INSTITUTE OF PHYSICS PUBLISHING SMART MATERIALS AND STRUCTURES
Smart Mater. Struct. **14** (2005) 869–875 doi:10.1088/0964-1726/14/4/047

Actuator placement optimization in a Kagome based high authority shape morphing structure

S L dos Santos e Lucato, R M McMeeking and A G Evans

Materials Department, University of California, Santa Barbara, CA 93106-5050, USA

Received 19 August 2004, in final form 11 January 2005
Published 28 July 2005
Online at stacks.iop.org/SMS/14/869

Abstract
An optimization protocol for a high authority shape morphing plate is described. The shape morphing design incorporates an active back-plane comprising a Kagome truss, capable of changing the shape of a solid face, connected to the back-plane by means of a tetrahedral truss core. Several members of the Kagome truss are replaced by actuators, enabling the structure to deform. The trusses to be replaced depend on the desired deformation, subject to the load capacity of the individual actuators. A two-level scheme is used comprising a heuristic algorithm with a simplex based optimization providing the cost function. It is shown that methods capable of avoiding entrapment in local minima, such as simulated annealing and the genetic algorithm, give good results.

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Shape morphing structures are designed to displace surfaces while being resisted by large pressure loads (or heavy weights) [1–6]. One approach for addressing this challenge is to use the Kagome structure depicted in figure 1 [2, 3, 6]. The basic structure consists of a solid face-sheet with a Kagome back-plane and a tetrahedral core. The configuration is rigidly supported at one end. Replacing various truss elements in the back-plane with linear actuators enables the shape of the solid face to be changed.

While this structure can undergo a wide range of deformations, it is limited by yielding and buckling failure of the passive structure as well as the load specification of the actuators. Previous investigations concluded that the deformations are limited by the incorporated actuators rather than by the structure itself [6–8]. Structural failure, if it is the limiting factor, can easily and cheaply be remedied by making it sturdier. The inherent increase in weight and resistance against actuation is small compared to the increase in load capacity of the structure [7]. In contrast, incorporation of more actuators or higher authority actuators to overcome actuation as the limiting factor usually results in a significant increase in weight and system complexity. It is therefore desirable to reduce the number of actuators to limit the overall cost and complexity of the system. Placement of a limited number of actuators becomes a crucial design aspect.

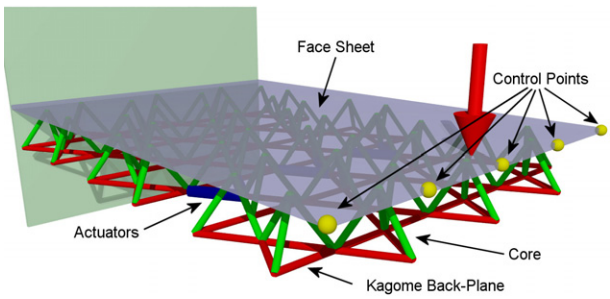

Figure 1. Schematic representation of the Kagome structure consisting of the face-sheet, the core and the Kagome back-plane. Actuators are placed in lieu of the Kagome members. The control points are used to define the target deformation.

The objective of the present study is to find actuator configurations that maximize the deformation. The procedure selects configurations of actuators and ascertains the quality of every configuration by a second (local) optimization. The results of the latter are used as cost functions for the global optimization that determines the best configuration for a limited number of actuators. Some key characteristics required for the optimization are calculated by means of a finite element

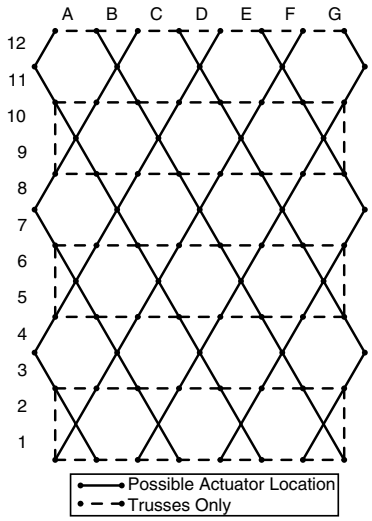

Figure 2. Kagome layer of the structure used in this study. Possible locations of actuators in the Kagome layer are shown as solid lines.

analysis. Here, the length of the panel was chosen to include six hexagonal units of the Kagome plate, while the width incorporated four, resulting in 96 possible actuator locations (figure 2). Multiple virtual control points define the shape change to be achieved [6].

