Proceedings of the ASME 2020
International Design Engineering Technical Conferences
and Computers and Information in Engineering Conference
IDETC/CIE2020
August 17-19, 2020, Virtual, Online

DETC2020-22393

BUILDING BLOCK BASED TOPOLOGY SYNTHESIS ALGORITHM TO OPTIMIZE THE NATURAL FREQUENCY IN LARGE STROKE FLEXURE MECHANISMS

Mathijs E. Fix, Dannis M. Brouwer

Precision Engineering
Faculty of Engineering Technology
University of Twente, Enschede, The Netherlands

Ronald G.K.M. Aarts*

Applied Mechanics & Data Analysis
Faculty of Engineering Technology
University of Twente, Enschede, The Netherlands
R.G.K.M.Aarts@utwente.nl

ABSTRACT

Flexure based compliant mechanisms suited for a large range of motion can be designed by handling the challenges arising from combining low compliance in the desired directions, high support stiffness, low stresses and high unwanted natural frequencies. Current topology optimization tools typically can't model large deflections of flexures, are too conceptual or are case specific. In this research, a new spatial topological synthesis algorithm based on building blocks is proposed to optimize the performance of an initial design. The algorithm consists of successive shape optimizations and layout syntheses. In each shape optimization the dimensions for some layout are optimized. The layout synthesis strategically replaces the most "critical" building block with a better option. To maximize the first unwanted natural frequency the replacement strategy depends the strain energy distribution of the accompanying mode shape. The algorithm is tested for the design of a 1-DOF flexure hinge. The obtained final layout agrees with results known from literature.

1 Introduction

Flexure based compliant mechanism are applied in consumer products as well as high-end position mechanisms [1–3]. The absence of backlash, wear and friction allows for high precision movements with great repeatability. One of the ongoing challenges in improving compliant mechanism is achieving high performance at a large stroke. Typical flexure hinges show a significant drop of the support stiffness when deflected which reduces the allowable payload. When a mechatronic system is build with such flexure hinge, it will exhibit lower natural frequencies, which limit the performance as the control bandwidth has to be reduced to avoid interference with these resonances.

To avoid such unwanted behavior, new and innovative flexure designs are needed to handle the challenges arising from trade-offs between compliance in the direction of the Degrees of Freedoms (DOF), support stiffnesses, stresses or unwanted natural frequencies. In case of large deflections, the relations between stress and strain become non-linear which has to be accounted for in the design method. Overviews of such methods are provided e.g. by Howell et al. [3] or Gallego and Herder [4].

In the rigid-body replacement method the joints in a rigid-body mechanism are replaced by a compliant equivalent [3]. The Pseudo-Rigid-Body model predicts the kinematic behaviour of the mechanism [5], but the detailed design of complex flexure joint may not be considered. This topic is e.g. addressed by Wiersma et al. [6] and Gunnink et al. [7] who optimize the dimensions of several parametrized 1-DOF flexure hinges to obtain the highest first unwanted natural frequencies or highest support stiffness. In such size or shape optimization methods the layout isn't modified, so other methods like the rigid-body replacement or the Freedom and Constraint Topology (FACT) method [8] are used to generate multiple layouts.

Numerous methods are known for structural topology op-

*Address all correspondence to this author.

timizations where the layout is not prescribed beforehand [9]. Many design problems can be solved in this way, but these approaches tend to become computationally expensive for more complex 3D mechanisms and special solution techniques are required to incorporate large deformations. Furthermore, the obtained solutions quite often show lumped compliances, whereas distributed compliance is generally advantageous for large stroke motion to avoid unacceptable high stress levels [3]. Meshing such long and slender flexures may pose another challenge to the finite element based methods commonly applied.

Design methods based on "building blocks" optimize the topology of a mechanism by replacing or combining these building blocks [10, 11]. These methods are computationally less expensive compared to the general structural topology optimization methods as the number of design parameters is greatly reduced and additionally the methods may benefit from prior knowledge of the building blocks. Naves et al. [12] present a building block method to maximize the first unwanted natural frequency of a large stroke ($\pm 45\,\text{deg}$) 1-DOF flexure hinge. The algorithm is initiated with a Three Flexure Cross Hinge (TFCH) and involves a number of iterations with separate shape optimizations and layout syntheses. In each shape optimization the dimensions of a specific layout are optimized. The layout synthesis requires user input to replace one of the building blocks. In just five layout iterations, the first unwanted natural frequency increases from 11 Hz to 102 Hz. Note that Naves et al. [12] propose to optimize the natural frequency as the mass and inertia of the applied load offer an implicit weighting of the various translational and rotational stiffnesses which is otherwise not trivial.

The aim of this paper is to develop a general spatial topology optimization algorithm using a building block approach similar to Naves et al. [12] but without the need of user input for the layout synthesis. The overall method is presented in Section 2 with a summary of the shape optimization in Section 3. A key step in the proposed method is the selection of the building block to be replaced in order to obtain a more optimal layout. The algorithm should be suited for different optimization goals, although in this paper we focus on the above mentioned case studied by Naves et al. [12] and detailed in Section 4. In this case the first unwanted natural frequency is maximized for which purpose a strain energy approach is presented in Sections 5 and 6 to determine the "critical" building block in a layout and to predict the most promising replacement. The algorithm involves two tuning parameters that largely affect the performance of the algorithm as can be concluded from the results in Section 7. Finally some conclusions are summarized in Section 8.

2 General optimization scheme

The optimization method developed in this paper is focussed on improving predefined spatial flexure mechanisms by the replacement of building blocks. For that purpose several building blocks will be identified in a predefined mechanism. The method aims at strategically replacing building blocks with minimal computational expense. In this section first the required steps of the algorithm are identified. Second, the resulting algorithm scheme is introduced along with the definition of applied terms and variables. Third, the algorithm steps are discussed in more detail. The case specific optimization scheme introduced by Naves et al. [12] is evaluated to identify the required steps of the new method. Their iterative sequence of shape optimizations and layout syntheses offers a straightforward base for an optimization algorithm. The following steps are identified:

1. *Shape optimization* is performed to determine the optimal dimensions for a specific parametrized layout. Dimensions are for instance the thickness, width, and height of the parametrized mechanism, or dimensions related to specific building blocks. This shape optimization is required to assure maximum performance for the given design layout. Shape optimization algorithms for flexure mechanisms exist in literature, see for instance [6] or [7].

