*Environment and Planning B: Planning and Design* 2000, volume 27, pages 615 – 626

DOI:10.1068/b2619

# A micro language: generating MEMS resonators by using a coupled form – function shape grammar

**Manish Agarwal¶, Jonathan Cagan**
Computational Design Laboratory, Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA; e-mails: manish_agarwal@mckinsey.com; jcag@andrew.cmu.edu

**George Stiny**
Department of Architecture, Massachusetts Institute of Technology, Cambridge, MA 02139, USA; e-mail: stiny@mit.edu
Received 10 March 1999; in revised form 14 February 2000

**Abstract.** Shape grammars are shown to be capable of generating coupled form – function devices by first satisfying the minimum required functionality and then modifying the device to obtain the desired specifications. Advantages of a weighted $U_{22}$ algebra in conjunction with a $U_{12}$ grammar are also discussed. The approach is demonstrated by describing a shape grammar for the design of MEMS resonators.

## 1 Introduction

Shape-grammar-based generative systems have traditionally been used in architecture and, to a lesser extent, in engineering to create classes of individual designs. Most of the chosen application domains have a well-defined function – form decomposition thereby allowing the underlying generative engine first to satisfy the functional requirements and then to create physical forms based on the desired functional attributes. However, a large number of engineering products do not have an obvious form – function decomposition (Rinderle, 1986) making it difficult to model their configuration design process. Ironically it is this class of products that stands to gain the most from formal design tools because the lack of a predefined generation sequence typically limits human designs to only a small number of previously 'proven' configurations. Thus radical changes and rapid performance improvements are difficult and a small change in the desired specifications may require a significant time investment.

An example of such an application domain is the design of micro-electromechanical systems (MEMS). With the rapid increase in the use of MEMS devices it is becoming increasingly important to be able to generate these devices automatically and rapidly so as to be able to meet a wide variety of functional specifications. Current MEMS design methods rely mainly on the expertise of the designer and very few formal design techniques exist. This leads to a substantial turnaround time in the creation of a device because most designs are carried out manually. The designs created must usually be prototyped and tested to verify if they meet the functional specifications, further adding to the overall time needed to generate a new design. Moreover, and perhaps more critically, most manual designs have certain similar characteristics that may limit the functional requirements that can be met. For example, certain spring types such as folded flexures or crab legs are commonly used within microresonators thus putting a limit on the range of resonant frequencies that can be obtained in a device.

One of the reasons that make the design of MEMS devices difficult is the strong form – function coupling. A small change in the topology of the design may result in

¶ Current address: 21 South Clark Street, Suite 2900, Chicago, IL 60603-2900, USA.

significant changes in performance. It is also not at all clear how to choose the best topology for a given set of specifications. These characteristics of the design space lead us to believe that an automated design strategy capable of representing and searching through the space would substantially increase the range of specifications that can be met by these devices. We propose a design methodology for MEMS based on shape grammars and illustrate it by describing a grammar for the design of microresonators. The proposed shape grammar also addresses issues of form–function coupling and illustrates how those can be dealt with within the shape grammar paradigm. It is our expectation that such an approach would shorten and make more effective the design cycle for MEMS devices in particular, and engineering products in general.

The proposed grammar is a two-dimensional (2-D) parametric shape grammar involving plane surfaces and their boundaries. The grammar starts with the central mass and builds the resonator around it. Labels are used to ensure that the shape has at least one actuator and one spring before the generation process terminates, thereby ensuring that the language specified by the grammar consists only of valid designs. The generality in the grammar arises primarily from a generality in the variety of possible spring types. The current shape grammar is restricted to using only comb-drive resonators and to Manhattan geometries, though it can be extended to other cases. The grammar can recreate most of the existing comb-drive resonators found in the literature and is able to generate an infinite number of new ones.

The rest of this paper is organized as follows: first, we discuss briefly shape grammars and their traditional application domains; second, we present the shape grammar for microresonators; third, some implementation issues are discussed; and, finally, we conclude with some general observations.

