International Journal for Numerical Methods in Engineering

WILEY

RESEARCH ARTICLE OPEN ACCESS

Compliant Mechanism Synthesis Using Nonlinear Elastic Topology Optimization With Variable Boundary Conditions

Lee R. Alacoque[1] | Anurag Bhattacharyya[2] | Kai A. James[3]

[1]Department of Aerospace Engineering, University of Illinois Urbana-Champaign, Urbana, Illinois, USA | [2]Intelligent Systems Laboratory, Palo Alto Research Center, Palo Alto, California, USA | [3]School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA

Correspondence: Lee R. Alacoque (leea2@illinois.edu; leealacoque@gmail.com)

Received: 15 January 2024 | **Revised:** 13 September 2024 | **Accepted:** 15 October 2024

Funding: This work was supported by the National Science Foundation through Grant No. 1752045.

Keywords: bistability | large displacements | morphing wing design | path generation | variable loads | variable supports

ABSTRACT

In topology optimization of compliant mechanisms, the specific placement of boundary conditions strongly affects the resulting material distribution and performance of the design. At the same time, the most effective locations of the loads and supports are often difficult to find manually. This substantially limits topology optimization's effectiveness for many mechanism design problems. We remove this limitation by developing a method which automatically determines optimal positioning of a prescribed input displacement and a set of supports simultaneously with an optimal material layout. Using nonlinear elastic physics, we synthesize a variety of compliant mechanisms with large output displacements, snap-through responses, and prescribed output paths, producing designs with significantly improved performance in every case tested. Compared to optimal designs generated using manually designed boundary conditions used in previous studies, the mechanisms presented in this paper see performance increases ranging from 47% to 380%. The results show that nonlinear mechanism responses may be particularly sensitive to boundary condition locations and that effective placements can be difficult to find without an automated method.

1 | Introduction

Topology optimization [1] is a computational design method used to automatically find an optimal distribution of material that minimizes an objective function while satisfying a set of optimization constraint functions. Originally introduced as a method for maximizing the stiffness of static structures [2], it required the user to specify the configuration of applied loads and rigid supports which would then remain fixed and unchanging during the optimization process. This feature persisted when topology optimization was later applied to the synthesis of compliant mechanisms [3, 4]. Compliant mechanism design has since become a common use of the method, with a wide variety of applications ranging from morphing wings [5] to micro-manipulators [6–9], flexures [10], switching mechanisms [11, 12], and more [13]. Yet it still remains standard practice for the boundary conditions to be predetermined by the user based only on intuition, despite the fact that in mechanism design the configuration of the boundary conditions has a significant influence on the resulting optimal geometry and its motion. Hence, topology optimization's effectiveness as a tool for designing compliant mechanisms can be significantly limited by requiring the user to manually choose boundary conditions.

Up until recently, efforts to incorporate the configuration of boundary conditions as automatically optimized design variables in topology optimization have been limited to only the supports, with the location and direction of the applied load still chosen by the user of the software. This has mostly been applied to static structures [14–18], with a few examples of compliant

mechanism design [19–21] where gains of up to a 77% increase in mechanism performance were shown in the study by Buhl [19]. Optimizing the applied load location and direction in topology optimization to maximize the performance of structural designs was first accomplished by Alacoque and James [22], where the material distribution, support locations, load location, and load orientation were simultaneously optimized to maximize the output displacement of a compliant displacement inverter mechanism. A 150% gain in performance was achieved over the optimal design generated using the conventionally chosen boundary conditions. Since then, Lee and Xie [23, 24] have adopted similar methods for optimizing applied loading conditions, achieving more efficient structural designs.

The results for compliant mechanism design in [22] show substantial improvements in objective function values when using variable boundary conditions; however, the linear elastic modeling used in the study has certain limitations when designing mechanisms which undergo large motions. The assumption of infinitesimal strain in linear elasticity is invalidated when large displacements or rotations are present in the deformed configuration of the body, and the theory is only able to model straight-line motions and linear load–displacement curves. Topology optimization using nonlinear elastic physics accurately models large deformations and produces different optimal material distributions than with linear analysis. Buhl et al. [25] modeled large displacements with small strains and showed that differences in designs can be large between linear and nonlinear analysis, especially in some cases with snap-through and buckling responses. Bruns and Tortorelli [26], who used a hyperelastic material model to capture geometric and material nonlinearities, produced a different compliant gripper design by using nonlinear modeling compared to linear. In a more recent study, Wallin et al. [27] optimized several different measures of stiffness, each producing different structures when loads levels are high enough to cause large displacements. Nonlinear elastic topology optimization also makes it possible to synthesize compliant mechanisms with more complex nonlinear behaviors such as bistability [28–30], tailored nonlinear load–displacement curves [31, 32], and nonlinear output paths [33, 34]. These behaviors are typically modeled using displacement-controlled algorithms, while the linear elastic studies [22–24] with variable boundary conditions apply the forces directly. However, none of the aforementioned studies on structures or compliant mechanisms, using either linear or nonlinear elasticity, attempt to optimize the location of a prescribed displacement boundary condition. Thus, to perform these interesting nonlinear optimizations with variable boundary conditions, a new method for a variable input displacement must be developed in addition to extending the previous methods of variable boundary conditions to nonlinear elastic physics.

In this work we develop the formulations necessary for optimizing the boundary condition configuration simultaneously with material distribution in displacement-controlled, nonlinear elastic topology optimization. This study is the first application of variable loads and supports to nonlinear elastic physics, and it includes the following novel contributions: The super-Gaussian projection function method from [22] is extended to control the effective locations of the boundary conditions by scaling the magnitudes of body forces and elastic support spring constants applied to each finite element. For complete control over the input actuation behavior and to enable the optimization of different nonlinear behaviors, we formulate a new displacement-controlled iterative solver that enables a continuously variable input displacement location and orientation. Novel adjoint sensitivity equations are derived by adding the displacement-control solver's constraint functions to the augmented Lagrangian function, allowing the use of gradient-based optimizers such as the Method of Moving Asymptotes (MMA) [35]. We use our method to generate compliant mechanisms using several different types of objective function and compare the improved performance of the resulting designs to those obtained using boundary conditions from other nonlinear topology optimization studies in the literature, demonstrating for the first time the significant advantages of variable boundary conditions in this setting.

The rest of the paper is organized as follows. Section 2 describes the density-based topology optimization formulation and the super-Gaussian projection method used to vary the boundary condition configuration. Section 3 describes the nonlinear, hyperelastic finite element formulation and the modifications to the residual and external load vector formulations that facilitate the variable-position displacement-control algorithm, which is outlined in Section 4. The adjoint sensitivity analysis is performed in Section 5 to derive generic derivative formulas that can be applied to any optimization objective or constraint function. In Section 6 we present and analyze the optimization results, and in Section 7 we discuss conclusions and the possibilities for future research directions.

2 | Topology Optimization With Boundary Conditions as Design Variables

The topology optimization problem we are considering is illustrated in Figure 1, which shows an elastic body in its undeformed configuration. The material density $\boldsymbol{\rho}$ is distributed within a design domain and is supported at locations $\left(X_s^{(i)}, Y_s^{(i)}\right)$. All degrees of freedom of displacement are set to zero as a boundary condition in the supported regions to represent fixed fastening components such as bolts, rivets, or welds. The number of "fasteners" is fixed and does not change during an optimization. A linear guided actuator applies a force to a location $\left(X_f, Y_f\right)$ to create a prescribed input displacement $\boldsymbol{U}_{\text{in}}$ at an angle θ and with stroke length $\|\boldsymbol{U}_{\text{in}}\|$. This input displacement causes an output displacement U_{out} at some other point on the body. Circular fastening components at the load and support locations are represented by movable, solid, non-designable regions of material. Other fixed solid or void non-design regions may also be present in the domain. The goal is to find the material distribution, support locations, actuator location, and actuator angle which achieves the desired mechanism motion along with any specified structural characteristics. The design space is parameterized by a concatenated vector of the design variables, $\boldsymbol{\zeta}$, consisting of both the material density distribution and the boundary condition configuration:

$$\boldsymbol{\zeta} = \left[\boldsymbol{\rho}\ \boldsymbol{X}_s\ \boldsymbol{Y}_s\ X_f\ \ Y_f\ \ \theta\right] \tag{1}$$