2. *Layout synthesis* as described by Naves et al. [12] involves user insight to change building blocks. To transfer this insight to an algorithm and to allow a more general approach, the synthesis is split into:

- **(a)** *Detect critical building block* $\mathbf{BB}_c$, i.e. replacing this building block should result in the largest improvement with respect to the optimization goal. The method for detecting this building block depends on the optimization goal.
- **(b)** *Replace* $\mathbf{BB}_c$ *with a better building block* which is selected from a set of predefined building blocks for further evaluation.

In general, the best replacement building block in step 2(b) can only be determined with absolute certainty after shape optimizations of all possible options. However, the shape optimizations are computationally expensive and hence it is preferred to evaluate only the most promising replacement building blocks. This should not restrict the algorithm too much as that would introduce the risk of overlooking feasible options in the solution space. In other words, a limited set of the best option(s) for replacing the $\mathbf{BB}_c$ should be selected for the next shape optimization(s) to allow multiple off-springs from the original layout while limiting the computational effort.

$\rightarrow$ *Repeat until layout is optimal*, i.e. the algorithm should finish if all building blocks in a design layout are optimal. A building block is optimal if no better option exists for its location in the current layout.

An algorithm structure to implement these steps is shown in Figure 1. The scheme is split up in a user input part and an algorithm part. The input required from the user can be split into

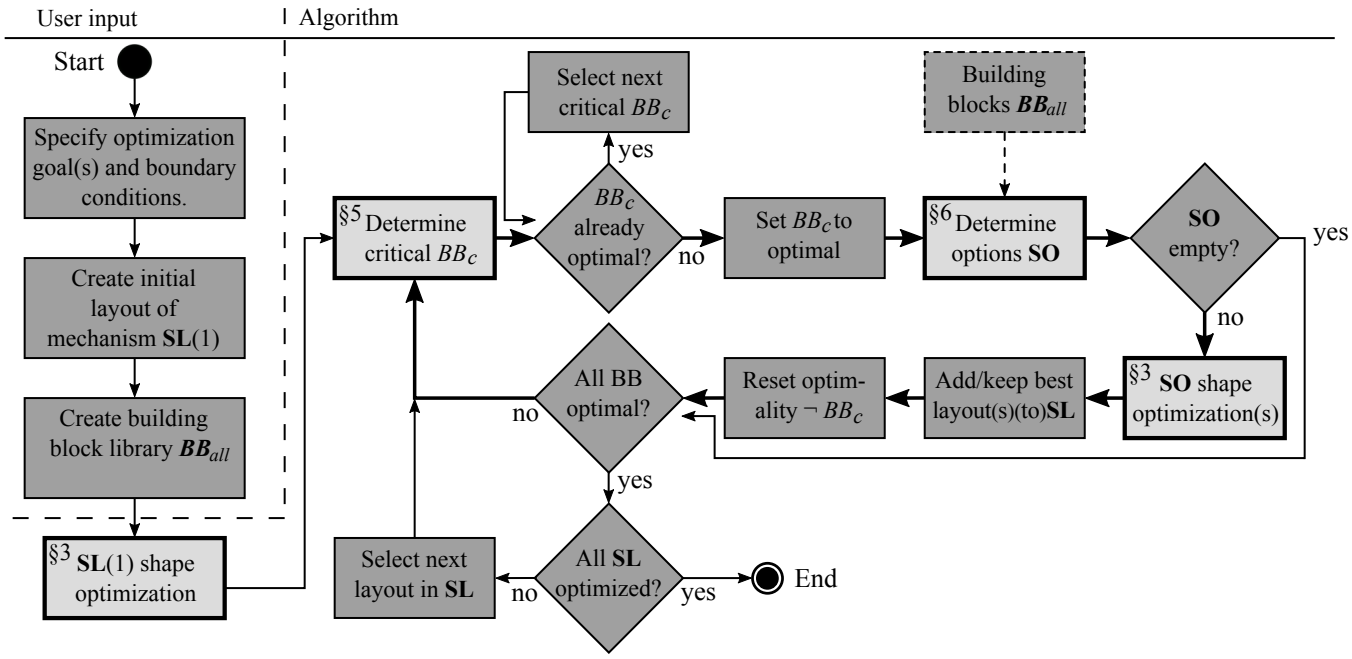


FIGURE 1. Algorithm scheme of the topology optimization algorithm with user input. Abbreviations: **SO**=*Set of options*, **SL**=*Set of layouts*, $\mathbf{BB}_c$=*Critical building block*. Symbol ¬ is the logical "not" operator.

three parts. Firstly, the optimization goal(s) and the corresponding boundary conditions should be specified. Various boundary conditions are possible, e.g. they can be related to the footprint, to specific kinematic behavior, to the minimum support stiffness, or to the maximum actuation power. Secondly, an initial layout of a mechanism is required. The designer may input a self-designed layout or select one from literature, see for instance the overviews provided by Howell [3] or Machekposhti and Herder [13]. This input should include the positions of the building blocks. Thirdly, the designer has to create the library of building blocks $\mathbf{BB}_{all}$ that will be considered in the optimization.

The algorithm is discussed in detail at the end of this section. The main loop is depicted with thicker arrows and the important steps for the algorithm are depicted in the light gray boxes. These steps are the **SO** *shape optimization(s)*, *Determine critical* $\mathbf{BB}_c$and *Determine options* **SO**, which are discussed in depth in sections 3, 5 and 6 respectively. The implementations of the latter two steps depend on the optimization goal.