## 2 Shape grammars

Shape grammars (Stiny, 1980) have been used successfully to generate a variety of architectural designs including villas in the style of Palladio (Stiny and Mitchell, 1978), Mughul gardens (Stiny and Mitchell, 1980), prairie houses in the style of Frank Lloyd Wright (Koning and Eizenberg, 1981), Greek meander patterns (Knight, 1986), suburban Queen Anne houses (Flemming, 1987), and windows in the style of Frank Lloyd Wright (Rollo, 1995). A shape grammar derives designs in the language that it specifies by successive application of shape transformation rules to some evolving shape, starting with an initial shape (I). In particular, given a finite set of shapes ($S$) and a finite set of labels ($L$), a finite set of shape rules ($R$) of the form $\alpha \rightarrow \beta$ transform a labeled shape $\alpha$ in $(S, L)^+$ into a labeled shape $\beta$ in $(S, L)^0$, where $(S, L)^+$ is the set of all labeled shapes made up of shapes in the set $S$ and symbols in the set $L$, and $(S, L)^0$ is the set that contains in addition to all of the labeled shapes in the set $(S, L)^+$ the empty labeled shape $<s_\phi, \phi>$. Shapes themselves can be transformed with Boolean operations. Parametric shape grammars are an extension of shape grammars in which shape rules are defined by filling in the open terms in a general schema. An assignment g that gives specific values to all the variables in $\alpha$ and $\beta$ determines a shape rule $g(\alpha) \rightarrow g(\beta)$ which can then be applied on a labeled shape in the usual way to generate a new labeled shape. Algebras of shapes can also be augmented by weights to obtain new algebras in which the shape union operation has been redefined to reflect different possible weights on different entities (Stiny, 1992). This work takes advantage of labels, weights, and parametric shapes in the derivation of a concise shape grammar for MEMS resonators.

There has been limited application of shape grammars to engineering design. Fitzhorn (1990) and Longenecker and Fitzhorn (1991) present shape grammars specifying the languages of constructive solid geometry and boundary representations

(that is, realizable solids). Brown et al (1993) present a manufacturing-oriented shape grammar that specifies the language of all axisymmetric objects manufacturable on a given lathe. Reddy and Cagan (1995), Shea et al (1996), and Shea and Cagan (1999) present a parametric shape grammar for the design of truss structures that uses the shape annealing technique of Cagan and Mitchell (1993) to generate optimal truss structures. Shea and Cagan (1997) also use the shape annealing technique to generate and optimize dome designs. Agarwal and Cagan (1998) present a shape grammar for the design of coffeemakers, the first to apply shape grammars to a class of individual consumer products. However, a clear function – form decomposition exists in that grammar. This work applies the shape grammar paradigm to coupled form – function devices.

## 3 Microresonator grammar

### 3.1 MEMS resonators

MEMS devices are micron-sized electromechanical structures fabricated by using IC technology. They integrate mechanical elements such as beams and springs with electrical circuitry and can be used in both the sensing and the actuation modes. MEMS devices have been used in a variety of applications, namely air-bag sensors, pressure sensors, and gyroscopes. Resonators are second-order mass – spring – damper systems with silicon serving as a proof mass, the ambient air as the damper, and beam elements as springs. Actuation is typically provided with electrostatic forces by elements in either the parallel plate mode or the fringe field mode. The grammar discussed here uses comb drives (a fringe field element) for actuation. Comb drives consist of interdigitated fingers that provide a mechanical force along the direction of motion. To function as valid devices, resonators must have at least one actuator and one spring in addition to the proof mass. Note that a valid device may not meet all the desired specifications but is guaranteed to function as a resonator. Figure 1 (see over) shows three typical resonators.

### 3.2 Resonator grammar

3.2.1 *Algebra*

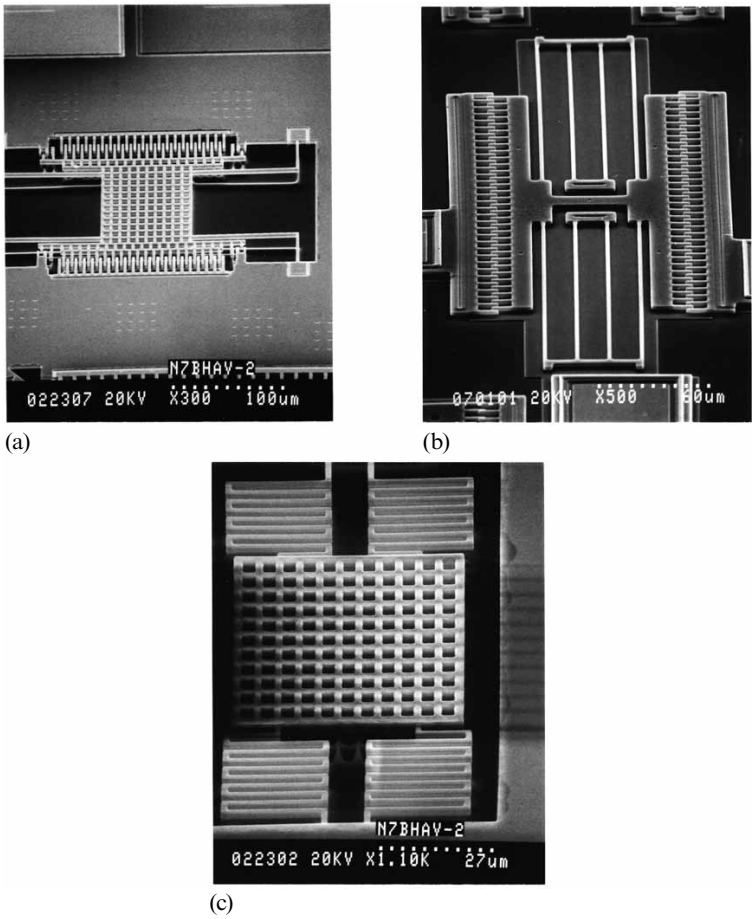
The proposed shape grammar for MEMS resonators is a parametric labeled 2-D grammar augmented by weights and labels. Formally the grammar is a $W_{22} \times U_{12} \times V_{02}$ grammar. The terminology used here follows that defined by Stiny (1992). In general, $X_{ij}$ refers to a grammar that has elements of dimension $i$, embedded in a space of dimension $j$. $U$, $V$, and $W$ refer to grammars in which shapes consist of only geometric information, geometric information and labels, and geometric information and weights, respectively. Note that a 2-D representation is sufficient because MEMS resonators are 2.5-D devices. However, $U_{12}$ and $V_{02}$ components are needed in the grammar along with the $W_{22}$ component to preserve boundary information.

