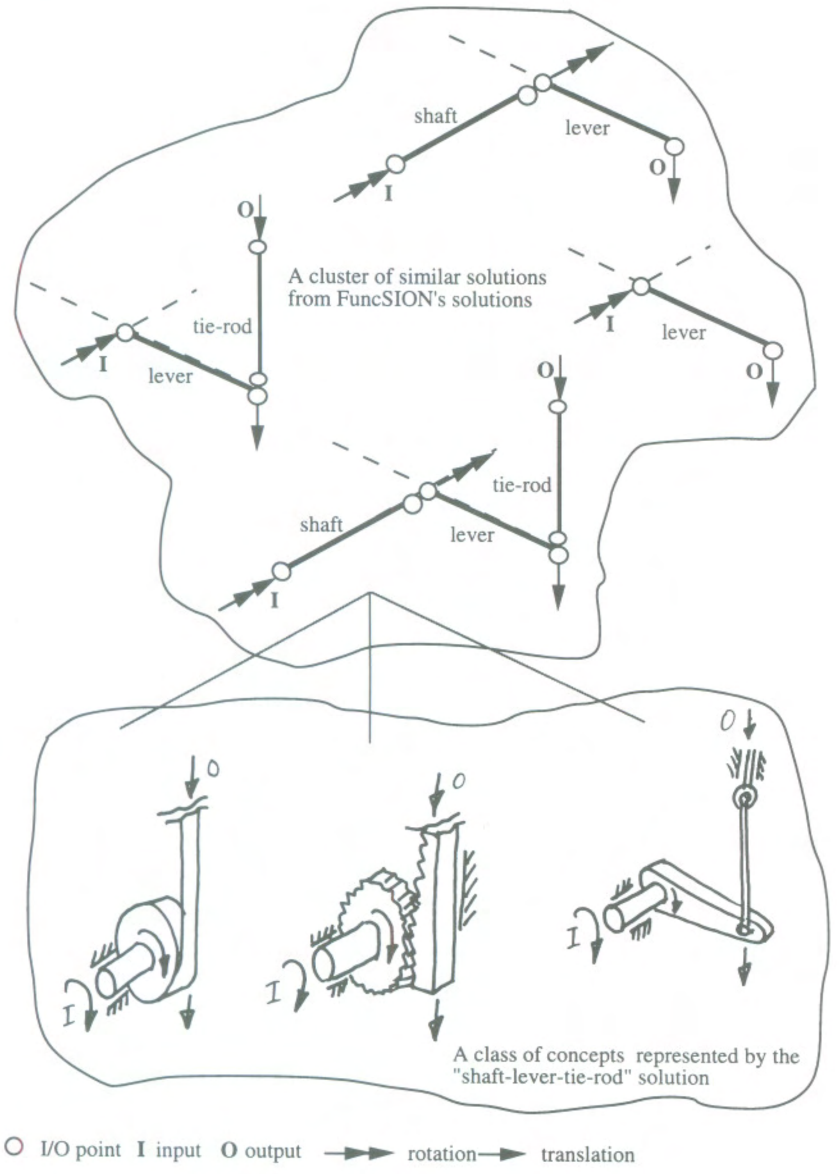

shaft
lever
I
O
tie-rod
A cluster of similar solutions from FuncSION's solutions
A class of concepts represented by the "shaft-lever-tie-rod" solution
I/O point
I input
O output
rotation
translation

Fig. 9. A class of concepts and a cluster of solutions.

sons might have been that: (1) right from the beginning this was taken as a redesign exercise, with the intention of modifying the previous design to alleviate the existing problems; (2) concepts generated by FuncSION were not easy to understand due to their user-nonfriendly abstract representation, and a lack of visualization of how they worked; (3) there were too many solutions to browse; (4) there were a number of infeasible or similar solutions, which discouraged the designers to explore further. However, these are only guesses, and need validating before they can be given serious consideration. We thus did some further testing for an evaluation, which is discussed below.

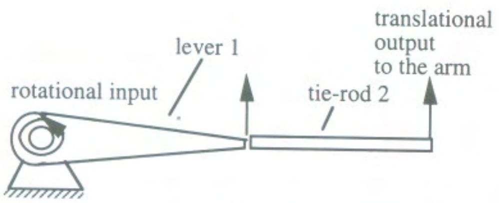


Fig. 10. One of the solutions that the designers did not conceive.

4.2. Hands-on experiments by experienced designers

Three experienced designers were asked to evaluate the solutions generated by FuncSION for aspects of their originality, feasibility, variants, and redundancy. They were also asked to comment on the ease of using FuncSION, and to make suggestions.

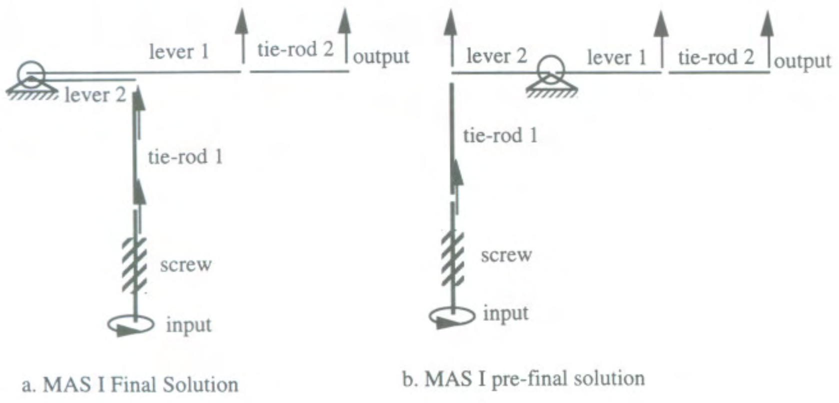


Fig. 11. The final two solutions in MAS I are but variations in the spatial configuration of the "same" topological solution in FuncSION.