The continuous representation of the elasticity problem shown in Figure 1 is discretized using finite element analysis, creating a

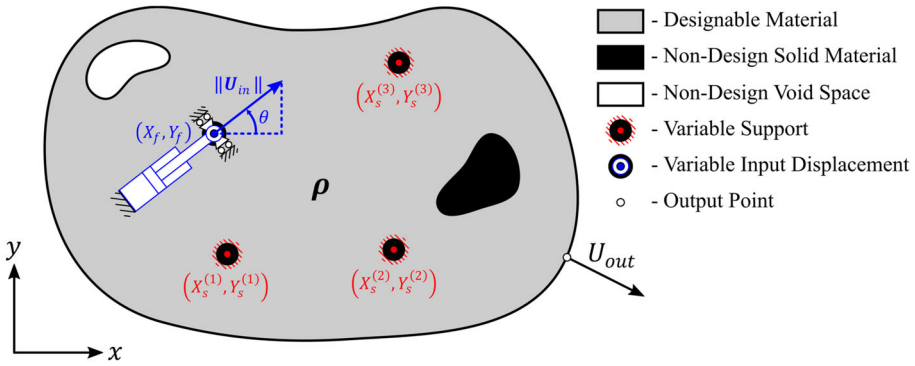

FIGURE 1 | The general design problem for compliant mechanisms with variable boundary conditions. The goal is to find a material distribution $\boldsymbol{\rho}$ along with support locations $\boldsymbol{X}_s, \boldsymbol{Y}_s$, actuator location X_f, Y_f, and actuator orientation θ which minimizes or maximizes a given objective function.

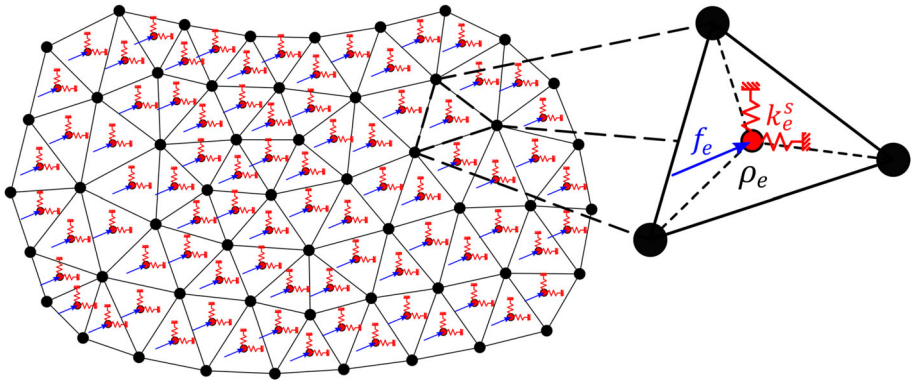

FIGURE 2 | A design domain discretized by finite elements. Forces are applied to each element centroid and spring elements connect the elements to rigid supports.

mesh of N_e elements, shown in Figure 2. Each element e in the mesh is assigned a material density design variable ρ_e which can vary continuously from 0 to 1, representing an interpolation from empty space to solid material.

2.1 | Filtering of Density Design Variables

We use the standard density filtering method of topology optimization to avoid checkerboard patterns and mesh dependence of designs [26]:

$$\tilde{\boldsymbol{\rho}} = \boldsymbol{W}\boldsymbol{\rho} \tag{2}$$

where the vector $\tilde{\boldsymbol{\rho}}$ contains the filtered densities, obtained by multiplying the vector of density design variables with the $N_e \times N_e$ filter matrix $\boldsymbol{W}$. The entries of the filter matrix are calculated as

$$W_{ei} = \frac{w_{ei} V_i}{\sum_{i=1}^{N_e} w_{ei} V_i} \tag{3}$$

where the weight factors w_{ei} are computed based on the filter radius, $r_{\min}$, and the distance between centroids of elements e and i, $\Delta(e, i)$:

$$w_{ei} = \max\left(0, r_{\min} - \Delta(e, i)\right) \tag{4}$$

In Equation (3), V_i is the element volume. Note also that the repeated indices do not imply summation.

Typically, the filtered densities are taken as the effective physical densities used to evaluate the element stiffness properties and solve the elasticity problem. However, in this work we perform one additional processing step to combine the filtered densities with the variable non-design regions at the boundary conditions points. This process is described in subsection 2.3.

2.2 | Variable Loads and Supports

The boundary conditions must also be formulated as smooth and continuous functions of the design variables. To achieve this, elastic supports are applied everywhere in the domain with the intent that their stiffness will be varied to control where the structure is effectively restrained. Areas with supports of high stiffness will simulate rigid fixtures, and areas with supports of low or zero stiffness will leave the structure free to deform. To this end, linear elastic springs are connected from rigid boundaries to the elements' centroids to resist deformation in each degree of freedom. This adds no additional degrees of freedom to the finite element model. Each element is assigned a corresponding spring constant for its support, k_e^s, the value of which will depend on the support location design variables. With the same idea for the applied load, a force magnitude f_e is assigned to each element centroid. The angle the forces are applied at will be determined by the displacement-control algorithm and is not the same as the input displacement angle θ, since the actuator is guided and generates a lateral reaction force.

The effective locations of the boundary conditions are controlled using a super-Gaussian projection function [22, 36], which is a feature-mapping technique [37] based on distance functions. Given a field of distance to points or lines, the function outputs a field of a specified value within shapes of given radius that smoothly transition to zero in the direction of increasing distance. The function is plotted and labeled in Figure 3 for a simple one-dimensional distance function to a single point. Its general form is given by

$$G = Ab^{-\left(\frac{d^2}{r^2}\right)^P} \tag{5}$$

where d is the distance field, A is the value the function takes within each geometric feature, P is a parameter that controls the sharpness of the transition region, and r sets the length from zero distance to where the function equals the value specified by the parameter b:

$$G(d = r) = \frac{A}{b} \tag{6}$$

For this work, we use the smoothed minimum distance from each element centroid $\left(\overline{X}_e, \overline{Y}_e\right)$ to a number of points (X_i, Y_i) placed within the design domain:

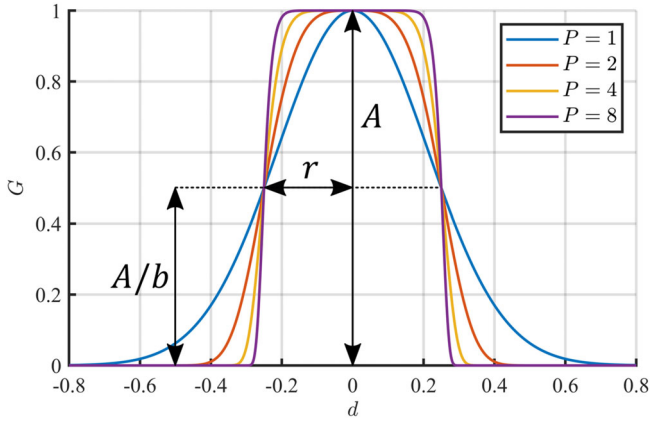

FIGURE 3 | The super-Gaussian projection function with a 1D distance function for $A = 1$, $b = 2$, $r = 0.25$, and several values of the exponent P.

$$d_e = \left(\sum_i \left(\sqrt{\left(\overline{X}_e - X_i\right)^2 + \left(\overline{Y}_e - Y_i\right)^2}\right)^{-Q}\right)^{-\frac{1}{Q}} \tag{7}$$

where larger values of Q more closely approximate the true minimum. An example of this distance field, and the circular features the super-Gaussian projection function creates from it, is shown in Figure 4.