The four main elements of a resonator are the central mass, actuators, anchors, and springs. Each of these is assigned a different weight and the shape addition operation for two shapes $A$ and $B$ is defined as the regular shape addition of Stiny (1992) for each subalgebra in the Cartesian product making up the grammar along with the following definition for the addition of weights:

$$\text{weight}(u) + \text{weight}(v) = \text{maximum}[\text{weight}(u), \text{weight}(v)] .$$

The shape difference operation for two shapes $A$ and $B$ is defined as the regular shape difference operation of Stiny (1992) for each subalgebra along with the following definition for the difference of weights:

$$\text{weight}(u) - \text{weight}(v) = 0 .$$




(a) (b)

(c)

**Figure 1.** Scanning electron micrographs (SEMs) of three typical resonators: (a) crab leg; (b) folded flexure; and (c) serpentine.

The subshape relation is defined in the regular way for the $U_{12}$ and $V_{02}$ components following Stiny (1992). The subshape relation for the $W_{22}$ component is also defined based on Stiny (1992). However, owing to the specific semantic meaning associated with the weights in this work, we require two elements with an embedded maximal element to have the same weight if they are to be subshapes of one another. Formally, if $A$ and $B$ are shapes in the $W_{22}$ algebra and $(e, u)$ and $(f, v)$ are ordered pairs of maximal elements and weights in $A$ and $B$, respectively, then the following subshape definition is obtained: $A \leqslant_W B$ whenever both $A^* \leqslant_U B^*$, and, if $e$ is embedded in $f$, then $u = v$.[(1)] Note that this definition uses the notation of and parallels the one developed by Stiny. Because the mass can be created in only one way (as the initial shape) it is desirable that the mass not be broken up by actuators, anchors, or springs. Hence it is assigned the highest weight of 4. Similarly actuators can also be created in only one way, though on each side of the mass; they are assigned a weight of 3. Springs are the least constrained of the four elements and are assigned a weight of 1. Finally anchors are assigned a weight of 2. The shape addition operator discussed above needs to be modified slightly; this modification is discussed later.

[(1)] Loosely speaking, $A^*$ is the shape formed from $A$ by disregarding the weight information.

3.2.2 *Initial shape rules*

Because microresonators consist of an actuated mass attached to springs anchored to the substrate, the grammar starts with the central mass as the initial shape and adds actuators and springs to complete the design. The initial shape is the rectangle shown in figure 2. Figure 2(a) shows the various parts of the shape on which the three components of the algebra ($W_{22}$, $U_{12}$, and $V_{02}$) act. Figure 2(b) shows all the parts of the shape together in one shape (the shading is not shown for clarity). Note that, whereas the representation in figure 2(b) will be used to depict the rules in the grammar, both the left-hand and right-hand sides of the rules actually consist of the three parts shown in figure 2(a).