Definitions of terms and variables

- A *Building block* **BB** is a predefined flexible part of mechanism with a certain set of intended DOFs. It is defined between a start and end position. A simple example of a building block is a leaf spring, which introduces three DOFs. A building block should match the kinematic requirements for its location in the mechanism.
- The *Building block library* $\mathbf{BB}_{all}$ stores the complete set of predefined building blocks.
- The *Design layout* defines the topology of a mechanism. It includes an overview of the locations at which a building block is present. The building blocks are the only flexible parts of a layout.
- The *Set of layouts* **SL** stores all the layouts that could potentially be optimized to the optimal layout. This set is expanded if an off-spring of the original layout is created. It keeps track of the optimality of the building blocks.
- An *Optimal building block* refers to a building block at a certain location in a layout for which no better building block exists at that location as long as the other building blocks in the layout are not replaced. If a layout is changed, the optimality of all building blocks should be reset.
- The *Critical building block* $\mathbf{BB}_c$ is the building block of which replacement potentially results in the largest improvement of the optimization goal. Therefore, it is the first building block that should be replaced by the algorithm.
- The *Set of options* **SO** specifies the replacement options for a critical building block $\mathbf{BB}_c$ that show promising potential. Each of these options is combined with the other building blocks of the layout currently being optimized to create new layouts.

Algorithm

The main steps of the algorithm, Figure 1, will now be discussed. This is followed by an example of one iteration. The main loop of the algorithm optimizes one layout from **SL** at a time.

- **SL**(1) *shape optimization* (see Section 3) is the start of the algorithm with a shape optimization of the parametrized initial design layout **SL**(1).
- *Determine critical* $\mathbf{BB}_c$ (see Section 5) follows a shape optimization to determine the critical building block with respect to the optimization goal.
- $\mathbf{BB}_c$ *already optimal?* checks whether the critical building block is already optimal in this location In that case it should not be replaced, but it should be identified which next critical element is not optimal yet.
- *Set* $\mathbf{BB}_c$ *to optimal* as the critical building block will be replaced by a building block that is expected to be more optimal at the location of $\mathbf{BB}_c$. If no better building block is found for this location, the current $\mathbf{BB}_c$ is the optimal one.
- *Determine options* **SO** (see Section 6) should select the most promising replacement options for $\mathbf{BB}_c$ from the building block library $\mathbf{BB}_{all}$.
- **SO** *empty?* is true if no promising options are found. The consequence is that $\mathbf{BB}_c$ is set to optimal in the previous step and the next steps of replacing $\mathbf{BB}_c$ can be skipped. If **SO** is not empty, the option(s) should be evaluated by means of the shape optimization(s).
- **SO** *shape optimization(s)* (see Section 3) involves the optimization of the parametrized dimensions for all entries of **SO**.
- *Add best layouts to* **SL** means that from **SO** the best layouts are selected and added to the *Set of layouts* **SL**. This selection depends on a so-called range parameter r_L. As will be detailed in Section 3 the optimization of a layout involves the minimization of an objective function F. The performance of layout i in **SL** is then expressed as $F(\mathbf{SL}(i))$. Suppose $\mathbf{SO}(j)$ is the best replacement option for $\mathbf{BB}_c$ in layout $\mathbf{SL}(i)$. Three different cases are identified:
  1. $F(\mathbf{SO}(j)) \cdot (1+r_L) < F(\mathbf{SL}(i))$: The entry of $\mathbf{SL}(i)$ is removed from **SL** and all options with lower cost than $F(\mathbf{SO}(j)) \cdot (1+r_L)$ are added to **SL**.
  2. $F(\mathbf{SL}(i)) \cdot (1+r_L) < F(\mathbf{SO}(j))$: Entry $\mathbf{SL}(i)$ is preserved and no layouts from **SO** are added.
  3. All other cases: Entry $\mathbf{SL}(i)$ is preserved. The layouts from **SO** that have a lower objective function value than $\min(F(\mathbf{SL}(i)), F(\mathbf{SO}(i))) \cdot (1+r_L)$ are added to **SL**.
- *Reset optimality* $\neg$ *("not")* $\mathbf{BB}_c$ means that in the case a layout is changed, the optimality of the applied building blocks should be re-evaluated. Only the optimality of replaced building block $\mathbf{BB}_c$ is known.
- *All* **BB** *optimal?* checks whether all building blocks are optimal in the current layout. If this is not the case, the current layout should be further optimized. Otherwise, the next layout in **SL** should be selected. If all layouts in **SL** are fully optimized, the algorithm ends.

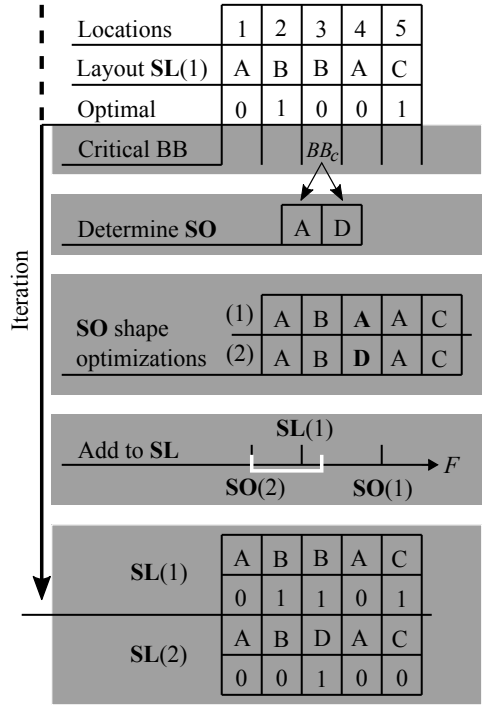


FIGURE 2. Example steps of algorithm for mechanism with 5 building blocks. The building blocks to choose from $\mathbf{BB}_{all}$ are labelled A, B, C and D. At the start of this iteration **SL** contains one layout, which has 2 optimal building blocks. In the step *Add to* ***SL*** criterion 3 is applied where **SL**(1) is preserved and only **SO**(2) is added from the options.

The steps of one example iteration are shown in Figure 2, which demonstrates the changes to the sets **SO** and **SL**.