Now applying the projection function to control the distribution of elastic support stiffness, the design variables $\boldsymbol{X}_s$ and $\boldsymbol{Y}_s$ are used in Equation (7) to create the smoothed distance field. The coefficient A from Equation (5) is chosen such that the spring constants within the support geometry are equivalent to the shear stiffness of a solid support material with shear modulus G_s and thickness t_s. Considering an element of area A_e with a force f_e applied to the centroid of its cross section, the average shear stress in its support material is

$$\tau_e = \frac{f_e}{A_e} \tag{8}$$

Using a linear stress–strain relationship and assuming a small transverse displacement δ_e gives:

$$\frac{\delta_e}{t_s} G_s = \frac{f_e}{A_e} \tag{9}$$

The equivalent spring constant for the solid support material of the element is determined by rearranging and comparing to the equation for a spring:

$$\frac{G_s}{t_s} A_e \delta_e = f_e \tag{10}$$

Substituting the spring constant for the coefficient of the super-Gaussian function yields:

$$k_e^s = \frac{G_s}{t_s} A_e b^{-\left(\frac{d_e^2}{r^2}\right)^P} \tag{11}$$

For the applied load, the design variables X_f and Y_f are used for the distance function:

$$d_e = \sqrt{\left(\overline{X}_e - X_f\right)^2 + \left(\overline{Y}_e - Y_f\right)^2} \tag{12}$$

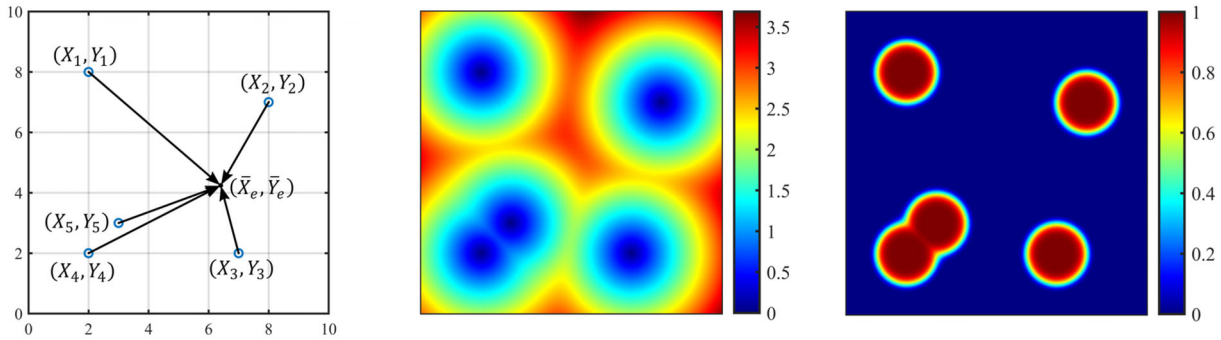

FIGURE 4 | Projecting circular shapes from zero dimensional points. (a) Several points in the domain with their spatial coordinates and distances to the centroid of an element e. (b) The smooth distance field for the set of points, using $Q = 10$. (c) The super-Gaussian projection of the smooth distance field, using $A = 1$, $b = 2$, $r = 1$, and $P = 4$.

The distribution of load magnitude is

$$f_e = A_f b^{-\left(\frac{d_e^2}{r^2}\right)^P} \tag{13}$$

where the coefficient A_f is chosen such that the total load applied in the design domain is equal to 1:

$$A_f = \frac{1}{\sum_{e=1}^{N_e} b^{-\left(\frac{d_e^2}{r^2}\right)^P} V_e} \tag{14}$$

The coefficient A_f is assumed to be a constant and is only calculated once before beginning the optimization. The Newton–Raphson solver is then able to use a load intensity factor to scale the magnitude of the applied force, which is described in detail in the following sections.

2.3 | Variable Non-Design Regions

Moveable non-design regions around the boundary condition points are used to ensure well-defined compliant hinges are generated at the supports [22], and so that the forces from the actuator are always applied to solid material. To create them, a density distribution is projected onto the mesh using the super-Gaussian function. The distance function is created using both the support location and load location design variables. The coefficient A is set to 1 for solid material, giving the projected density distribution as

$$\hat{\rho}_e = b^{-\left(\frac{d_e^2}{r^2}\right)^P} \tag{15}$$

The projected densities and the filtered densities are then combined into the physical density field using a smooth maximum function:

$$\overline{\rho}_e = \left(\tilde{\rho}_e^Q + \hat{\rho}_e^Q\right)^{\frac{1}{Q}} \tag{16}$$

The physical densities in each element, $\overline{\rho}_e$, define the design that is analyzed in the finite element analysis and is the density field that is presented as the results of topology optimization. It is used to calculate the elastic modulus of each element using the Solid Isotropic Material with Penalization (SIMP) interpolation scheme:

$$E_e = E_{\min} + \overline{\rho}_e^p (E_0 - E_{\min}) \tag{17}$$

where $E_{\min}$ is a small value of stiffness given to void elements to avoid singular stiffness matrices in the finite element analysis, E_0 is the elastic modulus of the solid material, and p is the SIMP penalization factor which reduces the stiffness-to-weight ratio of intermediate density elements causing the optimizations to converge to structures made up of mostly solid material.

3 | Nonlinear Finite Element Model

In each iteration of the topology optimization process, nonlinear finite element analysis is used to simulate the deformation of the current design. The results of the analysis are then used to characterize the performance of the mechanism by calculating the values of the optimization objective and constraint functions. The general theory and background of nonlinear finite element analysis, including the strong form of the nonlinear elastic boundary value problem, its equivalent weak form, and its discretization by the finite element method, can be found in references such as [38–40] and here we only present the equations that are needed for implementation. We use the total Lagrangian formulation and a hyperelastic material model to represent compliant mechanisms made from a rubber-like flexible material.

3.1 | Equilibrium Equations

Typically, displacement-controlled nonlinear finite element analysis finds the vector of nodal displacements, $\boldsymbol{U}$, and a load intensity factor, λ, describing an equilibrium configuration of the structure where the externally applied loads are balanced with the internally generated forces. The balance of forces is represented by the residual vector:

$$\boldsymbol{R} = \lambda \boldsymbol{F}^{\text{ext}} - \boldsymbol{F}^{\text{int}} = \boldsymbol{0} \tag{18}$$

where $\boldsymbol{F}^{\text{ext}}$ is a reference vector of external nodal loads and $\boldsymbol{F}^{\text{int}}$ is the vector of internal nodal forces. For the work here, considering the applied element forces f_e to be body forces per unit volume, the reference external load vector is given as an assembly of the body forces [41]:

$$\boldsymbol{F}^{\text{ext}} = \bigwedge_{e=1}^{N_e} \int_{V_e} \boldsymbol{N}^T f_e \begin{bmatrix} \cos(\phi) \\ \sin(\phi) \end{bmatrix} dV_e \tag{19}$$

where the symbol $\bigwedge$ denotes the finite element assembly operation, the matrix $\boldsymbol{N}$ contains the finite element shape functions, and ϕ is the (unknown) angle the load is applied at. Assuming that the element force f_e is evenly distributed to the nodes results in:

$$\boldsymbol{F}^{\text{ext}} = \bigwedge_{e=1}^{N_e} \frac{f_e}{n_n} \boldsymbol{\Phi} V_e \tag{20}$$

where n_n is the number of nodes in the element and $\boldsymbol{\Phi} = \begin{bmatrix} \cos(\phi) & \sin(\phi) & \dots \end{bmatrix}^T$ is a direction vector that applies the force components to the appropriate degrees of freedom. Since the load angle ϕ that causes the given input displacement angle is unknown, it must be found by the displacement-control algorithm. To facilitate this, the external load vector is split into x and y components and the residual vector is rewritten as

$$\boldsymbol{R} = \lambda_x \boldsymbol{F}_x^{\text{ext}} + \lambda_y \boldsymbol{F}_y^{\text{ext}} - \boldsymbol{F}^{\text{int}} = \boldsymbol{0} \tag{21}$$

where

$$\boldsymbol{F}_x^{\text{ext}} = \bigwedge_{e=1}^{N_e} \frac{f_e}{\sqrt{2} n_n} \boldsymbol{\Phi}_x V_e \tag{22}$$

$$\boldsymbol{F}_y^{\text{ext}} = \bigwedge_{e=1}^{N_e} \frac{f_e}{\sqrt{2} n_n} \boldsymbol{\Phi}_y V_e \tag{23}$$

with the direction vectors as $\boldsymbol{\Phi}_x = \begin{bmatrix} 1 & 0 & \dots \end{bmatrix}^T$ and $\boldsymbol{\Phi}_y = \begin{bmatrix} 0 & 1 & \dots \end{bmatrix}^T$. The two load intensity factors λ_x and λ_y are then determined by the displacement-control algorithm such that the specified input displacement $\boldsymbol{U}_{\text{in}}$ is achieved. This algorithm is described in the next section.