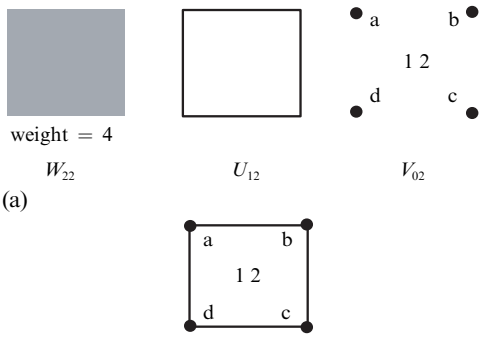



**Figure 2.** Initial shape for the grammar: (a) various subparts; and (b) combined shape.

The points labeled, a, b, c, and d are parametric points. Labels 1 and 2 are global labels on the shape that are used later in the generation process to determine if an actuator and a spring have been attached to the mass. Note that throughout this paper parametric points are shown by a solid circle and an associated label. Also, from a theoretical perspective, global labels may be assumed to be associated with a point at the center of the mass, and labels on lines may be assumed to be associated with their respective center points. The first set of rules, called the *initial rules*, is shown in figure 3. Rule 1 attaches labels to the four sides of the mass to designate if a side is constrained to have at least an actuator attached to it (A), at least a spring attached to it (S), or if the side is unconstrained (U). This rule also removes the labels a, b, c, and d. Note that the labels $L_1$, $L_2$, $L_3$, and $L_4$ are parametric labels and

$$L_i \in \{A, S, U\}.$$

Rule 2 allows the parametric label X to change to another parametric label Y where

$$X \in \{A, S, U\}, \quad Y \in \{A, S\}.$$

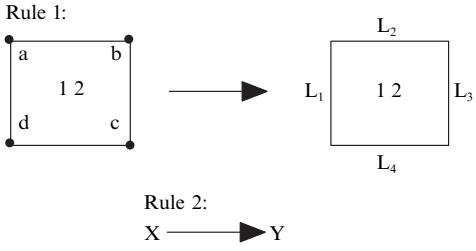



**Figure 3.** Intitial rules for the grammar.

Different instantiations of this rule allow the labels A and S to be interchanged and the label U to be converted to the labels A and S. This rule must be applied if the labels $L_1$, $L_2$, $L_3$, and $L_4$ are instantiated such that they do not contain both an A and an S. The rule can then be used to convert labels such that there is at least one A and one S, ensuring that at least one actuator and one spring would be created.

3.2.3 *Initial design rules*

The next set of rules, shown in figure 4, are called the *initial design rules*. These rules create one actuator and one spring based on the labels associated by the initial rules. Labels are used to ensure that both an actuator and a spring are created before the generation process can continue forward. Rule 3 attaches an actuator to the side of the mass labeled A. If more than one side has the label A, any one of those can be chosen for the application of this rule. This rule requires the labels 1 and 2 to be present on the shape and it changes the label 1 to 1′. It also deletes the label A from the side the rule is applied on. This ensures that the rule can only be applied once. Further, as subsequent rules require the label 1′, this rule must be applied. Note that the comb-drive actuator created on the right-hand side is also a parametric shape. A part of the actuator has been magnified and shown in figure 5(a). Rules 4 – 7 create springs on a side labeled S. The springs are also parametric and can be either a simple beam (rule 4), a crab leg (rule 5), a folded flexure (rule 6), or a serpentine (rule 7). The number of bends in the serpentine spring is parametric and is denoted by the label N, where N represents the number of half bends connected in the spring in a sequence of upright and flipped configurations. A half bend is shown in figure 5(b). These particular spring shapes were chosen because they occur frequently in existing designs and can be combined to generate arbitrary springs. The dark rectangles on the springs represent the anchors,

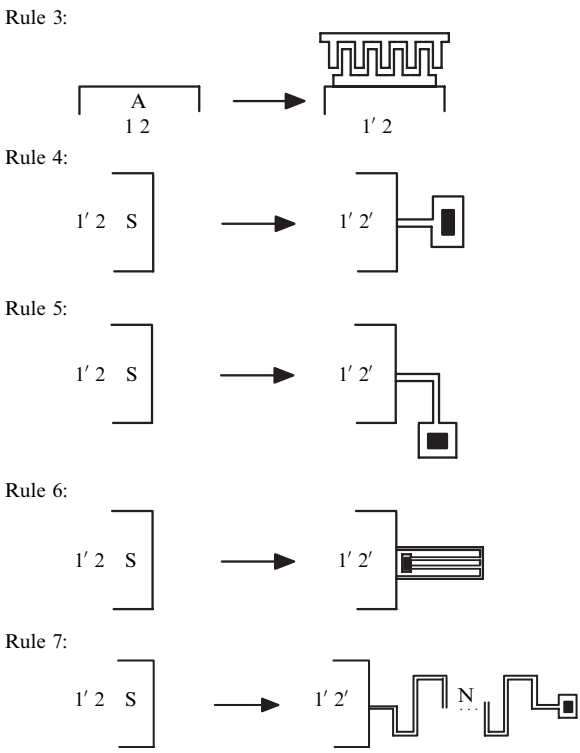



**Figure 4.** Initial design rules.

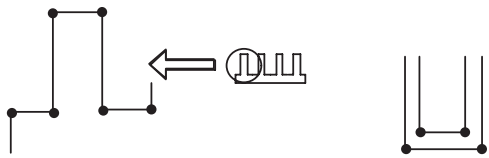



**Figure 5.** (a) Magnified view of a part of the comb-drive actuator, and (b) half bend of a serpentine spring.

that is, the locations where the springs are attached to the substrate. Each of these rules requires the labels 1′ and 2 to be present on the shape, changes the label 2 to 2′, and deletes the label S on the side it applies to. Subsequent rules require the label 2′ to be present on the shape to be applied. This ensures that one of the rules 4 – 7 is applied before the generation process continues. Also, note that at the end of this step the shape represents a valid microresonator.