The search space for global optimization is discrete and unordered, excluding the use of gradient based algorithms. Moreover, probing the entire search space is prohibitive. Optimization problems of this nature have been solved with heuristic methods, such as simulated annealing and genetic algorithms [9, 10]. The purpose of this study is to assess the performance of these algorithms in actuator placement optimization with the objective of actuating the structure to a final shape as close as possible to a predefined shape.

2. Local optimization and the cost function

The *local optimization* is devised to find the largest possible deformation for a given set of actuators subject to external constraints, e.g. the load limit of the actuators. For local optimization, n points are identified along the solid face (figures 1, 3). The vertical displacement $(v_i, i = 1, \ldots, n)$ of each point is controlled and maximized by selecting actuator strains $(\varepsilon_j, j = 1, \ldots, m)$ for m independent actuators (where $m > n$). The redundancy will be used to optimize the deformation within the force capabilities of the actuators: a crucial requirement for *actuator-limited* structures [6]. For the unloaded structure, a matrix $\boldsymbol{A}$ of influence coefficients can be constructed such that

$$v_i = \sum_{j=1}^{m} A_{ij}\varepsilon_j. \tag{1}$$

The null-space of the matrix $\boldsymbol{A}$ has dimension $m - n$ and consists of combinations of actuator strains giving rise to zero displacements at the control points along the edge to be twisted [11]. One orthonormal basis for the null-space is the set of $m - n$ vectors ε_j^k, $k = 1, \ldots, m - n$, such that

$$\sum_{j=1}^{m} A_{ij}\varepsilon_j^k = 0. \tag{2}$$

These basis vectors ε_j^k can be found from equation (2) by standard matrix manipulations [12]. Now calculate projections of the actuator strain array into the null-space of $\boldsymbol{A}$:

$$w^k = \sum_{j=1}^{m} \varepsilon_j^k \varepsilon_j. \tag{3}$$

Therefore, w^k, $k = 1, \ldots, m - n$, represent the degrees of freedom for zero displacement of the control points. Now introduce a matrix $\boldsymbol{B}$ defined as

$$B_{ij} = \left\{ \begin{array}{c} A_{ij}, i = 1, \ldots, n \\ \varepsilon_j^{i-n}, i = n+1, \ldots, m \end{array} \right\}. \tag{4}$$

It follows that

$$\sum_{j=1}^{m} B_{ij}\varepsilon_j = \left\{ \begin{array}{c} v_i, i = 1, \ldots, n \\ w^{i-n}, i = n+1, \ldots, m \end{array} \right\}. \tag{5}$$

Since all its rows are linearly independent, the matrix $\boldsymbol{B}$ is non-singular and inversion of equation (5) provides

$$\varepsilon_j = \sum_{i=1}^{n} B_{ji}^{-1} v_i + \sum_{i=n+1}^{m} B_{ji}^{-1} w^{i-n}. \tag{6}$$

The structure of (6) shows that a deformation can be achieved by specifying the n displacements, v_i, for the control points, plus any values for the remaining $m - n$ parameters, w_k, because the latter do not affect the control point displacements.

