A Bird’s-Eye View of Cyclic Spatial-Structural Transformations, Grammar Based Design, and Structural Optimization

Herm Hofmeyer 1, J.G.M. Kerstens 2
1 Visiting RWTH Aachen, Fakultät für Architektur; Computergestütztes Planen in der Architektur, Univ.-Prof. Dipl.-Ing. M.Arch. P. Russell, Germany. Normally at Technische Universität Eindhoven (TUE), Faculty of Architecture, Building, and Planning (ABP), The Netherlands, 2 TUE, ABP
http://www.tue.nl
1 h.hofmeyer@tue.nl, 2 j.g.m.kerstens@tue.nl

Abstract. Recently, in the research field of combined architecture and structural design, the idea of a so-called research engine has been developed. During its development, initially the intention was to develop a design-support tool, however, it now becomes clear that it may serve as a general framework for several applications. In this paper the new possibilities for the applications will be presented. As a result, a bird’s-eye view is developed on a set of applications that are often presented separately and without a larger framework in literature. In conclusion, several applications used for CAAD and CASD (Computer Aided Structural Design) can be seen as derivatives of a research engine. The other way round, the research engine can be modified and new applications may result. Keywords: Generative design; Optimisation; Spatial Design; Structural Design; Automated design.

Introduction

Recently, in the research field of combined architecture and structural design, the idea of a so-called research engine has been developed (Hofmeyer, 2007). This research engine carries out automated cycles using four types of transformations: (1) from spatial design to structural design, (2) from structural design to another structural design, (3) from structural design to spatial design, and finally (4) from a spatial design to another spatial design, as shown in figure 1. During the development of this idea, initially the intention was to develop a design-support tool, however, it now becomes clear that it may serve as a general framework for several procedures. In this introduction, some of these procedures will be introduced briefly, with a focus on spatial and structural design aspects.

We start with the traditional building design process, in which the architect designs a spatial
design and thereafter the structural engineer tries to develop a structural design that fits the architectural design. Often, feedback of the structural engineer will change the spatial design slightly, but it cannot be stated that there is strong interaction between the conceptual spatial design and the structural design. A similar situation exists for the traditional mechanical design process (Ullman, 2002) where first functional systems are designed, and then the spatial envelop to protect or hide the systems. Again feedback from the (spatial) designer may change the layout of the functional systems slightly, but it cannot be said that the spatial outer design will interact fully with the functional system design. Another procedure to mention is grammar based design where shape grammars (e.g. Kotspoulos, 2006) are applied to developed a spatial design. A grammar consists of a so-called engine that fires rules: a start rule that initiates the process, transformation rules of which one should be selected for each step, and a termination rule. Besides the development of a spatial design, the principles of grammar based design can also be used to develop structural designs as was shown by Shea et al, 2005.

The last procedure to introduce is the optimization of a structural design. It can be seen as minimizing or maximizing a function related to the structural design (e.g. strain energy or mass) by a correct prediction of the variable values. Frequently used techniques to carry out this optimization are simulated annealing, where a variant of the current variable values set replaces the current set with a probability derived from the function values for both sets and the ‘temperature’, which lowers during the process, e.g. (Lamberti, 2008). Genetic algorithms (a subset of evolutionary algorithms) have some resemblance with simulated annealing, but instead of one set of variable values, more sets are monitored during an optimization step. Probabilistic techniques are used to keep promising value sets and to combine them to new ones for the next step (Gero et al, 2005). Sometimes an ant colony strategy is used, a technique also related to genetic algorithms, but here the probability for a new value set (the direction of the ant) is predicted by determining how many other sets have been in this direction in a short period (ants’ pheromone trails), for instance demonstrated by Kaveh and Jahanshahi, 2008.

If the architectural design (a state of the design in discipline A) effects the structural design (a state of the design in discipline B), finding an optimal solution needs some more effort. To explain this further: assume that we try to find minimal building costs for an architectural design. Now the structural design is developed, but this seems to lead to a modified architectural design, thus the optimal solution is lost. Furthermore, it is also possible that a more expensive architectural design needs a less expensive structural design so that the optimal solution should be found in this way. This is in a nut-shell the domain of multi-disciplinary optimization (MDO). Some examples of used techniques for MDO will be presented below.