4.2.1. Experience of designer A

Designer A used FuncSION twice, to generate solutions to the MAS problem using a different database of elements each time. In the first experiment he used 5 elements: 2 lever types, 2 types of translational elements, and 1 screw-type element, as shown in Figure 8. He wanted to see if FuncSION produced the solution he had in mind, which it did amongst many. The second experiment used a database of four elements consisting of a cam, 2 lever types, and a shaft. He could not think of any sensible solutions using these elements, and wanted to see if FuncSION could surprise him.

To his amazement, there were three solutions that he found useful and interesting. However, there were a large number of solutions that he thought were redundant (with repetitive translational elements such as three shafts in series[6]), thus browsing and analyzing them was frustrating. For instance, of the 20 solutions in this second experiment, 4 were redundant. He found that he could not cope with more than 20 solutions, and suggested that grouping the solutions in user-defined categories (such as all solutions with a screw, or all solutions with levers only) would make handling a large number of solutions easier. The solutions were difficult to visualize or interpret, as we expected, and he thought having iconic representations coupled with a simulation facility would make visualization easier.

4.2.2. Experience of designer B

Designer B used a database of 8 elements, and asked for all solutions that used up to 5 of these. FuncSION produced a large number of solutions (more than 700). These were impossible to explore. He then reduced this to solutions having up to 4 elements, and there were still too many to investigate (255). He then eliminated the wedge element from the database, and the number of solutions dropped to 150, which still was too large. However, he went through the first 32. He then added a further constraint that the input and output should be vertical, which left 16 solutions to explore. Among these, he found that having tie-rods or shafts were just variations. Use of screws as the main drive with tie-rods was considered one cluster; seeing a vertical tie-rod on a screw gave him the idea of using a sleeve to isolate the translational component of the screw (insight). There were 5 additional distinct clusters of solutions as recorded in his notes. These were screws with tie-rods and levers, levers and tie-rods, two cams connected by a lever, a cam driving two levers, and a cam connected with various tie-rods.

To start with, levers were not seen as abstractions of gears; also, he did not realize that a lever, as used in FuncSION, was not a conventional seesaw type but a more fundamental element, which could be combined in various ways to produce bell-crank levers as well as seesaws. Cams, in the present representation, were hard to visualize, and he did not visualize tie-rods as axial links.

Regarding the procedures, Designer B felt that when he found an interesting cam-based design, he wanted to explore all the cam-based designs. So again, a user-defined clustering facility would be useful. He suggested that it would be useful to do the synthesis with only output specified.

He eliminated shafts and cams for the second experiment. The number of solutions now was about 50. He went through the first 10 systematically before realizing that tie-rods, as well as shafts, are spacers, and could be left out as long as they were the last elements in a solution. Of the 10 that were evaluated, those which had a tie-rod at the end or beginning, as well as those with tie-rods in series, were considered variants.

A solution having three levers in series followed by a tie-rod was interpreted as a feasible but not an exciting solution when the levers were interpreted as link-levers. But when they were interpreted as gears, the designer found the same solution a clever new idea as this became a rack and pinion solution. This means that being able to see the solution at levels of greater detail often reveals more insight as to how useful it might be.

In his third experiment, he used a database of just three elements: levers of the two types, and screws (as contained

[6] This type of redundancy will be eliminated in a future version of FuncSION.

in Fig. 8). This gave him just 5 solutions; all appeared to be distinct solutions. This set of solutions included both MAS I and MAS II final solutions.

4.2.3. Experience of designer C

Designer C initially chose 3 elements from the database (cam and the two lever types), and asked for solutions having at most 3 elements. There were two solutions: one with two levers and the other with two levers and a cam in between. He expected them both and there were no surprises. He then asked for solutions with 4 elements; there were 20, several redundant (i.e., there was one or many translators in series). Again, there were no surprises. This is not surprising because the database chosen was limited and there was not much scope for innovation.

He felt that visualization would be improved if the symbols were more self-explanatory, and if two solutions could be seen alongside one another.

5. OBSERVATIONS

The designers in the above experiments found that FuncSION, in general, generates a range of interesting solutions, and often comes up with surprisingly clever ideas and provides insight. However, it also generates a large number of redundant solutions, and this makes it difficult or frustrating to evaluate and explore the ideas in any depth. They had some difficulty in visualizing possible embodiments of the solutions with the present representation, and could visualize only when these solutions were shown at a lower level of abstraction.

On the whole, the above experiments suggested that the designers saw a common pattern to solutions that were considered similar and these could be clustered (see Fig. 9): they felt that if two solutions are different only by a translational element (e.g., a tie-rod or a shaft), then they are similar.

If the above criterion, of what similar solutions mean, were applied to cluster the solutions that FuncSION suggested in the MAS I and II cases described before, these would fall into 17 different clusters of solutions of an average size of about 10 (see Fig. 12). Of these, only 4 clusters were considered at any length at all by the designers in MAS I and MAS II, while just a single low-level instance was found for 2 of the 13 other clusters. This indicated the potential of FuncSION for suggesting different ideas and idea types. It is important to note that the above clusters were the result of solving the MAS problem using a database of only 5 elements. If this database were increased to 7 for instance, the number of distinct clusters would be as high as 29, of which only 6 would then have been considered at all by the designers, and only 4 of these in any depth.

Figure 12 shows how the number of solutions per cluster varies for each cluster (i.e., for each distinct solution how many variants or redundant solutions are produced), and how many solutions in a cluster were generated in the MAS projects. As seen in Case I of Figure 12, FuncSION, in the case of a rotation to translation requirement using a database of 5 elements with a maximum of 5 elements per solution, produced 14 clusters with a total of 138 solutions with solutions per cluster ranging from 1 to 31. Of these 14 clusters, only 3 were considered by the MAS designers in some depth (i.e., more than one variant was produced). Only one of the other 11 clusters was touched by any designer at all.