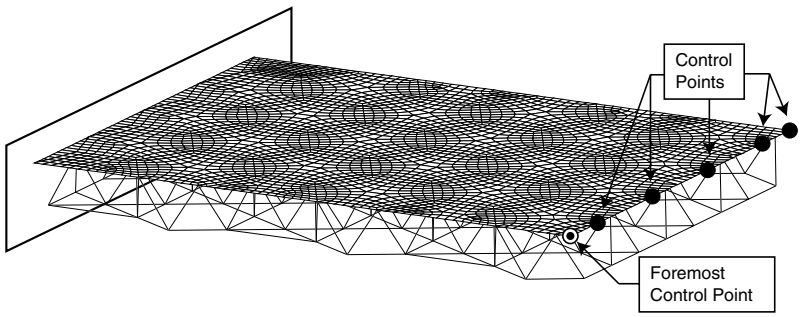

Figure 3. Finite element model of the Kagome structure. Six control points used in this study are highlighted. The foremost open control point is used to discuss the quality of the results.

Now maximize the deformation without exceeding the force limits of the actuators. In the absence of a load, the relationship between the actuator forces $P_k, k = 1, \ldots, q$, and the actuator strains $\varepsilon_i, i = 1, \ldots, m$, can be established as

$$P_k = \sum_{i=1}^{m} C_{ki}\varepsilon_i. \tag{7}$$

Note that q may be larger than m because actuators may be operated in pairs or sets to have the same actuator strain, but each actuator in general experiences different force levels, even when they are paired or put together in sets. That is, replacing two adjacent truss members with actuators being actuated to exactly the same strain results in such a scenario. Combining (6) and (7) gives

$$P_k = \sum_{j=1}^{m} C_{kj}\left(\sum_{i=1}^{n} B_{ji}^{-1}v_i + \sum_{i=n+1}^{m} B_{ji}^{-1}w^{i-n}\right). \tag{8}$$

The control point displacements can be expressed in terms of a single degree of freedom, ϕ, representing the deformation:

$$v_i = \phi u_i \tag{9}$$

where u_i are the displacements for unit value of ϕ. As a result, (9) becomes

$$P_k = \beta_k\phi + \sum_{i=1}^{m-n} D_{ki}w^i \tag{10a}$$

where

$$\beta_k = \sum_{j=1}^{m}\sum_{i=1}^{n} C_{kj}B_{ji}^{-1}u_i \qquad \text{and} \qquad D_{ki} = \sum_{j=1}^{m} C_{kj}B_{ji+n}^{-1}. \tag{10b}$$

The remaining task is to maximize ϕ in (10*a*) subject to minimum and maximum constraints on the actuator forces P_k. It is a straightforward linear programming problem [11] to find the $m - n$ coordinates of w^i. Once solved, the resulting values of ϕ and w^i are inserted into (9) and (10*a*) to compute the actuator strains that produce the maximum possible deformation within the actuator force constraints.

2.1. The cost function

The global optimization routines in the present work base their actuator selection criteria on some measure of the quality of the present actuator configuration. Therefore they require an externally computed cost function for assessing the quality of any given actuator configuration. In the present study the quality measure, i.e. the cost function, accounts for the difference between the optimized deformation and the desired deformation.

The above local optimization algorithm calculates the largest possible deformation for a single actuator configuration, e.g. the present one. The resulting deformation parameter ϕ as calculated using (10*a*) increases with increasing deformation and would provide a good measure of the achievable deformation. However, in our present computer implementation of the local algorithm, the deformation vector v_i is extended by w^i to form a generalized deformation vector as shown on the right-hand side of equation (5). Therefore ϕ cannot be used directly because it depends on the null-space of $\boldsymbol{A}$ and hence the actual actuator set. Instead the root mean square of the difference between the defined target deformation u_i and the optimized deformation v_i is used as an absolute measure:

$$f = \sqrt{\sum_{i=1}^{n}(u_i - v_i)^2}. \tag{11}$$

This cost function measures the quality of the optimized deformation relative to the target deformation. Note that with this choice of cost function, the target deformation should be chosen sufficiently large, since a penalty is imposed on solutions yielding a deformation larger than that defined. Although this is not a problem per se, it reduces the safety margin of the final assembly by requiring it to operate at its limit.