3.2 | Strain Energy Interpolation and Hyperelastic Material Model

In topology optimization with nonlinear finite element analysis, a numerical issue occurs where elements with low stiffness can distort excessively and prevent convergence of the iterative Newton–Raphson solver. Earlier works circumvented this issue by ignoring nodes surrounded by void elements in the solver's convergence criteria [25, 33], by removing elements with low densities from the mesh [42], or by optimizing the connectivity of completely solid elements [43]. More recent methods include interpolating to linear elastic modeling at low densities [44], adding an extra hyperelastic stiffness term to unstable elements [45, 46], and stabilizing low-density elements in a way that is also able to simulate contact between solid regions [47]. Here, we use the linear-nonlinear strain energy interpolation scheme by Wang et al. [44] since it is relatively simple to implement and is able to converge for adequately large deformations in most cases. The interpolation function in this method is a smoothed Heaviside step function that is dependent on the physical density of each element:

$$\gamma_e = \frac{\tanh(\beta\rho_0) + \tanh\left(\overline{\rho}_e^p - \rho_0\right)}{\tanh(\beta\rho_0) + \tanh(\beta(1-\rho_0))} \tag{24}$$

where the parameter β affects the sharpness of the step function and ρ_0 controls the location of the threshold. The deformation gradient in each element is computed as:

$$\boldsymbol{F} = \boldsymbol{I} + \gamma_e \nabla \boldsymbol{U}_e \tag{25}$$

where $\boldsymbol{I}$ is the identity matrix and ∇ is the gradient operator. Like in the original implementation of the method [44], we use a modified Neo-Hookean hyperelastic material model [48] where the second Piola-Kirchhoff stress is implemented as:

$$\boldsymbol{S} = \lambda_0\left(2J^2 - J\right)\boldsymbol{C}^{-1} + \mu_0\left(\boldsymbol{I} - \boldsymbol{C}^{-1}\right) \tag{26}$$

and the entries of the constitutive matrix are implemented using index notation as:

$$D_{ijkl} = \lambda_0\left(2J^2 - J\right)C_{ij}^{-1}C_{kl}^{-1} + \left(\mu_0 - \lambda_0\left(2J^2 - J\right)\right)\left(C_{ik}^{-1}C_{jl}^{-1} + C_{il}^{-1}C_{jk}^{-1}\right) \tag{27}$$

The constants λ_0 and μ_0 are the Lamé parameters for a unit elastic modulus, J is the determinant of the deformation gradient, and $\boldsymbol{C} = \boldsymbol{F}^T\boldsymbol{F}$ is the right Cauchy-Green deformation tensor.

This interpolation method allows the Newton–Raphson iterations to converge in many cases. However, intermediate density elements may still become unstable if displacements are too large which limits the size of actuator strokes that can be applied in the framework of this paper. Low-stiffness elements may also become unstable if external loads are applied to them, but this is prevented by the moveable solid non-design region placed under the input point.

3.3 | Internal Force Vector

The internal force vector is an assembly of the element force vectors for the continuum part of the mesh, interpolated between nonlinear and linear modeling, plus the internal forces of the support springs. Writing the nonlinear element force vectors as $\boldsymbol{f}_e^{NL}$ and the linear counterparts as $\boldsymbol{f}_e^{L}$, the global internal force vector is assembled as:

$$\boldsymbol{F}^{\text{int}} = \bigwedge_{e=1}^{N_e}\left(\gamma_e\boldsymbol{f}_e^{NL} + \left(1-\gamma_e^2\right)\boldsymbol{f}_e^{L}\right) + \boldsymbol{K}_s\boldsymbol{U} \tag{28}$$

where the interpolation formulation between the nonlinear and linear element force vectors has been adopted from [11, 49, 50] where it was successfully implemented for nonlinear problems. $\boldsymbol{K}_s$ is the global stiffness matrix of the supports, assembled as:

$$\boldsymbol{K}_s = \bigwedge_{e=1}^{N_e}\frac{k_e^s}{n_n}\boldsymbol{I}$$

The size of the identity matrix $\boldsymbol{I}$ is equal to the number of degrees of freedom in the element. The nonlinear element force vector is given by the integration

$$\boldsymbol{f}_e^{NL} = E_e\int_{V_e}\boldsymbol{B}_N^T\boldsymbol{S}dV_e \tag{29}$$

where $\boldsymbol{B}_N$ is the nonlinear strain–displacement matrix. The linear element force vector is given by

$$\boldsymbol{f}_e^{L} = E_e\int_{V_e}\boldsymbol{B}^T\boldsymbol{D}_0\boldsymbol{\varepsilon}dV_e \tag{30}$$

where $\boldsymbol{B}$ is the linear strain–displacement matrix, $\boldsymbol{D}_0$ is the linear constitutive matrix for unit stiffness, and $\boldsymbol{\varepsilon}$ is the linear strain in the element.

3.4 | Tangent Stiffness Matrix

The tangent stiffness matrix is also an interpolation between nonlinear and linear modeling for the continuum part of the mesh [11, 49, 50], with the contribution from the linear elastic supports added to it:

$$\boldsymbol{K}_T = \bigwedge_{e=1}^{N_e}\left(\gamma_e^2\boldsymbol{k}_e^{NL} + \left(1-\gamma_e^2\right)\boldsymbol{k}_e^{L}\right) + \boldsymbol{K}_s \tag{31}$$

The nonlinear element stiffness matrix is

$$\boldsymbol{k}_e^{NL} = E_e\int_{V_e}\left(\boldsymbol{B}_N^T\boldsymbol{D}\boldsymbol{B}_N + \boldsymbol{B}_G^T\boldsymbol{\Sigma}\boldsymbol{B}_G\right)dV_e \tag{32}$$

where $\boldsymbol{\Sigma}$ is a matrix of second Piola-Kirchhoff stresses and $\boldsymbol{B}_G$ is another strain–displacement matrix [38]. The linear element stiffness matrix is integrated using

$$\boldsymbol{k}_e^{L} = E_e\int_{V_e}\boldsymbol{B}^T\boldsymbol{D}_0\boldsymbol{B}dV_e \tag{33}$$

4 | Variable-Position Displacement-Control Algorithm

The displacement-control algorithm is an incremental-iterative (predictor–corrector) method [51, 52], where the input displacement is incremented in steps to obtain points along the load–displacement curve. For each increment, an iterative cycle solves for the unknown state variables $\boldsymbol{U}$, λ_x, and λ_y

to restore static equilibrium to within a given tolerance. The residual, Equation (21), provides the solver constraint functions necessary to find the unknown nodal displacements, $\boldsymbol{U}$. However, an additional two are needed to solve for the load intensity factors, λ_x and λ_y. Since the input displacement location is continuously variable and can be applied anywhere within the mesh, including at the interior of elements, this provides an additional two data points in the displacement field that can be used to define the extra solver constraint functions. The x and y components of displacement at the input actuation point can be obtained by using the finite element shape functions:

$$\begin{bmatrix} U_x^{\text{in}}(X_f, Y_f) \\ U_y^{\text{in}}(X_f, Y_f) \end{bmatrix} = \boldsymbol{N}(X_f, Y_f)\boldsymbol{U} \tag{34}$$

Then, the solver constraints can be defined as the difference between the displacement field at the input actuation point and the given input displacement, the two of which will be equal to each other at equilibrium:

$$\boldsymbol{c} = \boldsymbol{N}(X_f, Y_f)\boldsymbol{U} - \boldsymbol{U}_{\text{in}} = \begin{bmatrix} U_x^{\text{in}}(X_f, Y_f) - \|\boldsymbol{U}_{\text{in}}\| \cos(\theta) \\ U_y^{\text{in}}(X_f, Y_f) - \|\boldsymbol{U}_{\text{in}}\| \sin(\theta) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{35}$$

The ability to control displacement at any position in the domain, rather than only displacements at discrete nodal locations like in previous methods such as [51, 52], is the key modification of the existing algorithm that allows for the optimization of a continuously variable input displacement location.