In Case II (with translation to translation as requirement and using a database of 5 elements and a maximum of 3 elements per solution), FuncSION generated 3 clusters with 24 solutions in total, where the solutions per cluster varied from 5 to 14. MAS designers independently produced 4 variant solutions in one cluster, barely touched another, and did not touch the remaining one.

Figure 13 shows the solution clusters produced by FuncSION for the above two cases with an expanded database, and how these relate to the ideas generated in the MAS project. In Case III (rotation to translation as requirement, and using 7 database elements and a maximum of 3 elements per solution), FuncSION generated 18 clusters, of which 4 were independently generated by the MAS designers.

In Case IV, the number of clusters produced by FuncSION is 13 (for translation to translation as requirement, and using 7 database elements and a maximum of 3 elements per solution), of which 2 were considered by the designers.

Note that the only parameter different between Case II in Figure 12 and Case IV in Figure 13 is the number of elements in the database. This makes a large difference to the number of clusters produced by them (3 to 13), which illustrates how a wider range of elements in the database will lead to the generation of a wider range of solutions.

6. DISCUSSION

Based on the experience gained from the above case studies and experiments, it was felt that FuncSION needed further attention in four areas:

6.1. Too many solutions

One of the problems associated with the synthesis approach adopted by FuncSION is that it may generate so many solutions that it is difficult for the user to even browse through them. Some of these solutions were redundant. However, the exploration of "redundant" solutions often might be useful if the nonredundant ones cannot provide some additional functions.

Take the design of the gearbox in Figure 14a, as an example solution to the problem of transmitting the same torque and speed across a distance. Its representation in terms of elements generated by FuncSION is also shown in Figure 14a. Here lever-1 and lever-2 are used more than once (note that this is the first of an infinite number of solutions, each of which contains an odd number of gears), which

Type of solution cluster	*MAS/R*
Case I	
screw	3/31
lever-typeR	0/31
screw → lever-type T → screw	0/17
screw → lever-typeT → lever-typeR	2/17
lever-typeR → lever-typeT → lever-typeR	2/16
lever-type R → lever-type T → screw	1/18
screw → lever-type T → screw → lever-type T → screw	0/1
screw → lever-type T → screw → lever-type T → lever-typeR	0/1
lever-type R → lever-type T → screw → lever-type T → screw	0/1
lever-type R → lever-type T → screw → lever-type T → lever-type R	0/1
screw → lever-type T → lever-type R → lever-type T → screw	0/1
screw → lever-type T → lever-type R → lever-type T → lever-type R	0/1
lever-type R → lever-type T → lever-type R → lever-type T → screw	0/1
lever-type R → lever-type T → lever-type R → lever-type T → lever-type R	0/1
Case II	
tie-rods	4/14
lever-type T → screw	0/5
lever-type T → lever-type R	1/5

MAS/R: the ratio of no. of ideas generated in MAS to that by FuncSION in a given cluster

Fig. 12. For two test cases, the variation of the number of solutions in clusters, and how these solutions relate to those in the MAS project.

should not be considered "redundant" as this solution is not possible without at least this set. If we considered the next solution containing 5 gears, this may well contain redundancy, unless, for example, there was a space constraint restricting the size of the gears.

As a second example, take another solution that was generated by FuncSION for phase II of the MAS project (see Fig. 14b). There are two consecutive tie-rods, which might appear to be redundant, unless one were trying to provide an extra degree of freedom for the movement of the output point in the horizontal plane.

Take the MAS I project as a third example, in which the final two solutions are considered by FuncSION to be topologically the same (Fig. 11). They are only different in terms of the sense configuration. However, for the designers, one was considered to be a substantial improvement over the other, as it made the design more compact. So, whether a design is to be considered redundant or not, depends largely upon the other requirements that the design might have. Also, the exploration of redundant solutions might be useful if these can provide some additional functions that were not initially thought of.

However, it is clear that far too many solutions are typically produced by FuncSION to be explored by the designer meaningfully. For example, take a typical case of synthesis where only 32 topologically distinct solutions are