3 Shape optimization

The shape optimization involves a suitable optimization algorithm for parametrized mechanisms. Hence the "shape" of these mechanisms is described by the parameter vector $\mathbf{p}$. The algorithm should be able to handle various optimization goals as well as non-linear constraints. In line with earlier research [6, 7] the Nelder–Mead simplex algorithm [14] is used. This algorithm offers a robust optimization procedure without the use of derivatives to determine the optimum parameter set $\mathbf{p}_o$ that minimizes some objective function $F(\mathbf{p})$, i.e.

$$\mathbf{p}_o = \arg\min_{\mathbf{p}} F(\mathbf{p}). \tag{1}$$

However, the Nelder–Mead algorithm is an unconstrained method, which implies constraints must be enforced to it. Naves et al. [12] realize this with a penalty constraint function $P(\mathbf{p})$ that causes a steep increase of the objective function value when constraints are violated by defining

$$F(\mathbf{p}) = P(\mathbf{p})^5 \, g(\mathbf{p}), \tag{2}$$

where the optimization goal is to minimize $g(\mathbf{p})$. The penalty function $P(\mathbf{p})$ is defined such that it increases from 1 when one or more constraints are violated and the power 5 assures fast convergence towards the feasible region. Constraints can be imposed on parameter lower and upper bounds. Alternatively, constraints can also use results from the analysis of the system, e.g. the requirement that the maximum Von Mises stress occurring anywhere in the mechanisms for any configuration in the operation range should remain below some limit σ_m.

The Nelder–Mead method concludes a minimization if the difference between function values and parameter values in consecutive iterations reaches below certain thresholds, T_F and T_p respectively. In the considered case both thresholds are set to 0.5%. Furthermore, the minimization is aborted if some maximum number of iterations or function evaluation is reached. Both are set to $200n$, where n is the number of design parameters. Finally the algorithm is terminated if a negative natural frequency is encountered as this implies a negative stiffness which means buckling occurs.

The Nelder–Mead method starts from an initial parameter set. If the objective function contains local optima, the initial parameter quite likely determines the resulting optimum. Therefore, multiple minimizations, starting from different randomly generated parameters sets between the parameter bounds, are required to find the global optimum. The influence of the number of runs n_r will be investigated in this study.

4 Case study

Although the presented algorithm is intended to be rather generic, the algorithm steps *Determine critical* $\mathbf{BB}_c$ and *Determine options* **SO** depend on the specific optimization goal which will be outlined in this section. In order to facilitate a comparison with literature, the same hinge as investigated by Naves et al. [12] is used. This TFCH hinge is displayed in Figure 3. The goal is to maximize the first unwanted natural frequency f_u over the full rotation range of the hinge, i.e. angle $\theta \in [-45\,\text{deg}, 45\,\text{deg}]$. A load is connected to the hinge. The inertia and mass terms, I and m, are presented in Table 1. The mass is defined as a point mass in the center of rotation. The inertia is also defined around this point. Furthermore, the table includes the maximum actuation moment M_m and the material properties of the steel used for the building blocks, i.e. Young's modulus E, the shear modulus G and the maximum allowable Von Mises stress σ_m.

Building blocks

The building blocks for the example case are defined as a commonly used Leaf Spring (LS), a Torsionally Reinforced Leafspring (TRLS), and a Double Three Flexure Cross Hinge (DTFCH), see Figure 4. By applying the FACT method, it can be shown that these building blocks only introduce constraints that intersect the axis of rotation or are parallel to this axis [12]. The

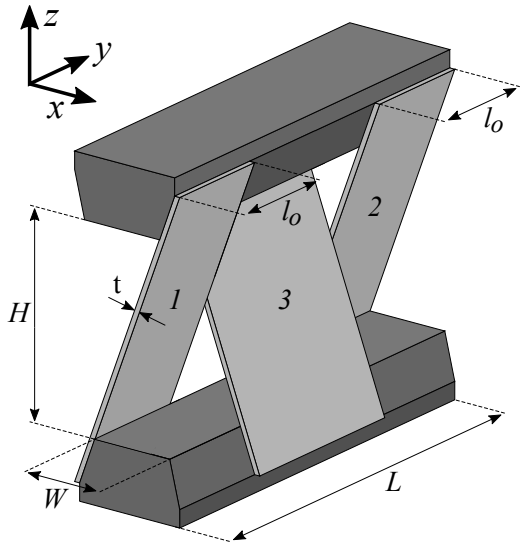


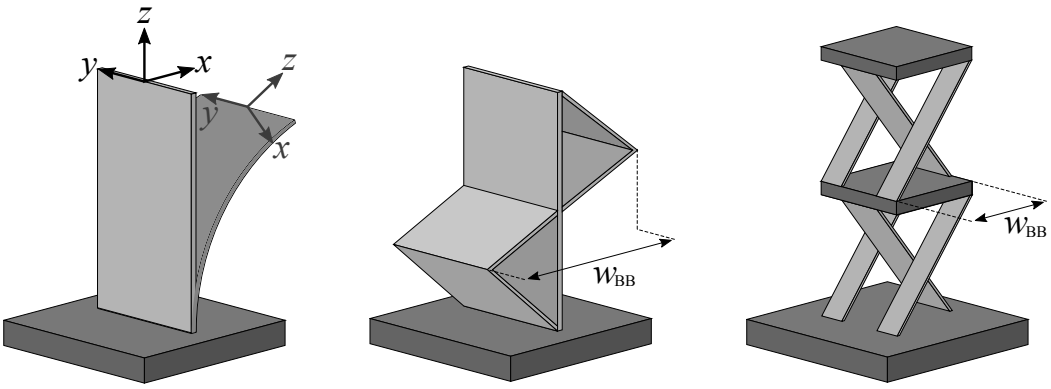
FIGURE 3. Three Flexure Cross Hinge (TFCH) with dimensions and building block numbers.

TABLE 1. Parameters used in the case study.

Parameter	Value	Unit
I_{xx}	0.02534	kg m^2
I_{xy}	0.01465	kg m^2
I_{xz}	0.00008	kg m^2
I_{yy}	0.01371	kg m^2
I_{yz}	-0.00034	kg m^2
I_{zz}	0.03826	kg m^2
m	0.5740	kg
σ_m	600	MPa
E	210	GPa
G	80	GPa
M_m	1.5	Nm

DTFCH requires two TFCH stacked in series to meet this requirement. A single TFCH introduces degrees of constraint spanning multiple planes due to the directional difference between inner and outer leaf springs. The building blocks all inhibit different stiffness characteristics. The LS is compliant in y rotation, but has quickly deteriorating support stiffnesses in deformed states. The TRLS has slightly lower compliance in y rotation, but offers more support in rotation around the z-axis. The DTFCH has a much lower compliance in y rotation, but does offer more support in y direction in deformed state.