The MultiDisciplinary Feasible method (MDF), as presented by Alexandrov and Kodiayalam, 1998, uses the design variables (e.g. location of building elements, loads, structural boundary conditions) to solve the state vector (a solution in which both the
spatial and structural design are not influencing each other anymore, a sort of convergence). Using a nonlinear programming technique, the derivatives of the function to be optimized to the design variables, using the known state vector, can be used to do a new prediction for the design variables. Note that with the MDF method for each step in the optimization process, a building design has to be generated that yields a good solution for all the disciplines, which is very costly in terms of calculation time.

The Collaborative Optimization approach (CO) tries to solve this problem by the introduction of constraints for the total system that ensure that the design is correct for all the disciplines (Alexandrov and Kodiyalam, 1998). Because the constraints have the form of a summation of terms for each discipline, their derivates, needed for the optimization process, can be calculated for each discipline separately. Thus during the optimization process, the design needs not to be correct for all disciplines, but for the optimum it should, because then the constraints should be satisfied.

Related to this CO method is the Individual Discipline Feasible method (IDF). Here, auxiliary variables are defined for the problem to be optimized. These auxiliary variables enable the solution of the individual disciplines including the auxiliary variables. In this way, only a simple nonlinear optimization problem has to be solved (Alexandrov and Kodiyalam, 1998).

Furthermore, variants of the Bi-Level integrated System Synthesis (BLISS) exist as presented by Kodiyalam and Yuan, 2000. For these variant, system and discipline optimizations are carried out in an alternating way. The discipline constraints are described by Langrange multipliers used in the system. An advantage of BLISS (and also some of the other methods) is that discipline evaluations can be carried out parallel, thus limiting the time needed for the total optimization process.

Finally, Pareto surfaces can simply be drawn thus providing the engineer with helpful information or an optimal design can be generated fully interactively (Wuppalapati et al, 2007). Based on the above presented information, in the next paragraph it will be shown how the research engine is able to provide a frame-work for the procedures mentioned in this introduction.

**Research engine as frame-work**

The research engine is shown in figure 1 and in figure 2 on the left upper corner. The idea is to transform a spatial design into a structural design by selecting a transformation procedure (in the figure ‘Trans. selection’). Hereafter, the structural design is modified into another structural design (in the figure ‘Struct. design n+1’), again by selecting a transformation procedure. The structural design is transformed into a spatial design (‘Spatial design n+1’) and this spatial design is modified into a new spatial design (not shown in the figure, but it can be imagined as ‘Spatial design n+2’ replacing ‘Spatial design n’). Every time the spatial or structural design is validated with a measure as shown in the figure by ‘Measure’.

Now it is very important to notice that the initial but still most important aim of the research engine is to study the effect of the selection of a specific transformation (which can be seen as representing a part of a design process) on the measures (which can be seen as specific properties of the design). Thus the research engine does not represent an optimization procedure in any form; it only makes it possible to study objectively the effect of chosen design methods on the design properties, without using these properties for validating the design quality itself (Hofmeyer, 2007). This is what is meant by ‘Study of the design process’ in figure 2.

The next two processes, traditional building design and traditional mechanical design, can be described by using only a part of the research engine as shown in figure 2 (top row right and bottom row left) with black/bold lines. The two traditional processes do not show significant signs of optimization, e.g. the selection of a specific structural design for a spatial design is often based on the fact that the
structural design fits the spatial design, and not that it strengthens the spatial design. Thus, the research engine needs not to be changed and only parts of it can be used.

The next procedure is grammar based design. By selecting the elements of the research engine as shown in figure 2 at the bottom row right, using shape grammars for the transformation, and implementing a feed-back loop for the designs, grammar based design procedures can be modelled.

Structural optimization (figure 3, top row left) is somewhat more complex to be described by the research engine. Only the three entities on the right of the engine are set active, besides two measures...
that now are used also. Structural component $n$ is modified by a transformation in structural component $n+1$. The application of the transformation (and not the selection of the transformation mechanism) is a function of the measures that are taken from the structural components. For classical (gradient-based) optimization techniques, the modification is such that the modified structural component is steered towards a required value of the measure. However, shape annealing, as presented in the introduction, generates modifications without this steering, but determines a probability for which the modified
solution is preferred to the previous one for the next step, based on the new measure value. Genetic algorithms and ant colony strategies work in a similar way, as explained in the introduction.

Finally, multidisciplinary optimization is treated as shown also in figure 3. The MultiDisciplinary Feasible method can be regarded schematically by using all instances of the research engine. For each optimization step, the cyclic application of transformations is continued until the measure of the design is constant, which is equivalent to the feasibility for all disciplines (Alexandrov and Kodiyalam, 1998). Then, if a new optimization step is taken, the design variable values are estimated, using normal nonlinear programming techniques, as indicated in the figure at ‘Spatial design n’.