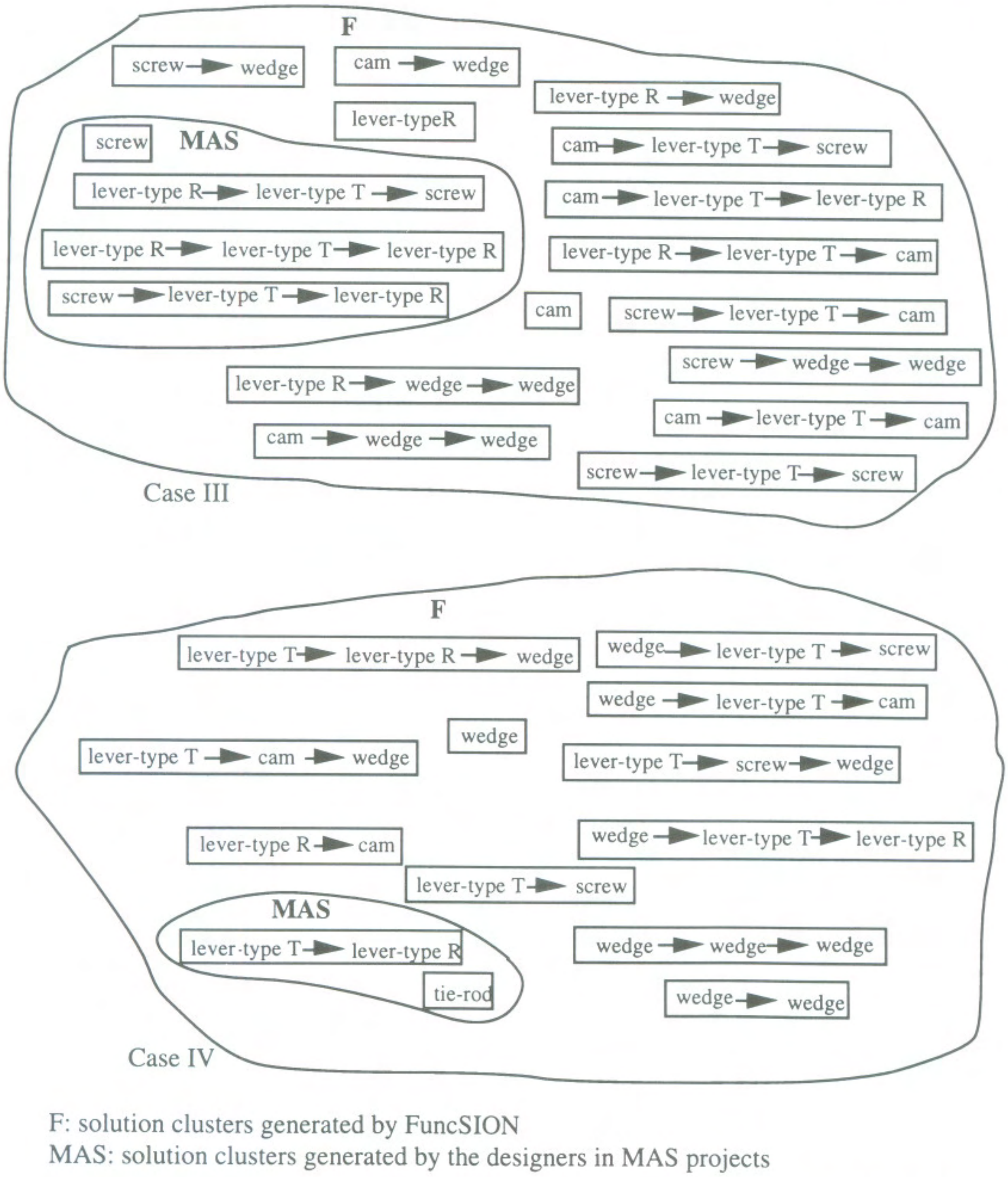

F
screw → wedge
cam → wedge
lever-type R → wedge
lever-typeR
screw
MAS
cam → lever-type T → screw
lever-type R → lever-type T → screw
cam → lever-type T → lever-type R
lever-type R → lever-type T → lever-type R
lever-type R → lever-type T → cam
screw → lever-type T → lever-type R
cam
screw → lever-type T → cam
screw → wedge → wedge
lever-type R → wedge → wedge
cam → lever-type T → cam
cam → wedge → wedge
screw → lever-type T → screw
Case III
F
lever-type T → lever-type R → wedge
wedge → lever-type T → screw
wedge → lever-type T → cam
wedge
lever-type T → cam → wedge
lever-type T → screw → wedge
lever-type R → cam
wedge → lever-type T → lever-type R
lever-type T → screw
MAS
lever-type T → lever-type R
wedge → wedge → wedge
tie-rod
wedge → wedge
Case IV
F: solution clusters generated by FuncSION
MAS: solution clusters generated by the designers in MAS projects

Fig. 13. How the solution clusters generated by FuncSION relate to those in the MAS project for two test cases.

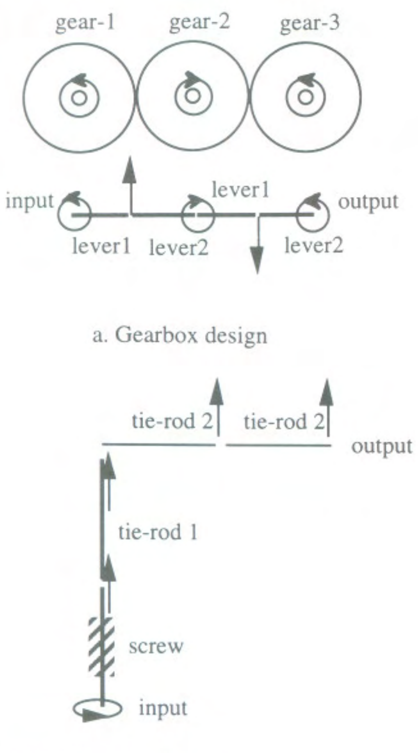
generated from a database of 5 elements and a single I–O. Each of these can have at least 4 spatial configurations, each of which can have at least another 3 physical concepts, giving a total of at least 384 solutions. The conclusion is that a strategy is needed to generate or present these solutions in a way which allows browsing with reasonable effort.

6.2. Too difficult to interpret and to visualize

Two main issues concerning the interpretation and the visualization of the synthesis results were considered vital by the designers. The first is that the representation of elements is too abstract. The second is that the static representation for functional elements and conceptual solutions makes it hard for the designer to imagine their likely temporal behavior. Thus, a means of visualizing solutions and their elements should be developed.

6.3. Some designs do not function temporally

So far, all the solutions that FuncSION generates work at one instant of time. For instance, a lever-type element represents a transformation from an instantaneous input to an



b. MAS II final solution

Fig. 14. Repetition of the same element need not be redundant.

instantaneous output. This could be an abstraction of a gear or belt-type element, which can provide translation at that point for an extended length of time, or it could be a link-type lever whose position and direction of output change with time. The conclusion is that a temporal reasoning facility is required to evaluate the potential of each such solution to function temporally.