2.2. Finite element analysis

The matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ are calculated by a finite element analysis. The mesh, depicted in figure 3, consists of linear beams and linear four-point shell elements. The calculations have been performed with the commercial FEA package ANSYS. The actuator displacements have been modeled by thermally expanding the beam elements by 1%. One FE calculation was performed for every possible (active) actuator position. Only one actuator was active in each run, resulting in 96 calculations. The stresses on all members are transposed to a file. Additionally, the displacements at 60 separate locations along the solid face were transposed. The control points for the local optimization were chosen from among those locations. The ANSYS script language APDL was used to automate the task.

3. Global optimization algorithms

In this section, several algorithms are used to determine the best possible placement of a limited number of actuators, based on a prescribed set of target deformations. The load limits of the actuators are incorporated into the optimization by means of the local routine. All possible actuator locations are numbered (shown as solid lines in figure 2). The following algorithms invoke one (or more) actuator sets, each consisting of m actuators. The number m is predetermined and is not part of the optimization. The following steps are common. The number of actuators and the desired target deformations are defined before the analysis is started. To calculate the cost function for a specific set of actuators, the matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ for the active set are assembled and passed to the local optimizer. Only the active control points and the active actuators are included. Completeness of the original files is required since any actuator position may become active during the optimization. All global and local algorithms are implemented in a custom program written in Borland Delphi.

3.1. The random algorithm

The random algorithm is used as a benchmark. All m actuators are chosen randomly in every step. The best solution is the only information retained from previous steps (figure 4). After the

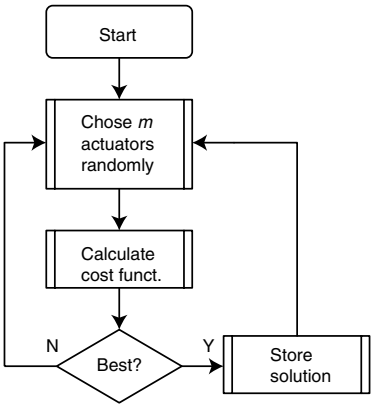

Figure 4. Flow diagram of the random search algorithm.

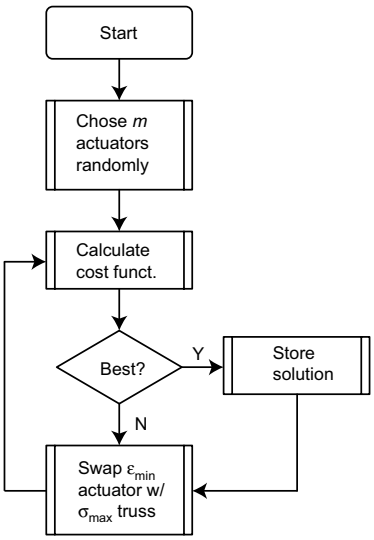

Figure 5. Flow diagram of the iterative replacement algorithm.

actuators are chosen, the cost function is calculated. If the cost is lower than that of the best previous solution, it is stored.

3.2. Iterative replacement

This algorithm utilizes some information inherent to the structure in addition to the cost function. The passive member subject to the highest stress is determined, along with the actuator that undergoes the least extension. Reasoning that the former position is most likely to benefit from being extended and the latter the least amenable to actuation, the two positions are interchanged. The new configuration is transitioned onto the local optimizer and the procedure repeated. After every pass, the new configuration is stored if more favorable than the previous best solution (figure 5).