3.2.4 *Augmentation rules*

The next rule called the *actuator design rule*, shown in figure 6, is similar to rule 3 except that it requires the labels 1′ and 2′ to be present on the shape and not the labels 1 and 2. The actuators designed by this rule are not necessary for the device to function as a resonator (as opposed to the actuator designed by rule 3); rather they modify the behavior of an existing resonator by changing the operating parameters such as resonant frequency and amplitude at resonance.

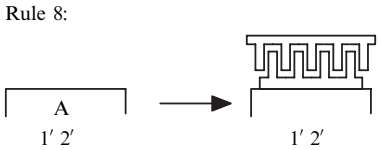



**Figure 6.** Actuator design rule.

The next set of rules, called the *spring design rules*, are shown in figure 7 (see over). These rules create and modify springs to try and meet the desired device functionality. Springs can be created in one of two ways—by directly adding macro shapes (crab leg, folded flexure, and serpentine) or by adding and modifying simple beam elements, even within macro shapes. Note that the springs can be added to any side of the mass, or even to existing springs. This gives the design process additional generality not commonly seen in existing designs which use relatively simple shapes. All of these rules require the labels 1′ and 2′ to be present. Thus, again, they modify existing resonators to change their behavior. Rule 9 adds a simple beam, rule 10 a crab leg, rule 11 a folded flexure, and rule 12 a serpentine. Again, the label $N_1$ on the right-hand side of rule 12 signifies the number of half bends. Rule 13 modifies an existing spring by adding a simple beam element to it, and rule 14 removes a beam element from an existing spring. All the shapes in the rules are parametric.

However, by using the shape addition operation defined earlier, it is possible that the spring modification rules be applied such that an existing spring loops back and its anchor is overlapped by the mass or an actuator. As an example, consider the design sequence shown in figure 8 (see over). Not all the steps or labels are shown and the figure is not to scale.

In such a situation, the 'spring' would cease to function as a spring and just serve to add extra mass to the device. Thus it is possible to create devices that do not have a functional spring on them. This problem is overcome by redefining the shape addition