6.4. Some designs generated in the MAS project were not generated by FuncSION

The ideas in the MAS project that cannot, at present, be generated by FuncSION fall into the following categories:

6.4.1. Ideas that require elements with an input (or output) of more than one kind

Take the example of the solution in Figure 15a. Here the final element, for instance, acts simultaneously as three basic elements: a transverse tie-rod, which takes an input from the actuating element to move up and down; an axial tie-rod, which moves in and out; and a lever, which rotates. This is essentially a multiple input single output system, where the single output is a combination of various I–O kinds, and where the final structural element acts simultaneously as three functional elements. The present version of FuncSION is restricted to using elements of only a single I–O kind at any I–O point. We could, however, modify the algorithm to deal with elements having multiple I–O kinds, if this is shown to be worthwhile.

6.4.2. Elements that have multiple I–O points

At present FuncSION only uses, as building blocks, elements that have a single input and a single output point. However, this restricts the use of elements such as the one in Figure 15b, which could be represented as a single rotation to translation transforming element, or as an element that transforms two opposite translations into a single translation, or as an element that transforms an input translation into an output translation, each in an intersecting way. Representing in the first and the third ways are already possible in FuncSION; however, the second is a two-input point element, which cannot be used at present in FuncSION, but could be incorporated. The reason for this restriction was to ensure generation of distinct solutions only (see Chakrabarti, 1991; and Chakrabarti & Bligh, 1994*a*).

6.4.3. Solutions that contain 'parallel paths'[7]

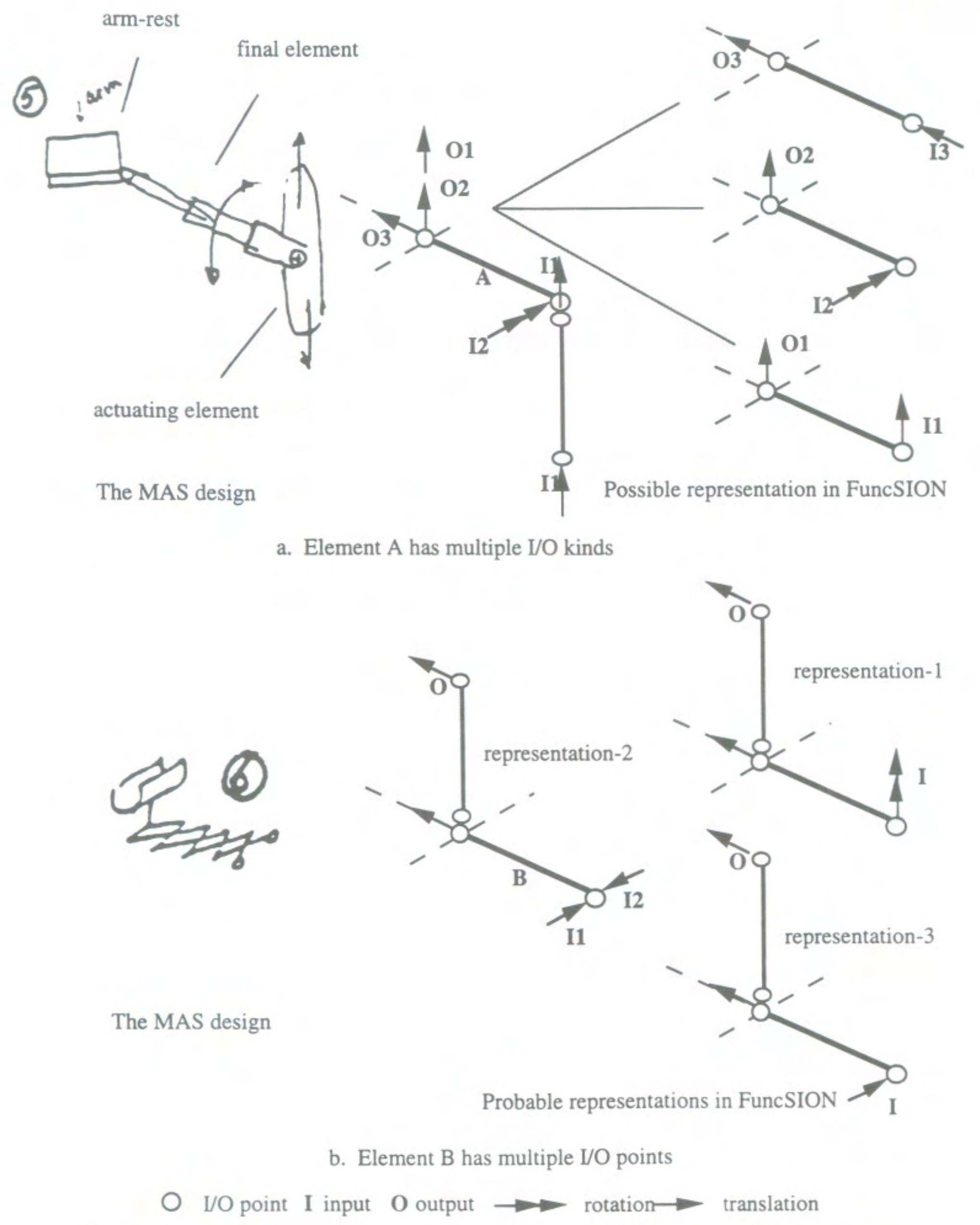
See the solution in Figure 16. Here the same input rotation is taken by a right-handed thread and a left-handed thread on the same shaft to move two nuts in opposing directions; these movements are transmitted to two points of element A to make it rotate. This rotation is converted into an arc-like movement via a lever. Here also, the first thread acts as a shaft and a screw. The output of the shaft is taken by another thread to produce motion in a direction opposite to the former screw, while that of the former screw is taken by tie-rods in another path. These two motions eventually act together to produce the rotational motion of the element A. In other words, input from a single point in the solution is taken through two different paths to produce a single output. That is, there are two parallel paths (i.e., two points in the network are connected via two alternative series of arcs) in the solution. Parallel paths are strictly avoided in the present version of FuncSION due to the possibility of redundancy. However, it should be possible to generate path-redundant variations of a given solution if required.