For Collaborative Optimization, the design instances ‘Spatial design n’ and ‘Struct. design n’ are regarded as feasible solutions for the discipline (Alexandrov and Kodiyalam, 1998). These can be used to determine the derivatives of the system constraints for a optimization step that uses a new prediction for the design variables based on these derivatives. This is shown in the figure at ‘Spatial design n’, but it is also possible to start with the structural design ‘Struct. design n’.

For BLISS, shown in the bottom row of figure 3 on the right, the figure is oversimplified because BLISS is too complex to be fitted correctly within the research engine. However, to illustrate the concept of BLISS, the figure shows an iteration with two alternating steps, a system optimization (for which the figure of MDF is used), and a disciplinary optimization (for which CO is used).

Summarized, most procedures in CAAD and CASD (Computer Aided Structural Design) can be conceptually understood using the research engine, although this engine was developed with the aim to research the design process. If it is realized that the research engine has this possibility, it is only a single step to use the research engine to generate new procedures. Some of these procedures may be ill-defined or far-fetched, some of them are worth to study further, as will be presented in the next section.
Research engine providing new applications

As was mentioned before, in this paper only spatial and structural design are taken into account, which is also the case for the idea of a research engine. To introduce other disciplines in the research engine, for instance constructional design, the research engine can be ‘multiplied’ as shown in figure 4. On the left the normal research engine is shown and on the right a research engine is shown for which constructional designs are used instead of structural designs. Both research engines run parallel and with the same speed. Every time they arrive at the left upper corner (spatial design), spatial design n of the spatial-structural engine and spatial design m of the spatial-constructional engine are compared and ‘averaged’. This averaging can be carried out in
several ways, but one of the possibilities would be to use a technique known from genetic algorithms: properties of the spatial designs are determined and rules should be developed that define the resulting property if two properties are combined. Then these rules are simply fired on the two spatial designs, and a new design results.

Using this approach, it is possible to combine many disciplines, simply by using more parallel research engines. Note however that they all should have a common design instance (for instance spatial design) that can be used to keep the engines connected. This means in figure 3 that a structural-constructional research engine on the right would also be possible, because then the structural designs could be ‘averaged’. In this way, a complete chain of research engines could be used, still yielding one design for each cycle. Note again that this approach is not related to optimization in any way, but studies the -now clearly multidisciplinary- measurable results of design approaches and transformations taken.

Another possibility is to let the research engine extend the possibilities of current used applications of structural optimization, which was first shown in figure 3 (top row, left). The first classic form is form-finding in which a geometry of a structural component is parameterized and optimal parameter values are found for minimal structural variables as energy, stress, etc., figure 5 top row left.

Using this form, the transformation (as shown in figure 3) consists of the change of geometry. The second form is topology optimization, figure 5 top row right. In this form, space is defined in which mass can be shifted from one place to another. This mass is then positioned optimally to distribute a force from the point of application to the supports. Here, the transformation (as shown in figure 3) consists of the relocation of structural mass. These two classic forms of structural optimization have been presented on the component level. However, they can be extended from component to building level (Hofmeyer, 1995) as shown schematically in figure 5. For this schematization, only a part of the research engine is active, as was already shown in figure 3. However, it is possible to set active the remaining parts of the research engine, as shown in figure 5 in the middle, which lead to new procedures. Examples of these procedures are shown at the bottom of figure 5 but only for the component level, as examples on the building level would be too complex to show. In the case of classic structural optimization (in the figure in the left column) nothing changes as the spatial design (the geometry) already changes due to the structural optimization. More precisely, the spatial design can be changed by its own transformation regardless the structural optimization changes, however, the structural optimization procedure will continue to evolve the design to its optimal geometry. For classic structural form-finding as shown in the right column, an interesting procedure results as structural mass is repositioned in a geometry with continuous changing boundaries. It is clear that structurally seen, this system will never converge to the optimal solution, however, if the design process is studied, interesting temporarily results may be generated.

Conclusions

A research engine carries out automated cycles of four types of transformations: (1) from spatial design to structural design, (2) from structural design to another structural design, (3) from structural design to spatial design, and finally (4) from a spatial design to another spatial design. Several procedures used for CAAD and CASD (computer aided structural design) can be regarded as derivatives of a research engine. In this way, the research engine provides a conceptual framework. The other way round, the available research engine can be modified and new applications result. For structural optimization, this leads to form finding with additional spatial constraints and topology optimization with changing spatial boundaries.
References