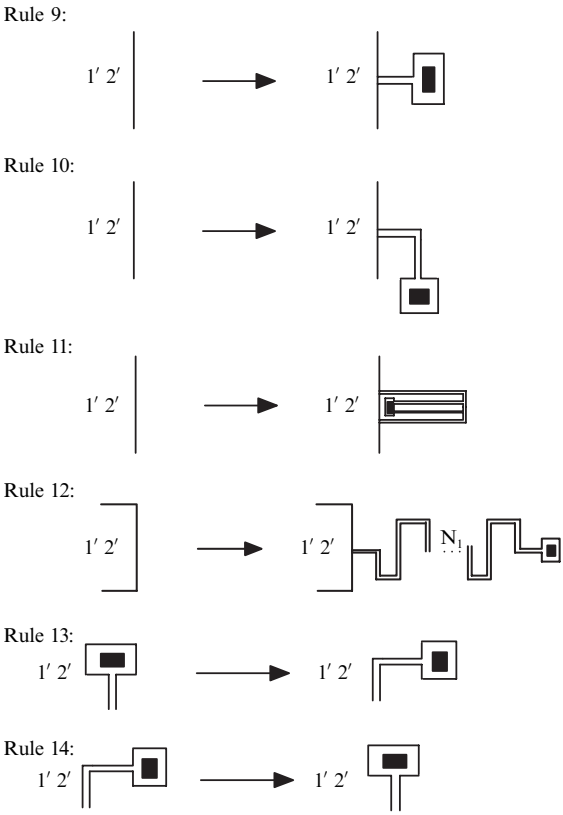



**Figure 7.** Spring design rules.

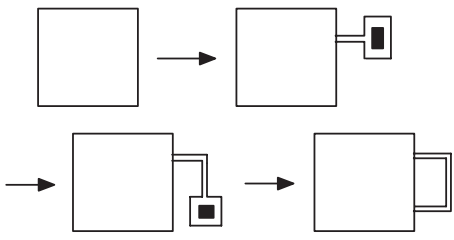


**Figure 8.** Sample design sequence that generates a loop.

operator for two shapes $A$ and $B$ in $W_{22}$ to prevent such a situation from arising. If weight($A$) $\in \{3, 4\}$ and weight($B$) $\in \{3, 4\}$, that is, the mass and the actuators overlap each other, the definition is the same as before. The definition is also the same if weight($A$) $\in \{1, 2\}$ and weight($B$) $\in \{1, 2\}$, that is, a spring or an anchor overlaps another spring or anchor. However, if either a spring or an anchor overlaps an actuator or the central mass, that is, weight($A$) $\in \{1, 2\}$ and weight($B$) $\in \{3, 4\}$, the addition operator, $\oplus$, is defined as:

$$A \oplus B \;=\; (A + B) + \mathrm{t}_A(C)\,,$$

where $C$ is an anchor placed at the end of the spring. The notation $\mathrm{t}_A(C)$ is used to express the fact that the location of the added anchor, $C$, depends on the location of shape $A$, that is the shape of the spring or the anchor, as the case may be. Formally

shape $A$ defines a transformation, $t_A$, that must be applied to shape $C$ before it can be added to shape $(A + B)$. Shape $C$ is transformed by $t_A$ from an arbitrary, but predefined, position (say, the global origin) to just outside the mass and at the end of the corresponding spring. Note that the operator $\oplus$ is still a binary operator because it takes only $A$ and $B$ as inputs. The rules for the addition of the weights remain the same. Also noteworthy is the fact that this modification to the definitions takes care of both partially and completely occluded anchors. With this new addition definition, the generation sequence of figure 8 would change to the one shown in figure 9. An alternative approach to obtaining the same result is to write a new spring modification rule that checks to see if shapes of weight 1 or 2 overlap with shapes of weight 3 and 4 as a precondition to its application. The right-hand side of such a shape rule would then explicitly add the anchor at the end of the spring to eliminate the overlap, thereby resulting in the same generation sequence as is obtained by using the modified addition operation.

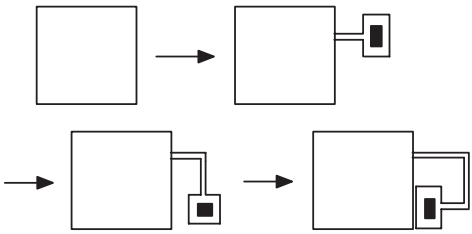


**Figure 9.** Sample generation sequence with the modified addition rule.

3.2.5 *Termination rule*

The final rule, called the *termination rule* shown in figure 10, changes the labels $1'$ to $1''$ and $2'$ to $2''$. This ensures that no further rules can be applied and the design process terminates.

Rule 15:

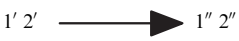



**Figure 10.** Termination rule.

## 4 Example

In this section we discuss a sample design sequence to demonstrate the application of the grammar. The process starts with an initial shape. For the example discussed here it is a square shown in figure 11(a). Next, rule 1 is applied to the shape to put labels A, A, S, and U on the shape as shown in figure 11(b). Then, based on label A, an actuator is added to the right-hand side of the mass by using rule 3. A folded flexure spring is then added to the top of the mass by using rule 6. The device at this stage is a functional resonator having a mass, a spring, and an actuator, and is shown in figure 11(c). Next, by means of rule 8, another actuator is added to the left-hand side of the shape and another folded flexure added to the bottom of the mass by using rule 11. The shape generated is shown in figure 11(d). Finally rule 15 is used to terminate the design generation process. The final design obtained is shown (without the labels) in figure 11(e). This design is schematically similar to the SEM of an actual device shown in figure 1(b).