7. RELATED WORK, CONCLUSIONS, AND FURTHER WORK

There are three main areas that relate to this work. One is design methodology and how synthesis aspects could be supported. The second is how these could be supported on computers, and the third is the user interface issue.

There is evidence in design theory and methodology that it is important to generate a wide range of concepts and to explore them sufficiently before homing in on promising ones for further development. In fact, in some of the protocol studies done in the recent past, it has been found that the

[7] A graph contains parallel paths if it has a pair of nodes that are connected via more than one path.



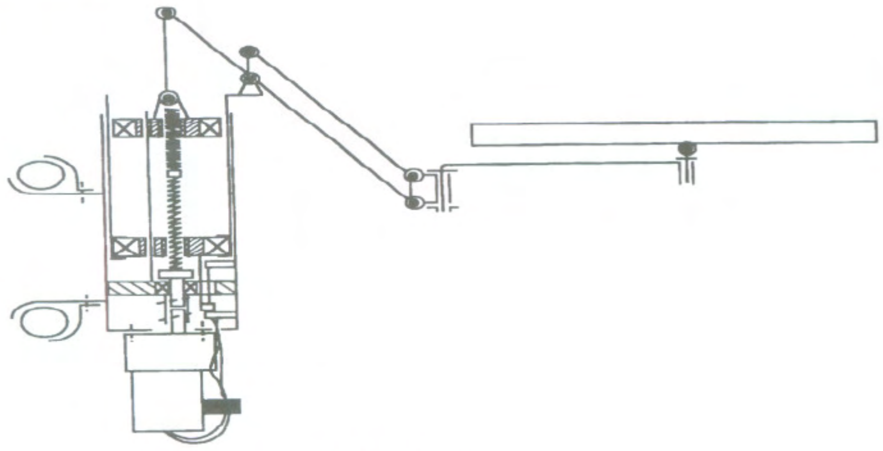
a. Element A has multiple I/O kinds

b. Element B has multiple I/O points

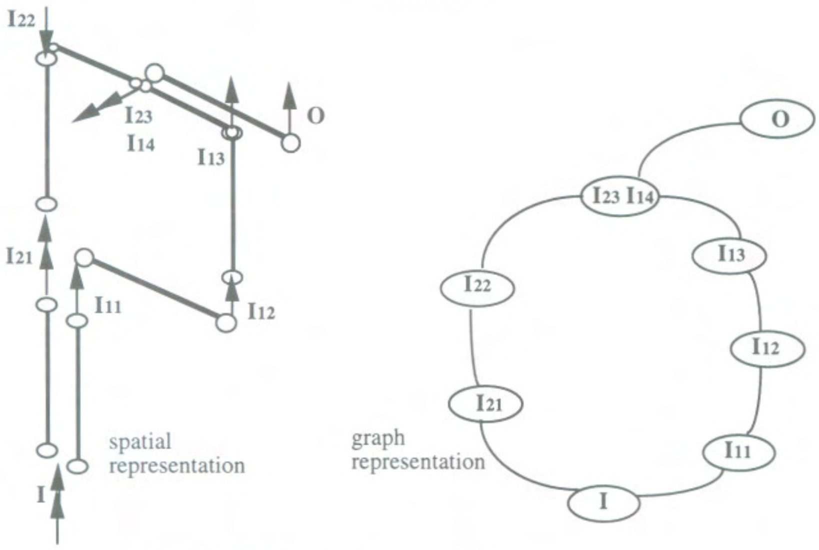
Fig. 15. Two solutions from the MAS project and their possible future representations in FuncSION.

best approach in the conceptual phase has been a consecutive expansion and narrowing down of ideas (Ehrlenspiel & Dylla, 1989; Fricke, 1992).

One of the crucial reasons why supporting generation and exploration of mechanical design concepts has been so hard, is due to the essentially coupled geometrical nature of these designs. Mechanism synthesis is probably the oldest area of endeavor in solving this problem. From the classic account of Reuleaux (1876), to recent attempts such as by Hoeltzel and Chieng (1990), a number of interesting attempts have been made. A more comprehensive review can be found in Chakrabarti and Bligh (1994*a*), but probably the two central reasons of why progress in mechanism synthesis has been less than adequate are: (1) the difficulty of complete abstraction of geometry by conceiving designs in terms of combinations of kinematic pairs, and (2) the freezing of types of relative motions between elements by the very specification of the kinematic pairs, which prevents dynamic changes in a kinematic pair type at a connection. A compromise is beginning to emerge with the use of Configuration Space (Lozano-Perez, 1983) in which a mechanism is represented by the relative spaces in which they overlap, touch, or do not touch, and where a valid configuration of a mechanism is defined by the relative position of its com-



The MAS concept



Possible representations in FuncSION

Fig. 16. A concept from the MAS project, which contains parallel paths.

ponents where they do not overlap. The motion of the mechanism could be simulated by starting with a valid configuration and then computing the series of possible valid configurations, which the mechanism can assume, given appropriate inputs (Faltings, 1987; Neilsen, 1988; Joskowicz, 1989). Although a few attempts have been cited in the literature about the use of Configuration Space for synthesis (Joskowicz & Addanki, 1988; Sun & Faltings, 1994) (these are often limited to interaction between two elements), the approach seems more suitable for mechanism simulation than for synthesis.

The other important strand of research has been based on Bond Graphs (Paynter, 1961), which is based on an elegant lumped-parameter simplification of the general energy equation. Here, a design is expressed as a combination of some transformers, dissipaters, sources, and connections between them. A number of attempts based on this can be found in the literature (Ulrich & Seering, 1988; Hoover & Rinderle, 1989; Finger & Rinderle, 1990). While this framework provides elegant generalizations across different domains of knowledge such as electrical, mechanical, and hydraulic systems, and therefore can support interdisciplinary designs, it is devoid of the all-essential geometric knowledge for the reason of generality, which renders it inadequate to reason about the functioning of mechanical devices.