To further expand the number building block options for the topology of the mechanism, the building blocks can be stacked. For instance, two TRLS can be connected in series to form a new building block. The stacks are limited to two building blocks



(a) Leaf Spring (LS). (b) Torsionally Reinforced Leafspring (TRLS). (c) Double Three Flexure Cross Hinge (DTFCH).

FIGURE 4. The three building blocks.

TABLE 2. Library of possible building block and series stacks, $\mathbf{BB}_{all}$, for the considered case study.

#	Type	Stack
1	LS	1
2	TRLS	1
3	TRLS	2
4	DTFCH	1
5	DTFCH	2

in this case study. The resulting building block library $\mathbf{BB}_{all}$ is presented in Table 2.

Symmetry

Engineering insight can be used to reduce the computational expense of the algorithm. For the case study of the TFCH it is expected that a symmetric layout with identical building blocks at both outer building block locations 1 and 2 is optimal due to the overall layout of the mechanism. Applying such symmetry reduces the number of building block locations and consequently the computational expense decreases. Tests will be performed with and without imposing symmetry.

Parametrization of TFCH and building blocks

For the shape optimization, the model of the TFCH has to be parametrized. The parameters and their dimensions are shown in Figure 3. The building blocks TRLS and DTFCH both introduce an extra parameter w_{BB} as illustrated in Figure 4. The thickness t is equal for all building blocks. The length L is set to 85 mm. The width of the outer building blocks l_o is equal for both sides. An extra parameter is added to take the orientation of the load case into account in the optimization. This parameter α describes the rotation around the y-axis through the center of rotation. The parameter bounds are given in Table 3.

TABLE 3. Parameters with their corresponding lower and upper bounds and units.

Parameter	Min	Max	Unit
H	0.005	0.1	m
W	0.005	0.1	m
t	0.2	1	mm
lo	0.001	0.04	m
α	$-\pi$	π	rad
w_{BB} (TRLS)	0.005	0.05	m
w_{BB} (DTFCH)	0.005	0.05	m

Modelling tools and objective function

For the case study, an objective function based on the maximization of the first unwanted natural frequency is required for the shape optimization. The non-linear finite element based program SPACAR for flexible multibody modelling [15] is used to determine the natural frequencies over the full rotation range of the hinge, i.e. $\theta \in [-45\,\text{deg}, 45\,\text{deg}]$. Flexures are modelled using multiple non-linear spatial flexible beam elements. Due to the independent non-linear formulation of the deformation modes, the number of finite elements can be greatly reduced, allowing for cheap calculations. Geometrical non-linearities are included in the beam stiffness formulation [16]. Constrained warping of flexures is accounted for by increasing the torsional stiffness of short leaf springs [6]. To compute the natural frequencies, the stationary solution of the non-linear equation of motion is first obtained for some deformed configuration, i.e. some angle θ in the specified range. Next, the equations of motion are (symbolically) linearized in this configuration from which the mass and stiffness matrices, $\mathbf{M}$ and $\mathbf{K}$, result. These are so-called reduced matrices that consider the motion expressed by a set of (independent) degrees of freedom. From the eigenvalue problem

$$\det(\mathbf{K} - \omega^2\mathbf{M}) = 0, \tag{3}$$

the natural frequencies ω (in rad/s) follow from the eigenvalues and the accompanying eigenvectors represent the mode shapes expressed in the degrees of freedom. The resulting minimum unwanted natural frequency at angle θ (in Hz) is expressed by $f_u(\mathbf{p},\theta)$ and the function $g(\mathbf{p})$ in Eq. (2) is then the minimum of $f_u(\mathbf{p},\theta)^{-1}$ for all allowable angles θ.

5 Critical building block

In order to improve the layout of a mechanism without checking every possible building block location, a "smart" selection is required that identifies the "critical" building block which, upon replacement, most likely introduces the largest improvement with respect to the optimization goal(s). Such a method is generally goal dependent. For instance, the maximization of support stiffness would focus on the building block with the lowest support stiffness in a certain direction.

For the case study, a method is required to identify which building block is the critical element for the first unwanted natural frequency f_u. It is postulated that the strain energy associated with the mode shape of the first unwanted natural frequency is well-suited for this purpose. In the modelling framework of SPACAR, this energy can be computed relatively easy once the first unwanted natural frequencies has been computed using Eq. (3). The corresponding mode shape $\mathbf{v}_u$ is then also known in terms of the (independent) degrees of freedom. The actual deformations ε_v of each flexible beam element in the SPACAR model due this mode linearly depend on the mode shape as

$$\varepsilon_v = \mathscr{E}_{,q}\mathbf{v}, \tag{4}$$

where Jacobian $\mathscr{E}_{,q}$ is the so-called first order geometric transfer function defined in the SPACAR modelling approach [15]. The loading of the beam element is expressed by so-called generalized stress resultants σ_v that depend linearly on the element's deformation as

$$\sigma_v = \mathbf{S}\varepsilon_v, \tag{5}$$

where $\mathbf{S}$ is the element's stiffness matrix which is known for a beam element that describes (a part of) a leaf spring [16]. In this way the strain energy U per beam element can be calculated with

$$U = \frac{1}{2}\sigma_v^T\varepsilon_v. \tag{6}$$

Adding the contributions from all beam elements in each building block enables then the selection of the critical building block of which the strain energy is largest.

6 Determine Set of Options

When the critical building block is determined, a better replacement building block has to be found. As mentioned in Section 2, checking all possible building blocks for $\mathbf{BB}_c$ by means of shape optimizations is computationally expensive. A pre-selection of potentially interesting building blocks should be made from the possible building blocks in the library $\mathbf{BB}_{all}$.