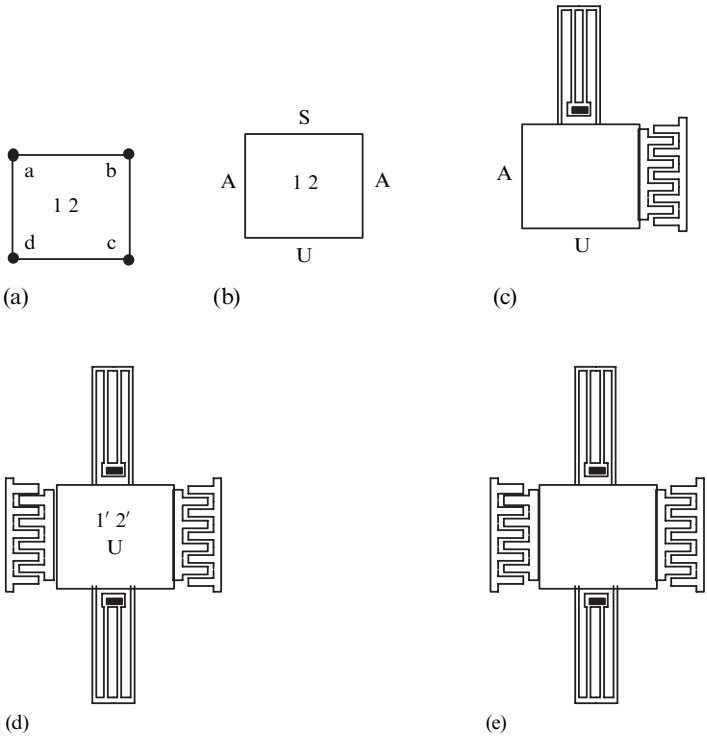



**Figure 11.** Example of a generation sequence.

The other devices shown in figure 1 can also be recreated by the grammar and are shown in figure 12(a). The grammar can also be used to create an infinite number of new designs, three of which are shown in figure 12(b).

## 5 Implementation

In this section we comment on the implementation aspects of the grammar. The grammar is implemented in the LISP programming language. However, a $V_{12}$ grammar is used for implementation rather than the grammar presented here. This is primarily because of the ease of implementation of $V_{12}$ grammars as discussed in Krishnamurti (1980; 1981). The languages of shapes generated by both the grammars, however, are equivalent. The rules for the $V_{12}$ grammar are significantly more complicated than those discussed here because labels on lines must be used to distinguish between the four main elements—springs, anchors, actuators, and mass. In addition, labels are required to distinguish between the inside and the outside of a shape to ensure that only valid designs are produced. For example, a spring cannot be created inside a mass but can be created on the outside. This is not relevant in the grammar presented in this paper because the addition operator can be used to ensure validity of designs. All of these changes result in a less succinct $V_{12}$ grammar (fifty-three rules as against fifteen presented here). However, the complication in the rules of the $V_{12}$ grammar is offset by the ease of implementation.

## 6 Conclusions and future work

The grammar discussed in this paper extends the shape grammar paradigm to the design of MEMS resonators as an example of the design of coupled form–function devices. The grammar achieves this coupling by abstracting the minimum functionality

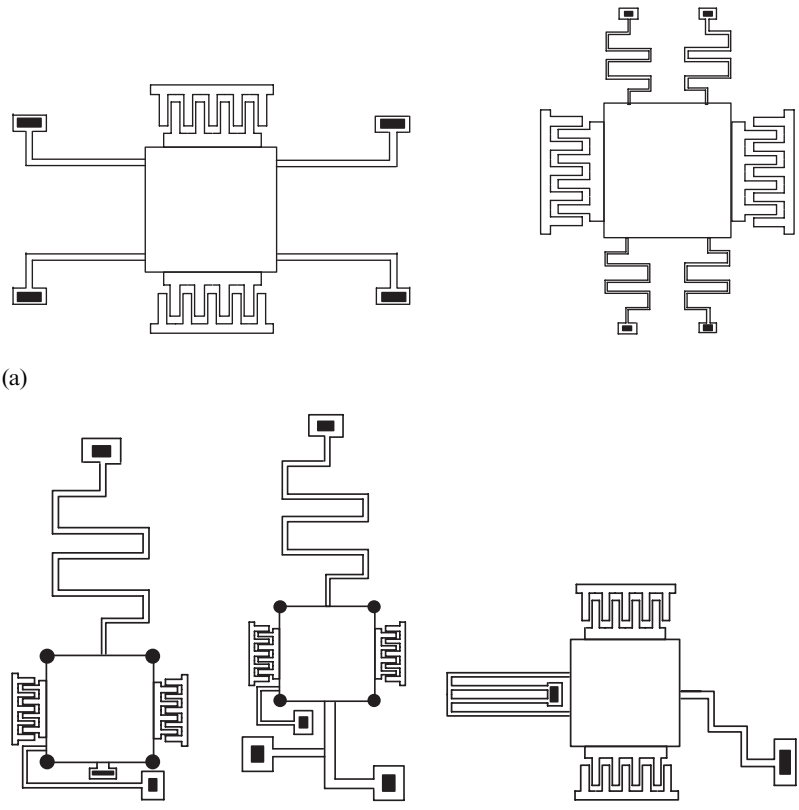


**Figure 12.** Designs generated by the grammar: (a) traditional; (b) novel.

required for a valid device (a mass, an actuator, and a spring in the case of resonators) and ensuring that those attributes are met before other aspects of the design (additional actuators and springs, for example) are created. Such an approach results in a language that consists entirely of valid designs and yet can support a wide range of specifications. Current work focuses on exploring this space by using powerful optimization algorithms to identify the set of effective designs for any particular application.