The displacement-control algorithm then proceeds as follows. For each displacement step i, the predictor phase begins by calculating reference displacement increment vectors from the two reference load vectors. These are solved for simultaneously using:

$$[\boldsymbol{K}_T]^{i-1}\left[\Delta\boldsymbol{U}^a \;\; \Delta\boldsymbol{U}^b\right] = \left[\boldsymbol{F}_x^{\text{ext}} \;\; \boldsymbol{F}_y^{\text{ext}}\right] \tag{36}$$

where the state at increment $i-1$ is from the previously converged displacement step, or else is the unloaded and undeformed state of the structure. With a given displacement step size $\Delta\overline{U}$, which is some fraction of the final input displacement length $\|\boldsymbol{U}_{\text{in}}\|$, the solver constraint functions of Equation (35) can be rewritten using the reference displacement increment vectors of Equation (36) as

$$\begin{bmatrix} \Delta U_x^a(X_f, Y_f) & \Delta U_x^b(X_f, Y_f) \\ \Delta U_y^a(X_f, Y_f) & \Delta U_y^b(X_f, Y_f) \end{bmatrix} \begin{bmatrix} \Delta\lambda_x \\ \Delta\lambda_y \end{bmatrix} = \begin{bmatrix} \Delta\overline{U}\cos(\theta) \\ \Delta\overline{U}\sin(\theta) \end{bmatrix} \tag{37}$$

where the coefficient matrix contains the x and y components of the displacement increment vectors taken at the input actuation location. These are found by using the finite element shape functions, similar to Equation (34). After solving Equation (37) for the load step increments $\Delta\lambda_x$ and $\Delta\lambda_y$, the last step in the predictor phase is to update the state variables as:

$$\boldsymbol{U}^i = \boldsymbol{U}^{i-1} + \Delta\lambda_x\Delta\boldsymbol{U}^a + \Delta\lambda_y\Delta\boldsymbol{U}^b \tag{38}$$

$$\lambda_x^i = \lambda_x^{i-1} + \Delta\lambda_x \tag{39}$$

$$\lambda_y^i = \lambda_y^{i-1} + \Delta\lambda_y \tag{40}$$

The tangent stiffness matrix, internal force vector, and residual vector are also updated at this point using the state of the structure predicted in Equations (38–40).

If the norm of the residual is greater than a given tolerance, the algorithm moves into an inner loop. This is the corrector phase. For corrector iteration j, three reference displacement increment vectors are calculated using:

$$[\boldsymbol{K}_T]^j\left[\Delta\boldsymbol{U}^a \;\; \Delta\boldsymbol{U}^b \;\; \Delta\boldsymbol{U}^c\right]^j = \left[\boldsymbol{F}_x^{\text{ext}} \;\; \boldsymbol{F}_y^{\text{ext}} \;\; \boldsymbol{R}^j\right] \tag{41}$$

Using the shape functions to get the values of the reference displacement increment vectors at the input load location, they are used in the solver constraint functions of Equation (35) to solve for the load intensity increments:

$$\begin{bmatrix} \Delta U_x^a(X_f, Y_f) & \Delta U_x^b(X_f, Y_f) \\ \Delta U_y^a(X_f, Y_f) & \Delta U_y^b(X_f, Y_f) \end{bmatrix}^j \begin{bmatrix} \Delta\lambda_x \\ \Delta\lambda_y \end{bmatrix}^j = \begin{bmatrix} -\Delta U_x^c(X_f, Y_f) \\ -\Delta U_y^c(X_f, Y_f) \end{bmatrix}^j \tag{42}$$

The total displacement increment vector is then given by

$$\Delta\boldsymbol{U}^j = \Delta\lambda_x^j\Delta\boldsymbol{U}_j^a + \Delta\lambda_y^j\Delta\boldsymbol{U}_j^b + \Delta\boldsymbol{U}_j^c \tag{43}$$

and the state variables are updated as:

$$\boldsymbol{U}^{i,j+1} = \boldsymbol{U}^{i,j} + \Delta\boldsymbol{U}^j \tag{44}$$

$$\lambda_x^{i,j+1} = \lambda_x^{i,j} + \Delta\lambda_x^j \tag{45}$$

$$\lambda_y^{i,j+1} = \lambda_y^{i,j} + \Delta\lambda_y^j \tag{46}$$

The tangent stiffness matrix and internal force vector are reassembled again here, and the residual vector is recalculated. The corrector phase continues to iterate until the residual norm convergence criterion is satisfied, meaning the displacement step is converged, and the algorithm then moves to increment $i+1$. Once the algorithm converges at the given final displacement $\boldsymbol{U}_{\text{in}}$, it ends and outputs the results.

The corrector can fail to converge for various reasons. Possible causes include that the chosen step size is too large, that deformations larger than the interpolation scheme [44] is able handle have caused intermediate density elements to become unstable, or that a displacement limit point in the load–displacement curve has been encountered. To avoid these issues as much as possible, the algorithm is written to automatically try again with a bisected step length if the corrector passes a given maximum number of iterations. Resolving these issues would require better methods for stabilizing low-density elements, and a different solver such as an arc-length method [53] that can trace the load–displacement curve through limit points. These problems are left for future research.

5 | Adjoint Sensitivity Analysis

Gradient-based methods of optimization require the derivatives of the optimization objective and constraint functions with respect to each design variable. In topology optimization, the large number of design variables and the computational expense

of finite element analysis means that the gradient computations must be performed efficiently. The adjoint sensitivity analysis method is used here to derive analytical sensitivity formulas that are inexpensive to evaluate. We employ a novel approach where we add the displacement-control solver's constraint functions to the augmented Lagrangian function, rather than a residual vector for prescribed degrees of freedom which does not exist in the current method.

For any optimization objective or constraint function $f(\boldsymbol{\zeta}, \boldsymbol{\eta}(\boldsymbol{\zeta}))$, which is considered to be an *explicit* function of the design variables $\boldsymbol{\zeta}$ and the state variables $\boldsymbol{\eta}$, and where the state variables are considered *implicit* functions of the design variables, an augmented Lagrangian function $g(\boldsymbol{\zeta}, \boldsymbol{\eta}(\boldsymbol{\zeta}))$ is formed by multiplying the governing equations by Lagrange multipliers and adding these terms to the function f. Since the governing equations are equal to zero at equilibrium, the augmented Lagrangian function g is equivalent to the original function f.

In conventional displacement-controlled topology optimization with constant boundary conditions applied to discrete nodes, the residual vector contains all of the governing equations. These include the equations for the *free* degrees of freedom, where the displacements are unknown and are solved for, and the equations for the *prescribed* degrees of freedom, where the displacements are given while the external loads are the unknowns [26]. However, in the context of variable boundary conditions (the subject of this work), all degrees of freedom in the mesh are "free" with unknown displacements. Thus, only adding the residual vector in the augmented Lagrangian function is not enough to capture the total derivative, and another term is needed. This missing term of governing equations is the displacement-control solver's constraint functions, $\boldsymbol{c}$, that were defined in Equation (35).