3.3. Simulated annealing

This algorithm is amongst the most widely used algorithms when little knowledge of the objective function is available [13], especially when the search space is unordered. To overcome local optima, the method permits some higher

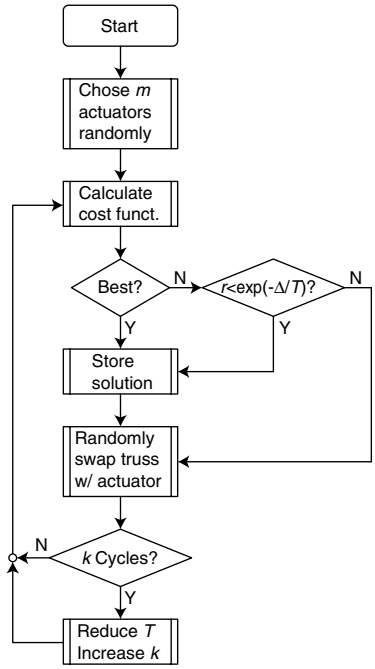

Figure 6. Flow diagram of the simulated annealing algorithm.

cost steps to proceed. After the cost function has been calculated with the local optimization algorithm, one randomly chosen actuator in the set is swapped with a randomly chosen truss member and the fitness of the new set is determined. If the new set is superior to the old one, it is accepted. Moreover, if the new set has a higher cost than the old, it is still accepted provided that

$$r \leqslant \exp(-\Delta / T), \tag{12}$$

where r is a random number ($0 \leqslant r < 1$), T is the annealing temperature and Δ ($=f_{\text{new}} - f_{\text{old}}$) is the difference in cost between the new and the old solutions. If equation (12) is satisfied, the new set is stored and the cycle restarted (figure 6).

The annealing temperature T has to be adapted during optimization. Various methods have been devised for determining the initial values and the subsequent cooling [13]. In the present study, a simple cooling schedule is used. That is, after a predetermined number of cycles (k), T is reduced to αT ($0 < \alpha < 1$). Simultaneously, the number of cycles to be completed is increased to βk ($1 < \beta$). Initially, T is chosen to accept almost all bad sets and slowly lowered ($0.8 < \alpha < 1$). The parameters are $\alpha = 0.9$, $\beta = 1.1$ and an initial cycle length $k = 100$.

3.4. The genetic algorithm

This algorithm is a technique that 'converges' to the best possible solution [14, 15]. Similar to simulated annealing, it is suitable when no prior knowledge is available regarding the optimum and especially where the search space is unordered. The genetic algorithm works with ℓ sets of m actuators simultaneously. Each set of actuators is mapped into a chromosome. Each actuator corresponds to a particular position in a chromosome. The chromosome length depends on the number of actuators in a set. A binary representation of the location number with 7 bits/actuator was chosen for

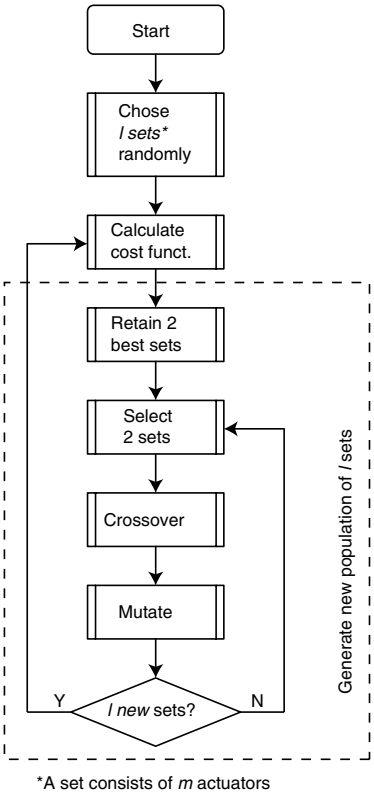

Figure 7. Flow diagram of the genetic algorithm.

this study: bits 1–7 represent the first actuator, bits 8–14 the second etc. The chromosomes are grouped into a population of ℓ chromosomes, with population size determined by the length of each chromosome [16]. In the present study a population size of 20 satisfies the minimum requirements.

The initial population is chosen randomly and the cost function, also known as the fitness, is calculated. The cost of every individual chromosome is obtained by decoding, resulting in a set of location numbers. The maximum realizable deformation is calculated for every set using the local optimization routine. Finally, the population is sorted by cost, starting with the best. A new population is generated. In order to retain the best solutions so far, the two best chromosomes are copied to the new population (known as 'elitism'). The remainder of the new population is filled by selecting chromosomes from the previous population ('selection'), exchanging information between them ('crossover') and randomly changing bits ('mutation') to introduce new information. Once the new population is filled, the fitness is calculated and the cycle restarted (figure 7).