The other important attribute of the grammar is the fact that it uses a $W_{22}$ algebra in a fundamental way. This allows us to simplify the shape rules greatly and makes the grammar more succinct. Perhaps more importantly, it models more closely the physical design of a resonator that employs overlapping *planes* of different materials to create the final device. We believe that this makes the grammar more intuitive and easy to use for the designers. In terms of the theoretical contribution, this is the first published shape grammar that effectively uses weights and a 2-D representation. The result is a significant reduction in the number of rules. The shape grammar can be further augmented and made more physically based by associating functional properties of springs and actuators with the corresponding shapes. Such an approach would associate additional meaning with what are otherwise abstract geometrical planes and also aid in the analysis of the designs.

Finally, the approach outlined in this paper provides a new framework for the design of MEMS devices. Traditionally, design of micromechanical structures has been somewhat ad hoc and has relied heavily on designer experience leading to a

significant turnaround time for some designs. Developing a formal, implementable design representation capable of generating a wide variety of devices is the first step in aiding the designer and automating the design creation process. This work presents such a representation for MEMS resonators; similar representations for other devices such as accelerometers and gyroscopes are also possible and would result in an easier and quicker design cycle for MEMS devices.

**Acknowledgements.** This work is supported by the National Science Foundation under grant # DMI-9713782. The research effort was also sponsored by the Defense Advanced Research Projects Agency (DARPA) and Rome Laboratory, Air Force Material Command, USAF, under agreement number F30602-96-2-0304 and by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number F49620-98-1-0172. The US Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, Rome Laboratory, AFOSR or the US Government.

**References**

Agarwal M, Cagan J, 1998, "A blend of different tastes: the language of coffeemakers" *Environment and Planning B: Planning and Design* **25** 205 – 226

Brown K N, McMahon C A, Sims Williams J H, 1993, "A formal language for the design of manufacturable objects", in *Formal Design Methods for CAD (B-18)* Eds J S Gero, E Tyugu (North-Holland, Amsterdam) pp 135 – 155

Cagan J, Mitchell W J, 1993, "Optimally directed shape generation by shape annealing" *Environment and Planning B: Planning and Design* **20** 5 – 12

Fitzhorn P A, 1990, "Formal graph languages of shape" *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* **4**(3) 151 – 164

Flemming U, 1987, "More than the sum of the parts: the grammar of Queen Anne houses" *Environment and Planning B: Planning and Design* **14** 323 – 350

Knight T W, 1986, "Transformations of the meander motif on Greek geometric pottery" *Design Computing* **1** 29 – 67

Koning H, Eizenberg J, 1981, "The language of the prairie: Frank Lloyd Wright's prairie houses" *Environment and Planning B* **8** 295 – 323

Krishnamurti R, 1980, "The arithmetic of shapes" *Environment and Planning B* **7** 463 – 484

Krishnamurti R, 1981, "The construction of shapes" *Environment and Planning B* **8** 5 – 40

Longenecker S N, Fitzhorn P A, 1991, "A shape grammar for non-manifold modeling" *Research in Engineering Design* **2** 159 – 170

Reddy G, Cagan J, 1995, "An improved shape annealing algorithm for truss topology generation" *ASME Journal of Mechanical Design* **117**(2A) 315 – 321

Rinderle J, 1986, "Implications of function – form – fabrication relations on design decomposition strategies", in *Computers in Engineering: Proceedings of the ASME International Computers in Engineering Conference and Exhibition* (American Society for Mechanical Engineers, New York)

Rollo J, 1995, "Triangle and T-square: the windows of Frank Lloyd Wright" *Environment and Planning B: Planning and Design* **22** 75 – 92

Shea K, Cagan J, 1997, "Innovative dome design: applying geodesic patterns with shape annealing" *Artificial Intelligence in Engineering Design, Analysis and Manufacturing* **11** 379 – 394

Shea K, Cagan J, 1999, "The design of novel roof trusses with shape annealing: assessing the ability of a computational method in aiding structural designers with varying design intent" *Design Studies* **20** 3 – 23

Shea K, Cagan J, Fenves S J, 1996, "A shape annealing approach to optimal truss design with dynamic grouping of members" *ASME Journal of Mechanical Design* **119**(3) 388 – 394

Stiny G, 1980, "Introduction to shape and shape grammars" *Environment and Planning B* **7** 343 – 351

Stiny G, 1992, "Weights" *Environment and Planning B: Planning and Design* **19** 413 – 430

Stiny G, Mitchell W J, 1978, "The Palladian grammar" *Environment and Planning B* **5** 5 – 18

Stiny G, Mitchell W J, 1980, "The grammar of paradise: on the generation of Mughul gardens" *Environment and Planning B* **7** 209 – 226