With the state variables $\boldsymbol{\eta}(\boldsymbol{\zeta}) = \left[\boldsymbol{U}(\boldsymbol{\zeta})\ \boldsymbol{\lambda}(\boldsymbol{\zeta})\right]^T$, where $\boldsymbol{\lambda}(\boldsymbol{\zeta}) = \left[\lambda_x(\boldsymbol{\zeta})\ \lambda_y(\boldsymbol{\zeta})\right]^T$, the augmented Lagrangian function is formed by multiplying the residual and the solver constraint functions with Lagrange multipliers $\boldsymbol{\psi}_R$ and $\boldsymbol{\psi}_c$, respectively, and adding the terms to the original function:

$$g(\boldsymbol{\zeta}, \boldsymbol{U}(\boldsymbol{\zeta}), \boldsymbol{\lambda}(\boldsymbol{\zeta})) = f(\boldsymbol{\zeta}, \boldsymbol{U}(\boldsymbol{\zeta}), \boldsymbol{\lambda}(\boldsymbol{\zeta})) + \boldsymbol{\psi}_R^T \boldsymbol{R}(\boldsymbol{\zeta}, \boldsymbol{U}(\boldsymbol{\zeta}), \boldsymbol{\lambda}(\boldsymbol{\zeta})) + \boldsymbol{\psi}_c^T \mathbf{c}(\boldsymbol{\zeta}, \boldsymbol{U}(\boldsymbol{\zeta}), \boldsymbol{\lambda}(\boldsymbol{\zeta})) \tag{47}$$

Denoting total or implicit derivatives by the operator $d/d\zeta$, and partial or explicit derivatives by the operator $\partial/\partial\zeta$, the total derivative is taken with respect to an arbitrary design variable using the multivariable chain rule of calculus:

$$\frac{dg}{d\zeta} = \frac{\partial f}{\partial \zeta} + \frac{\partial f}{\partial \boldsymbol{U}}\frac{d\boldsymbol{U}}{d\zeta} + \frac{\partial f}{\partial \boldsymbol{\lambda}}\frac{d\boldsymbol{\lambda}}{d\zeta} + \boldsymbol{\psi}_R^T\left(\frac{\partial \boldsymbol{R}}{\partial \zeta} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}\frac{d\boldsymbol{U}}{d\zeta} + \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\lambda}}\frac{d\boldsymbol{\lambda}}{d\zeta}\right) + \boldsymbol{\psi}_c^T\left(\frac{\partial \mathbf{c}}{\partial \zeta} + \frac{\partial \mathbf{c}}{\partial \boldsymbol{U}}\frac{d\boldsymbol{U}}{d\zeta} + \frac{\partial \mathbf{c}}{\partial \boldsymbol{\lambda}}\frac{d\boldsymbol{\lambda}}{d\zeta}\right) \tag{48}$$

This expression is rearranged to isolate the implicit derivatives of the state variables:

$$\frac{dg}{d\zeta} = \frac{\partial f}{\partial \zeta} + \boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \zeta} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \zeta} + \left(\frac{\partial f}{\partial \boldsymbol{U}} + \boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \boldsymbol{U}}\right)\frac{d\boldsymbol{U}}{d\zeta} + \left(\frac{\partial f}{\partial \boldsymbol{\lambda}} + \boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\lambda}} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \boldsymbol{\lambda}}\right)\frac{d\boldsymbol{\lambda}}{d\zeta} \tag{49}$$

Since the residual vector and solver constraint functions are equal to zero, the values of the Lagrange multipliers are arbitrary and can be chosen freely without affecting the value of the sensitivity. The terms with the implicit derivatives can be eliminated by finding Lagrange multipliers that cause the sum of the terms inside the brackets to vanish:

$$\frac{\partial f}{\partial \boldsymbol{U}} + \boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \boldsymbol{U}} = \mathbf{0} \tag{50}$$

$$\frac{\partial f}{\partial \boldsymbol{\lambda}} + \boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\lambda}} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \boldsymbol{\lambda}} = \mathbf{0} \tag{51}$$

Solving Equations (50) and (51) simultaneously leads to the following systems of linear equations that yield the values of the Lagrange multipliers:

$$\boldsymbol{\psi}_c^T\left(\frac{\partial \mathbf{c}}{\partial \boldsymbol{\lambda}} - \left(\frac{\partial \mathbf{c}}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}^{-1}\right)\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\lambda}}\right) = -\left(\frac{\partial f}{\partial \boldsymbol{\lambda}} - \left(\frac{\partial f}{\partial \boldsymbol{U}}\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}}^{-1}\right)\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\lambda}}\right) \tag{52}$$

$$\boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} = -\left(\frac{\partial f}{\partial \boldsymbol{U}} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \boldsymbol{U}}\right) \tag{53}$$

where

$$\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}} = -\boldsymbol{K}_T \tag{54}$$

$$\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\lambda}} = \begin{bmatrix} \boldsymbol{F}_x^{\text{ext}} & \boldsymbol{F}_y^{\text{ext}} \end{bmatrix} \tag{55}$$

$$\frac{\partial \mathbf{c}}{\partial \boldsymbol{\lambda}} = \mathbf{0} \tag{56}$$

$$\frac{\partial \mathbf{c}}{\partial \boldsymbol{U}} = \boldsymbol{N}(X_f, Y_f) \tag{57}$$

Now, without ever having had to derive expressions for the implicit derivatives of the iteratively computed state variables, the total derivative of any function f is given by Equations (52–57) and the following formula:

$$\frac{dg}{d\zeta} = \frac{\partial f}{\partial \zeta} + \boldsymbol{\psi}_R^T\frac{\partial \boldsymbol{R}}{\partial \zeta} + \boldsymbol{\psi}_c^T\frac{\partial \mathbf{c}}{\partial \zeta} \tag{58}$$

Depending on the specific function f being implemented, only the three explicit derivatives $\partial f/\partial\zeta$, $\partial f/\partial \boldsymbol{U}$, and $\partial f/\partial\boldsymbol{\lambda}$ need to be found and plugged into Equations (52), (53), and (58), which is typically simple to do.

We note that the derivatives of the shape functions with respect to the input displacement coordinates, which appear in the derivative $\partial \mathbf{c}/\partial\zeta$, are discontinuous for shape functions that are not smooth across element boundaries. However, for our implementation using first-order elements with piecewise linear shape functions, we did not encounter any oscillations or other problems in the convergence of the topology optimization process that could be attributable to this.

6 | Numerical Examples

In this section, the framework described in Sections 2 through 5 is implemented in MATLAB and is used to generate several types of compliant mechanisms based on examples from nonlinear topology optimization in the literature. Three types of design

problems are investigated: displacement maximization, bistability, and path generation. We run each example twice: once with the boundary conditions fixed at the initial locations chosen by the other studies, which represent the boundary conditions good designers might use based on their intuition or other conventional methods, and the second time allowing our algorithm to automatically adjust the boundary conditions to find optimal placements. The starting distribution of material is uniform in both cases. The performance of the designs obtained using variable boundary conditions is compared to those obtained using fixed boundary conditions to quantify the advantages of the method over conventional topology optimization. The results shown in this section may not be globally optimal solutions, because different local minima may be obtained by using different starting positions of the boundary conditions. For practical design problems, multiple optimizations should be performed using different starting configurations and the best results can then be chosen from the set.

Several quantities of interest are used to define the various optimization objective and constraint functions in the following example problems. The displacements at the output points of the mechanisms, at any point m on the load–displacement curve where the state variables have been solved for, are found using

$$U_{\text{out}}^{(m)} = \boldsymbol{L}^T \boldsymbol{U}^{(m)} \tag{59}$$

where $\boldsymbol{L}$ is a vector of all zeros, except for a value of one at the degrees of freedom that are selected by the user. The input force applied by the actuator is in-line with the input displacement and is found from the applied force's x and y components as

$$F_{\text{in}}^{(m)} = \sqrt{\left(\lambda_x^{(m)}\right)^2 + \left(\lambda_y^{(m)}\right)^2} \sin\left(\theta + \tan^{-1}\left(\frac{\lambda_x^{(m)}}{\lambda_y^{(m)}}\right)\right) \tag{60}$$

and the reaction force generated by the guide structure is the force perpendicular to the input displacement, given by

$$F_p^{(m)} = \sqrt{\left(\lambda_x^{(m)}\right)^2 + \left(\lambda_y^{(m)}\right)^2} \cos\left(\theta + \tan^{-1}\left(\frac{\lambda_x^{(m)}}{\lambda_y^{(m)}}\right)\right) \tag{61}$$

The volume fraction of material distributed in the domain is found from the physical densities:

$$V_f = \frac{\sum_{e=1}^{N_e} \overline{\rho}_e V_e}{\sum_{e=1}^{N_e} V_e} \tag{62}$$

These quantities can be used in many different ways to control the characteristics of the mechanisms that are generated. The output displacements can be used in objective functions to create large geometric advantages, or to define specific output paths. Used in optimization constraint functions, they can control any unwanted motions of the mechanism. Similarly, the input forces can be used to control the shape of the load–displacement curve, or to limit the amount of force that can be applied. Real mechanical actuators and their guide structures have finite strength, and the optimizer will often attempt to use extremely large input loads if it is allowed to. To prevent this, we place upper-bound constraints on the magnitudes of F_{in} and F_p in each example. The volume fraction is used as an upper-bound constraint to limit the mass of the resulting designs.