However, there has been a general problem with all of these approaches, which is that they generate too many solutions. So the other hard problem is managing complexity. How can the designer be supported in exploring solutions without compromising their range? In their attempts, Lee et al. (1992) found that granularity of building blocks is particularly important for managing complexity, and they felt complexity could be tackled using a few important parameters at a time. However, this is only part of the problem. Even if the problem is solved using a few parameters at a time, there would still be a large number of feasible alternatives to compare, evaluate, and modify. We feel that the major part of complexity arises from the conflict about the right level of abstraction, which allows high explorability as well as for a wide range of solutions. We feel that this needs to be tackled by (1) clustering designs based on designers' heuristics of similar designs; (2) providing range by generating solutions at a high level of abstraction, while allowing visualization at lower levels for each of these solutions; and, (3) by allowing the designer to navigate the solution space.

To summarize, the functional synthesis approach presented in this paper has been implemented and tested using a number of case studies and hands-on experiments. On the whole, besides being able to generate a wide variety of ideas for the above cases, which included more than 80% of the ideas generated by the designers, the system has managed to suggest a wide variety of other feasible and novel ideas that the designers did not generate.

Although much more validation is required before it can be stated conclusively, the above provides substantial evidence to say that this approach has demonstrated that, in principle, a designer can be assisted in the generative aspects of design. However, in the present state, the program allows one to define a design problem in terms of only instantaneous characteristics of input and outputs, and therefore, only ensures that the designs generated do satisfy the function at this instant. What about the time-varying characteristics? If the design problem for the scotch-yoke mechanism is presented as a single input–output function to convert a continuous rotary motion into a reciprocating linear motion, the function that can be satisfied by designs generated at present is conversion of an instantaneous input rotation into an instantaneous translational output. What else is required to ensure that the time-varying characteristics are satisfied as well? In other words, what else is necessary to move from the solution generated by FuncSION in Figure 17 (i.e., the stick diagram) to the scotch-yoke mechanism embodiment shown there?

We believe that the answer to this question lies in how spatial characteristics interact in time, and therefore, a spatio-temporal extension of the, at present, solely spatial representation of problems and solutions, is necessary. Such a representation will solve not only the question of how to generate designs that would satisfy design problems represented by time-varying functions, but also the questions of how designers could be allowed to manipulate the functionality of an existing design, and how the problem of combinatorial explosion associated with any exhaustive generative algorithm (and this program is by no means free from this) could be reduced. The reason for the latter is that by trying to solve a time-varying problem as an instantaneous problem, one often makes it highly under-constrained, thereby allowing consideration of designs which would not be feasible for the time-varying problem. By bringing the additional constraints associated with a time-varying problem into the generative process, generation of solutions that would not work at subsequent times can be eliminated, thereby reducing combinatorial problems. Work in this area has already started, and a theory of how this could be done, based on a sequence diagram concept, is outlined in Chakrabarti and Bligh (1994*b*).

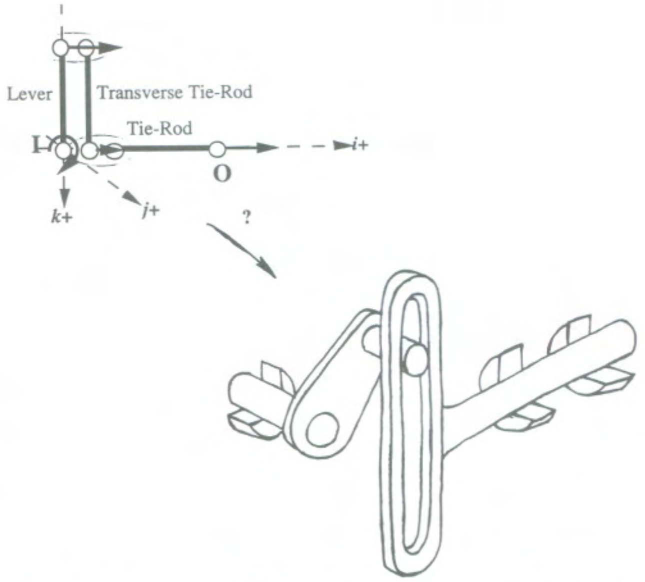


Fig. 17. How can we generate the scotch-yoke mechanism embodiment from the solution generated by FuncSION?

However, there also have been problems about coping with too many solutions, and about visualizing solutions and elements, and their behavior. Work in supporting visualization is being tackled by developing more user-friendly and multilevel representations of the elements and solutions, as well as animation facilities for visualizing their behavior. Supports for coping with too many solutions are being developed by providing a framework that would allow generation and browsing of solutions at multiple levels of abstraction, and clustering them as variants, as well as using user-defined clustering criteria (Chakrabarti & Tang, 1996).

Another problem has been that there are designs generated by designers that should also have been generated by FuncSION, but were not. There are two reasons. Some of these solutions are due to implementing only part of what can be implemented. The others are due to more fundamental restrictions (such as not allowing parallel paths within the solution topologies) and therefore, need further research in terms of (1) whether it is important to remove these restrictions (i.e., how important these solutions are) and (2) how this could be done.

ACKNOWLEDGMENTS

The work presented in this paper is currently being funded by the EPSRC, UK. We acknowledge the support from Dr. Stuart Burgess, Mark Nowack, and Doug Isgrove, who acted as the designers in the experiments reported in this paper. We also thank Ming Xi Tang for allowing us to use his Functional Modeller interface for the experiments.