TABLE 4. Typical stiffness values for the TRLS and DTFCH for varying w_{BB}. The average value of w_{BB} between bounds is set as 100%, where the bounds are taken from Table 3. The other parameters are $H = 0.05$ m, $t = 0.3$ mm, $l_o = 0.02$ m. Angles α, β, γ represent rotations around x, y and z axis respectively.

	TRLS		DTFCH	
	$w_{BB,min}$	$w_{BB,max}$	$w_{BB,min}$	$w_{BB,max}$
S_{xx}	-20%	+26%	+46%	-34%
S_{yy}	0%	+3%	+184%	-69%
S_{zz}	0%	+1%	+210%	-71%
$S_{\alpha\alpha}$	0%	+3%	+198%	-70%
$S_{\beta\beta}$	-21%	+28%	+46%	-34%
$S_{\gamma\gamma}$	+67%	-95%	-90%	-3%

To evaluate possible replacements the first unwanted natural frequencies f_u are compared that are obtained when the critical building block $\mathbf{BB}_c$ in the current layout $\mathbf{SL}(i)$ is replaced by every other building block from the library $\mathbf{BB}_{all}$. The resulting natural frequencies can be simulated for some parameter set $\mathbf{p}$ and the best replacements can be selected for the Set of options $\mathbf{SO}$. However, the optimal parameters for the evaluated layouts are not known yet as that would involve the computational expensive shape optimization, which we are actually trying to avoid. This chicken and egg problem is solved by using "smart guesses" for the parameters and introducing a selection range r_O.

The smart guess basically uses the parameters from the most recent shape optimization of layout $\mathbf{SL}(i)$. This may result in some unknown parameters as the replacement building blocks can have other or additional parameters compared $\mathbf{BB}_c$ they replace, e.g. w_{BB} for the TRLS. This implies that the optimal parameter set of $\mathbf{SL}(i)$ may not be not sufficient to simulate the design for some of the building blocks in $\mathbf{BB}_{all}$ and some procedure is needed to specify the unknown parameters.

The possibly unknown for the case study are w_{BB} for both the TRLS and DTFCH. In an attempt to guess the optimal value for this parameter, the six main stiffnesses for these building blocks are evaluated at three different parameter settings. Table 4 shows the increase or decrease of the stiffnesses by varying the parameter w_{BB}. The stiffness of rotation around y-axis should be low, as this is the required DOF. The other support stiffnesses should be high. It appears that the support stiffness $S_{\gamma\gamma}$ of the TRLS shows a 67% increase and 95% decrease between maximum and minimum values of w_{BB}. The support stiffness in x-direction displays opposite behaviour. The DTFCH shows a large increase of almost all support stiffnesses for the minimum value of w_{BB}, but also shows a large decrease of the support stiffness around the z-axis. Overall, no clear optimal value for w_{BB} is

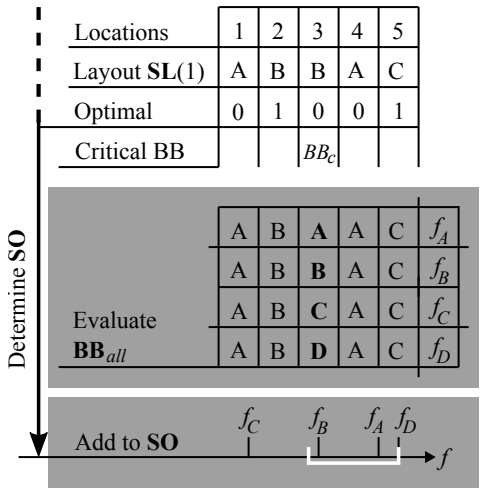


FIGURE 5. Steps for *Determine options **SO*** implemented on an example case. The example case matches with Figure 2. $\mathbf{BB}_{all}$ consist of building blocks A, B, C, and D. f_A is the first unwanted natural frequency of the layout with building block A. In this case f_D is f_u^{max}.

found. Hence to evaluate the replacement of $\mathbf{BB}_c$ with a TRLS, the first unwanted natural frequency f_u is simulated for three different parameter values, i.e. the minimum, average, and maximum value of w_{BB}.

The best results from all possible replacements are then compared as follows. Suppose the first unwanted natural frequency of entry j of $\mathbf{BB}_{all}$ is defined as f_u^j. The maximum frequency found for the complete set of $\mathbf{BB}_{all}$ is f_u^{max}. The options that are added to the Set of Options **SO** must have f_u^j in the range expressed by

$$f_u^{max} \cdot (1 - r_O) \leq f_u^j \leq f_u^{max}. \tag{7}$$

If the building block $\mathbf{BB}_c$ of layout $\mathbf{SL}(i)$ has the highest frequency f_u^{max} without any options from $\mathbf{BB}_{all}$ in the expressed range, **SO** will be left empty. In all other cases, a non-empty Set of Options will be found. In Figure 5, an example of the steps for Determine options **SO** is shown. The example case is the same as in Figure 4.

7 Results and discussion

A number of tests have been performed to evaluate to which extent the outcome of the algorithm depends on settings of parameters that control its behavior. Table 5 gives an overview of the setting that are considered. The range selection parameters r_L and r_O as introduced in Sections 2 and 6 are expected to play an important role. In addition, as described in Section 4 it is considered to impose symmetry. The total number of possible layouts with and without symmetry equals $5^2 = 25$ and $5^3 = 125$, respectively. The symmetry only applies to the layout and not to the building block specific parameter w_{BB}. In Section 3 it was pointed out that the likeness to get close to the global optimum depends on the number of runs n_r of the shape optimization from randomly generated parameter sets. A default value of 12 is assumed. It is tested whether 4 runs would suffice and convergence is checked with as many as 24 runs.

TABLE 5. Parameter settings used for the tests.

Variables	Values		
r_L	5%	10%	25%
r_O	5%	10%	25%
Symmetry	Yes	No	
n_r	4	12	24
n_b	1	2	

Most tests are performed by modelling the building blocks with sufficient beams to ensure accuracy. For modelling the LS, four beams are used. The six parts of the TRLS and the leafsprings of the DTFCH are modelled with two beams. This model ensures sufficiently accurate results while still allowing for reasonable computation times. In the sequel, the expression $n_b = 2$ refers this model. Several test with half the number of beams are performed as well, since the resulting computational expense is significantly lower. The expression $n_b = 1$ refers to these fast and less accurate tests. All tests are performed with the same sets of random starting parameters for the shape optimizations.