In some cases, particularly in those with snap-through behavior or longer structural members loaded in compression, we have encountered oscillations in the optimization objective function caused by buckling bifurcations in the load–displacement curves, which prevent convergence. Having multiple stable configurations of a structure is a discontinuity in the design space and can prevent convergence if a different solution is found in each iteration, such as if a member has no strong preference for buckling to one side compared to the other. The same buckling problem is reported by Bruns and Sigmund [54] as a tendency for the optimization to oscillate between structures that do and do not have the buckling response. Li et al. [31] also mention that non-unique solutions are a challenge for considering slender structures with local and global instability. Thus, this issue is not unique to the variable boundary condition formulation of this work. However, the decision to use the guided actuator formulation was partially motivated by the problem since it adds resistance to buckling by preventing lateral deflection, and this appreciably alleviated the issue of oscillating objective functions caused by different buckling modes occurring in consecutive optimization iterations. In some of the following examples, other strategies to avoid unwanted buckling behaviors have been used as well, such as by placing optimization constraints on the load–displacement curve or by using a larger density filter radius to avoid slender features.

All examples are generated on a desktop computer with unstructured meshes of planar 3-node triangular finite elements under plane stress conditions. Unstructured meshes, as opposed to structured grids of 4-node rectangular elements, are used so that design domains of any shape can be meshed easily. The Method of Moving Asymptotes (MMA) [35] is used as the optimizer with its default settings. Objective and constraint functions are scaled so that values of less than 100 are sent to the MMA function [55]. The physical scale of the examples we use ensure that the values of position design variables are on the order of one or less. The input displacement angle design variable is implemented in radians with bounds equal to the starting angle plus and minus π, so that its numerical value is appropriately scaled for MMA as well. Depending on the problem, bounds on the positional design variables are either set to a distance of $2r$ greater than the maximum or less than the minimum coordinates of the design domain (to allow the optimizer to effectively remove supports by moving them outside of the design domain), or to a distance of r less than the maximum and greater than the minimum (to prevent the optimizer from removing supports). The specific choice of bounds used for the positional variables are noted in the following subsections for each example. Table 1 shows the common parameters used in all of the following examples. For the structure of the compliant mechanisms, we model a flexible rubber-like material by using an elastic modulus of $E_0 = 10$ MPa and a Poisson's ratio of $\nu = 0.49$. A much stiffer material is used for the support structure with elastic modulus $E_s = 2000$ MPa and Poisson's ratio $\nu_s = 0.3$, giving the shear modulus as $G_s = E_s/(2(1+\nu_s))$. The SIMP stiffness penalty parameter is set to a constant of $p = 3$. For the strain energy interpolation between linear and nonlinear modeling, we use values for β ranging from 500 to 2000 depending on

TABLE 1 | Optimization Parameters: Common.

Parameter	Symbol	Value
Design Material Elastic Modulus	E_0	10 MPa
Design Material Minimum Elastic Modulus	$E_{\min}$	$E_0 \times 10^{-9}$
Design Material Poisson's Ratio	ν	0.49
Support Material Elastic Modulus	E_s	2000 MPa
Support Material Poisson's Ratio	ν_s	0.3
SIMP Penalty	p	3
Energy Interpolation Threshold	ρ_0	0
Super-Gaussian Base	b	2
Super-Gaussian Superscript	P	4
Smooth Min/Max Superscript	Q	12

the problem. We use $\rho_0 = 0$ in all problems, as this resulted in better convergence to solid-void solutions compared with the higher values of this parameter used in the original paper for the method [44], which tended to leave more gray areas in some problems. In the super-Gaussian projection function, we use $b = 2$ so that the radius r corresponds to the contour of 50% density, and a superscript of $P = 4$ to create flat-topped projections with sufficiently smooth edges. The parameter for the smooth minimum and maximum functions is set to $Q = 12$. In the displacement-control algorithm, the convergence tolerance for the residual norm in the corrector loop is set to 1×10^{-6} and the maximum number of corrector iterations is set to 20. The topology optimization is considered converged when all constraint functions are satisfied and the average change in the density design variables compared to the previous iteration is less than 1×10^{-4} [56].

In the design plots, the elements are shaded according to their physical density values, $\overline{\rho}_e$. The optimized designs will always have intermediate density (gray) elements present due to the density filtering of Section 2.1, which enforces a minimum length scale over which the physical density can transition from solid to void. Larger density filter radii leave more gray elements in the final designs. The support locations are shown by red dots and the load location is shown by a blue dot. The input displacement direction is shown by the blue arrow. The mesh has been colored red and blue for the supports and load, respectively, and the opacity of the colored meshes represents the relative magnitude of the spring stiffness or load magnitude to visually display how the boundary conditions are applied.

6.1 | Maximum Displacement

Maximizing the displacement at specified output points given a fixed input force or displacement is a common design objective in topology optimization of compliant mechanisms. A frequently used benchmark problem is a gripper mechanism with a single input force and two supports. In the majority of cases in the literature, a square design domain is used with the load applied at the middle of the left edge and the supports applied at the upper left and lower left corners [4, 33, 34, 46, 57, 58]. We use this configuration for the starting condition, which is illustrated in Figure 5.

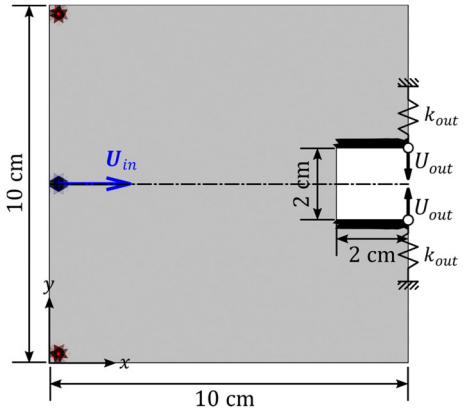

FIGURE 5 | The initial conditions for the maximum displacement gripper problem.

TABLE 2 | Optimization parameters: Gripper.

Parameter	Symbol	Value
Element Size	h	1.5 mm
Number of Elements	N_e	9,812
Domain Thickness	t	1 cm
Support Thickness	t_s	1 cm
Density Filter Radius	$r_{\min}$	3 mm
Super-Gaussian Radius	r	2.5 mm
Input Displacement Length	$\|\boldsymbol{U}_{\text{in}}\|$	5 mm
Output Spring Stiffness	k_{out}	300 N/m
Energy Interpolation Sharpness	β	500
Move limits: $\boldsymbol{\rho}$		0.2
Move limits: $\boldsymbol{X}_s, \boldsymbol{Y}_s$		2.5 mm
Move limits: X_f, Y_f		2.5 mm
Move limit: θ		5°

The design domain is a 10×10 cm square with a 2×2 cm square cut away from the center of the right edge to form the jaws of the gripper. The output points are at the upper and lower right corners of the cut-out, and maximizing the output displacement causes these two points to move together as shown by the black arrows. Spring elements of stiffness k_{out} are attached to the output points to simulate the stiffness of a workpiece, which ensures the mechanism can transfer a reasonable amount of load from the input. Fixed non-design regions of solid material are placed along the jaws to guarantee that the entire gripping surface is present in the final design.

The optimization parameters specific to this example are listed in Table 2. A mesh of approximately 10,000 elements is used, with the design and support material thicknesses both set to 1 cm. The radius of the density filter is set to 3 mm, the radii of the boundary condition points are each set to 2.5 mm, the actuator stroke is set to 5 mm, and the output springs are given a stiffness of 300 N/m. The move limits, which set the maximum allowed change in the design variables in a single iteration of the optimization, are set to 0.2 for the densities, 2.5 mm for the boundary condition locations, and 5 degrees for the input displacement angle.