Several variations have been defined [15, 16]. The simplest selection procedure is to choose two chromosomes randomly. This method does not rely on information about the cost of the individual chromosomes. Other methods such as 'roulette-wheel' or 'rank' selection base the probability of selecting a chromosome on its cost, giving the best chromosomes the highest chance of passing their information onto the next generation. The crossover procedure exchanges information between two selected chromosomes mimicking sexual reproduction. This is performed by cutting the parent chromosomes into several pieces at the same randomly selected locations. The number of cuts is reflected in the name (a single cut is a one-point crossover etc). A four-point crossover with a probability of 95% and rank selection provides consistent results in the present study. The new chromosomes are assembled by alternating pieces from the two parents. The last step is the mutation of the newly formed chromosomes, conducted by randomly flipping some bits with a preset probability. In the present study a probability of 5% gives the best results.

4. Scenarios and results

4.1. Scenarios

The four algorithms are used to optimize the actuator placement for four different scenarios: hinging and twisting, both using 8 and 16 actuators. Hinging refers to uniform lifting of the free edge, while twisting raises one side of the free edge while simultaneously lowering the other, while maintaining the free edge in a straight line. Note that there are 10^{11} and 10^{17} combinations for placing 8 and 16 actuators, respectively, on 96 possible locations. All optimizations are performed using six control points located along the outer edge (figure 3).

All algorithms are continued for 10 000 cycles for each scenario. Every scenario is optimized 10 times to reduce the effect of an unfavorable initial choice. The commonest actuators of the 10 runs are manually selected and passed through the local optimizer. If the cost of the manual selection is worse than the best optimization result, a new manual set (replacing some of the less common positions) is rerun. Improvements were always found using fewer than five manual runs.

4.2. Optimization results

These comprise the locations of the actuators as well as the predicted displacement of every control point and the required displacement of every actuator. The placement of actuators for hinging is relatively simple. For 8 and 16 actuators, filling all positions along row 3 and along rows 3 and 4, respectively, maximizes the achievable deformation. Twisting is more intricate (figure 8), but symmetric with respect to the central axis.

The displacement at the control point on the edge is used to illustrate the quality of the results (figure 3, open control point), assembled in table 1 and visualized in figure 9. The iterative replacement algorithm performs well in the first few cycles but is then trapped in a local optimum. Conversely, the simulated annealing and genetic algorithms perform well because they accept some inferior solutions that inhibit trapping at a local minimum. The difference in their performance is within one standard deviation (table 1) and statistically insignificant.

Note that hinging yields higher deformations, because twisting stretches the solid face, increasing the actuation resistance. Doubling the number of actuators from 8 to 16 triples the achievable hinging (from 78 to 216 mm) and doubles the twisting (from 19 to 53 mm). The results have been verified by FE analysis (figure 10) with actuators at the locations revealed by the optimization. The predicted displacements are exactly reproduced.

Table 1. Examples of the worst, average and best results, as well as the standard deviation. The best results from manual post-processing as well as from the FEM verification are included. The displacement of the top left edge has been used.

Min/avg/max (mm) Std deviation (mm)	Hinge		Twist	
	8 actuators	16 actuators	8 actuators	16 actuators
Random	1.7/2.2/2.6 0.4	22.2/31.3/53.3 9.0	1.9/2.1/2.8 0.3	12.1/15.0/20.4 2.6
Iterative replacement	0.1/0.5/1.3 0.3	3.0/14.1/38.5 12.1	0.2/0.7/2.0 0.6	2.0/6.1/18.5 4.8
Simulated annealing	3.2/6.7/20.8 5.3	104.4/140.8/176.6 21.5	5.2/13.1/19.3 5.8	32.5/38.6/42.0 3.2
Genetic algorithm	7.2/23.7/67.3 15.3	129.8/149.6/164.5 11.1	6.9/12.6/19.3 4.2	34.4/39.9/46.3 4.2
Manual post-processing	78.1	216.2	19.3	53.3
FEM verification	78.1	216.2	19.3	53.3