The results of the optimization tests are presented in two tables. Table 6 shows the results with emphasis on the differences caused by varying the number of runs n_r for the shape optimizations and symmetry conditions. Table 7 provides results for varying r_L and r_O. The building block sequences in the tables are presented in agreement with the numbers in Figure 3. For example, DDT corresponds to a DTFCH at both outer positions and a TRLS in the middle position. The shape optimization of the initial layout (Figure 3) modelled with $n_b = 2$ and optimized with $n_r = 12$ yields a first unwanted natural frequency of 8 Hz.

Number of shape optimizations n_r
Using only 4 runs of shape optimizations generally decreases the performance of the algorithm, as some optimal parameter sets are not found. However, from Table 6 it appears that $n_r = 4$ already suffices to find the optimal layout compared to $n_r = 12$. Only in the last example these results differ. The results with $n_r = 24$ show larger differences, but the final layout differs only in one case. Apparently, finding the optimal layout does not require many runs for the shape optimization. Nevertheless, the optimal dimensions for a layout do benefit from more shape optimization runs.

Number of beams n_b
In Table 6 a clear distinction is visible between the results found from models with different amounts of beams n_b. The model

TABLE 6. Results for varying numbers of runs for the shape optimizations n_r and symmetry conditions. The first unwanted natural frequency f_u is expressed in Hz, r_L and r_O in %. The building block sequence agrees with the order in Figure 3. The right columns indicate the number of layout iteration (#it) and shape optimizations (#sh).

n_b	r_L	r_O	Sym	n_r	Type	Stack	$F(\mathbf{p}_o)$	f_u	#it	#sh
1	5%	5%	y	4	DDT	1,1,1	0.011	90	3	4
		5%	y	12	DDT	1,1,1	0.011	90	3	4
		5%	y	24	DDT	1,1,2	0.009	109	4	5
	5%	10%	y	4	DDT	1,1,2	0.010	103	4	5
		10%	y	12	DDT	1,1,2	0.010	103	4	5
		10%	y	24	DDT	1,1,2	0.009	109	4	5
	5%	10%	n	4	DDT	1,1,2	0.010	103	7	9
		10%	n	12	DDT	1,1,2	0.010	103	7	9
		10%	n	24	DDT	1,1,2	0.009	109	7	9
2	5%	10%	y	4	DDT	1,1,2	0.023	43	4	6
		10%	y	12	DDT	1,1,2	0.022	45	5	8
		10%	y	24	DDT	1,1,2	0.021	48	4	8

TABLE 7. Results for varying range selection parameters r_L and r_O. Fixed settings: Models with $n_b = 2$, number of runs $n_r = 12$, and applying symmetry. Tests with r_L = 25% are not included in the table as practically all options were evaluated which is very computationally inefficient.

#	r_L	r_O	Type	Stack	$F(\mathbf{p}_o)$	f_u	#it	#sh
1	5%	5%	DDT	1,1,2	0.022	45	4	7
2		10%	DDT	1,1,2	0.022	45	5	8
3		25%	DDT	1,1,2	0.022	45	5	11
4	10%	5%	DDT	1,1,2	0.022	45	4	7
5		10%	DDT	1,1,2	0.022	45	5	8
6		25%	DDT	1,1,2	0.022	45	7	12

with $n_b = 1$ is not able to describe a torsional movement that is found with $n_b = 2$, which wrongfully causes a higher stiffness and unwanted first natural frequency f_u. Hence the results with $n_b = 1$ are only used for the evaluation of the number of runs for the shape optimization. For the evaluation of the overall performance and the parameter settings, the more accurate model $n_b = 1$ is used.

Range selection parameters r_O and r_L

The results for different settings of the parameters r_O and r_L are shown in Table 7. The tests with symmetry applied all yield the same optimum layout. Smaller values of the parameters r_O and r_L reduce the amount of shape optimizations (#sh), thus reducing computational expense. The range parameter for the selection of options r_O appears to have more influence on the required amount of shape optimizations than r_L. Varying r_L from 5% to 10% does not affect the amount of shape optimizations if r_O is set to 5% or 10%.

TABLE 8. Optimal parameter set for overall best design, [DDT] and stack [1,1,2]. All values in mm, except α in deg.

H	W	t	l_0	α	$w_{BB,1}$	$w_{BB,2}$	$w_{BB,3}$
49.3	36.0	0.34	6.4	-102	10.4	12.3	30.7

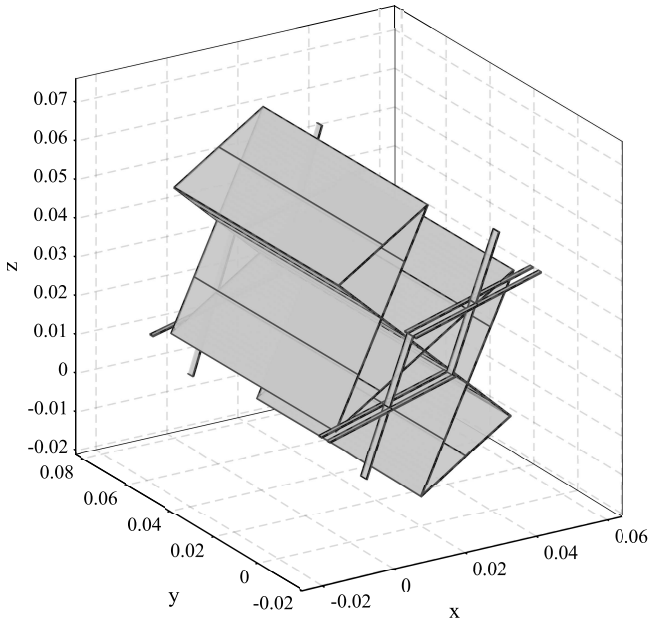


FIGURE 6. Optimum design DDT with stack 1,1,2. Corresponding parameters are shown in Table 8. Rigid parts and loadcase are not shown. All start nodes of building blocks are in the $z = 0$ plane.