REFERENCES

Bauert, F. (1993). *The mobile arm support phase I: design, manufacture, testing, software tools*. Technical Report No. CUED/C-EDC/TR 13, Cambridge University, UK.

Berliner, C., & Brimson, J.A. (Eds.) (1988). *Cost management for today's advanced manufacturing: The CAM-I conceptual design*. Harvard Business School Press, Boston.

Chakrabarti, A. (1991). *Designing by functions*. Ph.D. Thesis, University of Cambridge, UK.

Chakrabarti, A., & Abel, C. (1994). The mobile arm support project: Evolution of the management, design, tools, and documentation processes. *Proc. 16th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. Conf.*, 486–487.

Chakrabarti, A., & Bligh, T.P. (1994*a*). An approach to functional synthesis of solutions in mechanical conceptual design. Part I: Knowledge representation. *Int. J. Res. Eng. Design 6(3)*, 127–141.

Chakrabarti, A., & Bligh, T.P. (1994*b*). A two-step approach to conceptual design of mechanical devices. *Proc. 3rd Int. Conf. on AI in Design*, 21–38.

Chakrabarti, A., & Bligh, T.P. (1996*a*). An approach to functional synthesis of solutions in mechanical conceptual design. Part II: Kind synthesis. *Int. J. Res. Eng. Design 8(1)*, 52–62.

Chakrabarti, A., & Bligh, T.P. (1996*b*). An approach to functional synthesis of solutions in mechanical conceptual design. Part III: Spatial configuration. *Int. J. Res. Eng. Design 8(2)*, 116–124.

Chakrabarti, A., & Tang, M.X. (1996). Generating conceptual solutions on FuncSION: Evolution of a functional synthesiser. *Proc. 4th Int. Conf. on AI in Design*, 603–622.

Ehrlenspiel, K., & Dylla, N.D. (1989). Experimental investigation of the design process. *Proc. Int. Conf. Eng. Design*, 77–95.

Faltings, B. (1987). Qualitative kinematics in mechanisms. *Proc. Tenth IJCAI*, 436–442.

Finger, S., & Rinderle, J.R. (1990). A transformational approach for mechanical design using a bond graph grammar. EDRC Report No. 24-23-90, Carnegie-Mellon University, Pittsburgh, USA.

Fricke, G. (1992). Experimental investigation of individual processes in engineering design. In *Research in Design Thinking* (Cross, N., Doorst, K. and Roozenburg, N., Eds.), pp. 105–109. Delft University Press, Delft.

Harlequin Limited (1991). *LispWorks reference manual*. Harlequin Limited, UK.

Hoeltzel, D.A., & Chieng, W.-H. (1990). Knowledge-based approaches for the creative synthesis of mechanisms. *Comput. Aided Design 22(1)*, 57–67.

Hoover, S.P. & Rinderle, J.R. (1989). A synthesis strategy for mechanical devices. *Res. Eng. Design 1*, 87–103.

Joskowicz, L. (1989). Reasoning about the kinematics of mechanical devices. *AI Eng. 4(1)*, 22–31.

Joskowicz, L., & Addanki, S. (1988). From kinematics to shape: An approach to innovative design. *Proc. AAAI, 1*, 347–352.

Lee, C.-L., Iyenger, G., & Kota, S. (1992). Automated configuration design of hydraulic systems. *Proc. 2nd Int. Conf. in AI in Design*, 61–82.

Lozano-Perez, T. (1983). Spatial planning: A configuration space approach. *IEEE Comput. C-32(2)*, 108–120.

Neilsen, P. (1988). A qualitative approach to mechanical constraint. *Proc. AAAI, 270–273.*

Pahl, G., & Beitz, W. (1984). *Engineering design*. The Design Council, London.

Paynter, H.M. (1961). *Analysis and design of engineering systems*. The MIT Press, MA.

Reinschke, K.J. (1988). *Multivariable control: A graph theoretic approach*. In *Lecture Notes in Control and Information Sciences, 108* (Thoma, M. and Wyner, A., Eds.), Springer Verlag, Berlin.

Reuleaux, F. (1876). *The kinematics of machinery: Outline of a theory of machines*. (Published in 1876. Reprinted in 1963.) Dover, USA.

Sun, K., & Faltings, B. (1994). Supporting creative mechanical design. *Proc. 3rd Int. Conf. on AI in Design*, 39–56.

Ulrich, K.T., & Seering, W.P. (1988). Computation and conceptual design. *Robot. Comput. Integrated Manufact. 4(3/4)*, 309–315.

Amaresh Chakrabarti received a BE in Mechanical Engineering from the University of Calcutta in 1985, an ME in Mechanical Design from Indian Institute of Science in 1987, and a Ph.D. in Engineering Design from Cambridge University in 1991. He has since been associated with the Engineering Design Centre at Cambridge University, as a Senior Research Associate. In 1994, his software FuncSION won a prize in the UK Morgans-Grampian Manufacturing Industry Achievement Awards competition. His main interest is in design methodology, particularly in the earlier phases of design. This includes requirements identification, functional representation, conceptual design, and engineering design research methodology.

Thomas Bligh received his B.Sc. M.Sc. (Hovercraft), and Ph.D. (Detonations) from the University of Witwatersrand. After 4 years in industry, he joined University of Minnesota developing explosive tools for oil and gas recovery, energy conservation, earth-sheltered buildings (160,000 built), a solar concentrator for the largest solar heating and cooling project, and received ASCE's Outstanding Engineering Achievement (1983). In 1979 he joined MIT, designing a new solar concentrator for the world's largest solar power plant, and consulted for US Windpower's first "wind farm." His roof-integrated solar collectors are widely used. In 1986 he joined Cambridge University working on design and performance of yachts and ships, AI for CAD, and vision assisted robots for the Human genome programme, manufactured by his company BioRobotics Ltd. He publishes widely and holds 6 patents.