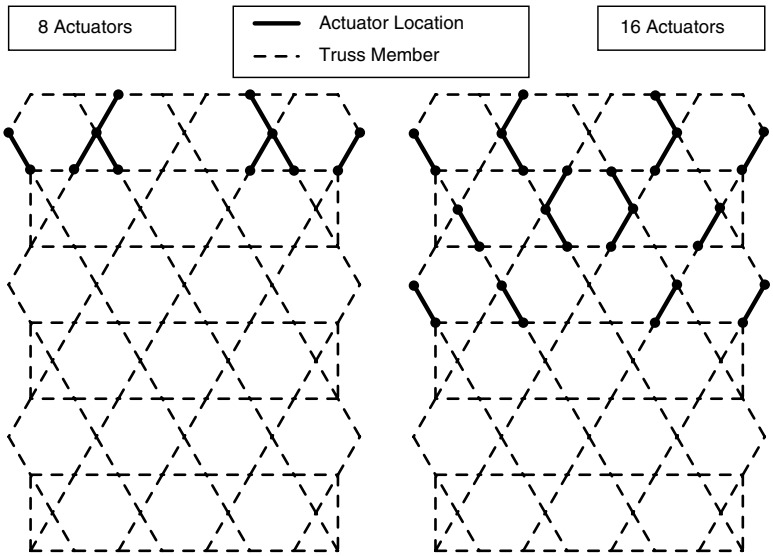

Figure 8. Optimized locations of the actuators for twisting with 8 and 16 actuators.

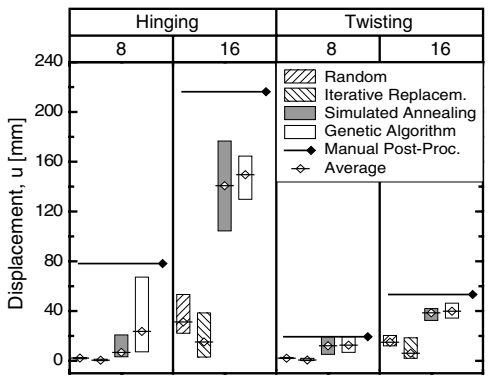

Figure 9. Optimized deformation result for all scenarios and algorithms. The span is marked by boxes with the average highlighted by an open diamond. Closed diamonds represent the result of the manual post-processing.

5. Concluding remarks

A heuristic global optimization routine combined with a fast local optimization algorithm is efficient in assessing the best placement of actuators in a Kagome morphing structure. Because of the large, unordered search space, methods that accept inferior cost moves must be used to avoid trapping at a local minimum—such as simulated annealing and a genetic algorithm. The two methods yield results of equal quality. Given their random nature, repeating the runs and manually assembling the common features increases the quality of the results.

While all optimizations have been performed for only one target shape, it is straightforward to implement an optimization for multiple deformations. In that case, the local algorithm would be run for every target deformation and the results stored in a local cost vector. The global cost would be determined from the local cost vector by means of an appropriate norm. This method would allow simultaneous optimization for more than one objective. For example, active vibration damping could be explored, with some Kagome members replaced with sensors and high frequency actuators. The resulting vibration amplitude could be calculated by a modal FE analysis and used as a second cost function.