Overall performance

The outcome of this research concerning the optimal layout DDT and stack 1,1,2 is similar to the findings of Naves et al. [12]. The layout is illustrated in Figure 6 with the optimal parameter set in Table 8. The maximum stress and the maximum actuation moment equal the constraints of 600 MPa and 1.50 Nm in this optimal design. The total number of evaluated layouts with symmetry applied is at least seven for the accurate model with $n_b = 2$, where 25 different layouts are possible. The case specific method of Naves et al. required only three layout evaluations to obtain this layout. They apply a parametrization with more parameters, including multiple thicknesses and different heights for building blocks. The optimum result found in this way has a first unwanted natural frequency of 81 Hz for the optimal layout. This demonstrates that a more detailed parametrization can yield better results. Applying their parametrization to the algorithm introduced in this study should yield comparable results.

8 Conclusion

In this paper a novel topology optimization algorithm based on building blocks is introduced. The algorithm combines successive shape optimizations with a layout syntheses that improve the layout of a starting configuration in an efficient way. For that purpose the design of a flexure based mechanisms is split into building blocks. In each iteration it is determined which building block deteriorates the optimization goal the most and subsequently this building block is replaced by the best option available from a library with predefined building blocks. The optimization allows for non-linear constraints, the inclusion of load cases, and large deformations of the mechanism.

The algorithm has been implemented for a case study of a 1-DOF rotation joint of which the lowest unwanted natural frequency is maximized. The performance of the algorithm depends on two parameters r_L and r_O which define ranges of natural frequencies that control the replacement of the building blocks and hence the layouts considered for optimization. Setting these parameters to small values limits the number of evaluations, but may result in overlooking promising layouts. With overly large values computational effort is wasted on unlikely topologies. It appears that the algorithm is rather robust for the parameter settings. The optimal layout is found most efficiently setting both parameters to 5% and imposing a symmetric layout. Only seven shape optimizations are required to increase the unwanted natural frequency from 8 Hz of the initial TFCH to 48 Hz. Compared with a case specific method from literature, the amount of shape optimization and thus the computational expense is approximately doubled. The new method explores a larger solution space which is still small in comparison with the total possible number of layouts that would be evaluated when using a standard brute-force approach.

ACKNOWLEDGMENT

This research was partly funded by the Innovative Research Incentives Scheme VIDI (Stichting voor de Technische Wetenschappen) (14152 NWO TTW) of the Ministry of Education, Culture and Science of the Netherlands.

REFERENCES

[1] Smith, S. T., 2000. *Flexures: Elements of elastic mechanisms*. CRC Press, Boca Raton, Florida.

[2] Cosandier, F., Henein, S., Richard, M., and Rubbert, L., 2017. *The Art of Flexure Mechanism Design*. EPFL Press, Lausanne, Switzerland.

[3] Howell, L. L., Magleby, S. P., and Olsen, B. M., 2013. *Handbook of compliant mechanisms*. John Wiley & Sons Ltd, Chichester, United Kingdom.

[4] Gallego, J. A., and Herder, J., 2009. "Synthesis methods in compliant mechanisms: An overview". In International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. Volume 7: 33rd Mechanisms and Robotics Conference, Parts A and B, pp. 193–214.

[5] Howell, L. L., 2001. *Compliant Mechanisms*. John Wiley & Sons Inc, New York, United States.

[6] Wiersma, D., Boer, S., Aarts, R., and Brouwer, D., 2014. "Design and performance optimization of large stroke spatial flexures". *Journal of Computational and Nonlinear Dynamics,* ***9***(1), p. 011016.

[7] Gunnink, K., Aarts, R., and Brouwer, D., 2013. "Performance optimization of large stroke flexure hinges for high stiffness and eigenfrequency". In Proceedings of the 28th Annual Meeting of the American Society for Precision Engineering (ASPE), pp. 1–6.

[8] Hopkins, J., 2007. "Design of parallel flexure systems via Freedom and Constraint Topologies (FACT)". Master's thesis, Massachusetts Institute of Technology.

[9] Sigmund, O., and Maute, K., 2013. "Topology optimization approache: A comparative review". *Structural and Multidisciplinary optimization,* ***48***, pp. 1031–1055.

[10] Bernardoni, P., Bidaud, P., Bidard, C., and Gosselin, F., 2004. "A new compliant mechanism design methodology based on flexible building blocks". In Smart Structures and Materials 2004: Modeling, Signal Processing, and Control, R. C. Smith, ed., Vol. 5383, International Society for Optics and Photonics, SPIE, pp. 244–254.

[11] Krishnan, G., Kim, C., and Kota, S., 2010. "An intrinsic geometric framework for the building block synthesis of single point compliant mechanisms". *Journal of Mechanisms and Robotics,* ***3***(1), 11, p. 011001.

[12] Naves, M., Brouwer, D., and Aarts, R., 2017. "Building block-based spatial topology synthesis method for large-stroke flexure hinges". *Journal of Mechanisms and Robotics,* ***9***(4), p. 041006.

[13] Machekposhti, D. F., Tolou, N., and Herder, J., 2015. "A review on compliant joints and rigid-body constant velocity universal joints toward the design of compliant homokinetic couplings". *Journal of Mechanical Design,* ***137***(3), 03, p. 032301.

[14] Nelder, J., and Mead, R., 1965. "A simplex method for function minimization". *The Computer Journal,* ***7***(4), pp. 308–313.

[15] Jonker, J., and Meijaard, J., 1990. "SPACAR — Computer program for dynamic analysis of flexible spatial mechanisms and manipulators". In *Multibody Systems Handbook*, W. Schiehlen, ed. Springer, Berlin, Heidelberg, pp. 123–143.

[16] Jonker, J., and Meijaard, J., 2013. "A geometrically nonlinear formulation of a three-dimensional beam element for solving large deflection multibody system problems". *International Journal of Non-Linear Mechanics,* ***53***, pp. 63–74.