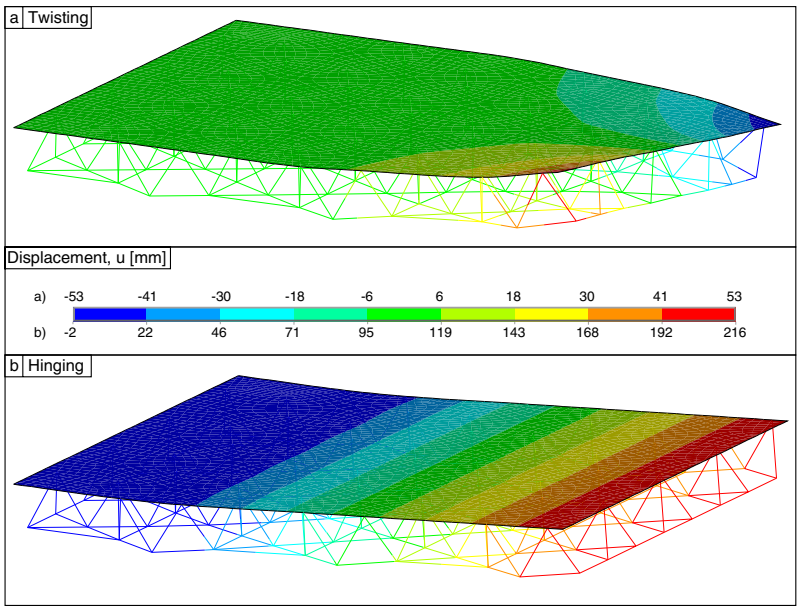

Figure 10. Result of the FEM verification of the optimization results for twisting and hinging using 16 actuators.

Acknowledgments

The authors would like to thank John Hutchinson from Harvard University and Simon Guest from Cambridge University for very valuable discussions and input to this study.

References

[1] Lu T J, Hutchinson J W and Evans A G 2001 Optimal design of a flexural actuator *J. Mech. Phys. Solids* **49** 2071–93
[2] Hutchinson R G, Wicks N, Evans A G, Fleck N A and Hutchinson J W 2003 Kagome plate structures for actuation *Int. J. Solids Struct.* **40** 6969–80
[3] Hyun S and Torquato S 2002 Optimal and manufacturable two-dimensional, Kagomé-like cellular solids *J. Mater. Res.* **17** 137–44
[4] Christensen R M 2000 Mechanics of cellular and other low-density materials *Int. J. Solids Struct.* **37** 93–104
[5] Wicks N 2003 Optimization and actuation of truss structures *PhD Thesis* Engineering Sciences, Harvard University, Cambridge, MA
[6] Dos Santos e Lucato S L, Wang J, McMeeking R M and Evans A G 2004 Design and demonstration of a high authority shape morphing structure *Int. J. Solids Struct.* **41** 3521–43
[7] Dos Santos e Lucato S L and Evans A G 2005 The load capacity of a Kagome based high authority shape morphing structure *J. Appl. Mech.* at press
[8] Wicks N and Hutchinson J W 2004 Sandwich plates actuated by a Kagome planar truss *J. Appl. Mech.* **71** 652–62
[9] Sadri A M, Wright J R and Wynne R J 1999 Modelling and optimal placement of piezoelectric actuators in isotropic plates using genetic algorithms *Smart Mater. Struct.* **8** 490–8
[10] Mir M and Hasan Imam M 1996 Analytic annealing for macrocell placement optimization *Comput. Electr. Eng.* **22** 169–77
[11] Luenberger D G 1973 *Linear and Nonlinear Programming* 2nd edn (Reading, MA: Addison-Wesley)
[12] Kreyszig E 1999 *Advanced Engineering Mathematics* 8th edn (New York: Wiley)
[13] Sait M S and Youssef H 1999 *Iterative Computer Algorithms with Applications in Engineering* (Los Alamitos, CA: IEEE Computer Society)
[14] Goldberg D 1989 *Genetic Algorithms in Search, Optimization, and Machine Learning* (Reading, MA: Addison-Wesley)
[15] Holland J H 1975 *Adaptation in Natural and Artificial Systems* (Ann Arbor, MI: University of Michigan Press)
[16] Reeves C and Rowe J 2003 *Genetic Algorithms—Principles and Perspectives* (Boston, MA: Kluwer Academic)