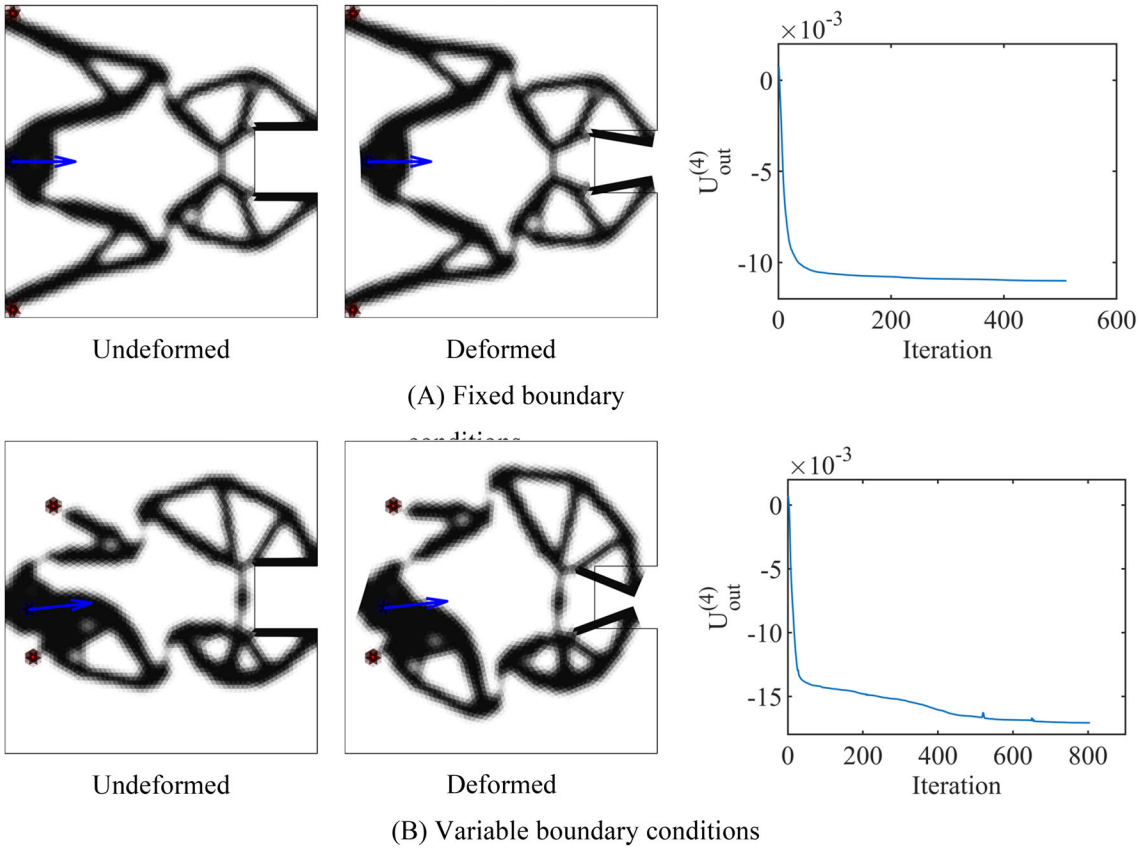

FIGURE 6 | Results of the maximum displacement gripper problem. The undeformed and deformed configurations are shown for each design. (a) The design generated with the boundary conditions fixed in their initial configuration. (b) The design generated when allowing the boundary conditions to move from their starting positions.

Solving for four displacement steps in the nonlinear finite element analysis, the optimization problem consists of maximizing the displacement at the final step. The volume fraction of material is limited to 30%, and the input forces are constrained at each of the four steps such that the upper limit of actuator force increases linearly with input displacement, helping to prevent the optimization from attempting to take advantage of buckling responses. At the maximum stroke length, the maximum allowed input forces are set to 30 N and 7.5 N for the in-line and lateral components, respectively. Mathematically, this optimization problem is given by the statement:

$$\begin{aligned}
&\underset{\zeta}{\text{maximize}}: \; U_{\text{out}}^{(4)} \\
&\text{subject to}: \; V_f < 0.3 \\
&\qquad F_{\text{in}}^{(m)} < \tfrac{30}{4} m \text{ N}, \qquad\qquad m = 1,2,3,4 \\
&\qquad -\frac{7.5}{4} m \text{ N} < F_p^{(m)} < \frac{7.5}{4} m \text{ N}, \qquad m = 1,2,3,4
\end{aligned} \tag{63}$$

where the output displacement objective function has been calculated using two non-zero degrees of freedom in the selector vector of Equation (59), making it a sum of the two output displacements. The minimum and maximum allowed values for the load and support location design variables are set to keep them inside the design domain, at least a distance of r from the edges of the square.

The optimization is performed once with the boundary conditions constrained to remain in their initial conditions, resulting in the design shown in the first row of Figure 6 which achieves an objective function value of $U_{\text{out}}^{(4)} = 1.09$ cm. This is a typical outcome for the gripper problem and similar designs can be seen in many other papers [4, 33, 46, 57, 58]. The second row of Figure 6 shows the results after running the optimization with unconstrained boundary conditions. A much different, asymmetrical design is produced where the boundary conditions have moved further inside the domain with the actuator at an angle. For the same input stroke length, the variable boundary condition design achieves an objective function value of $U_{\text{out}}^{(4)} = 1.71$ cm, a 57% improvement over the fixed boundary condition design. The force-displacement curves are plotted in Figure 7, showing that many of the constraints on the force have become active with variable boundary conditions, but none have in the case of fixed boundary conditions due to the more restricted design space. The optimization converges in 511 iterations with fixed boundary conditions, and in 623 with variable boundary conditions. Although not all problems have required more iterations with variable boundary conditions, this may be happening in some cases because a larger design space is being traversed with small move limits on the positional variables.

We also observe that despite using a symmetrical initial design with symmetrical boundary conditions, the optimizer was able to find a higher-performing asymmetrical design. As noted by

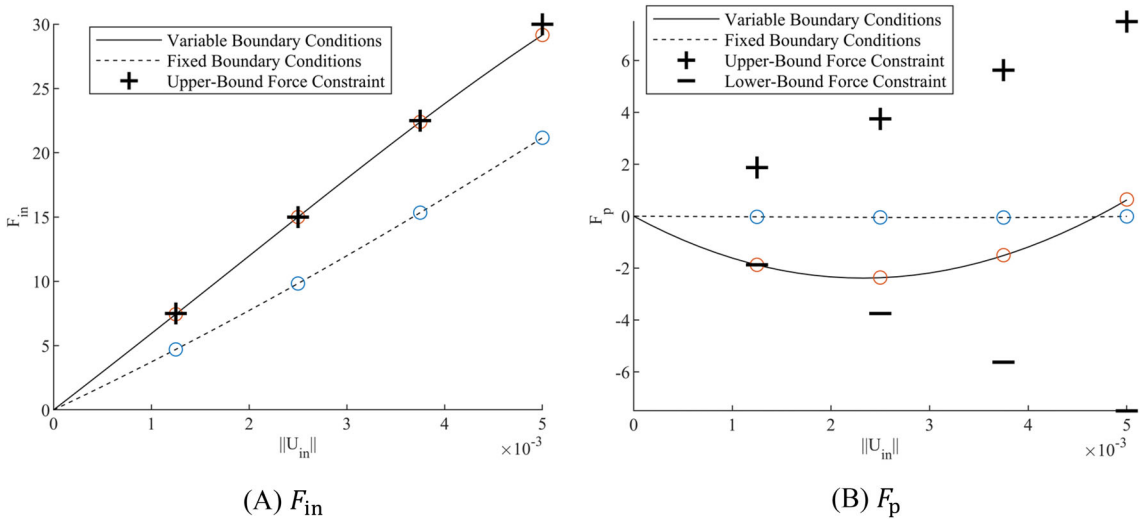

FIGURE 7 | Force-displacement curves for the maximum displacement gripper designs. (a) Force applied by the actuator, F_{in}. (b) Lateral reaction force on the actuator's guide structure, F_{p}.

Bruns et al. [53], this may be because of physically asymmetric deformation modes or numerical round-off errors that introduce asymmetry in the design. It is not clear exactly where the asymmetry is being introduced, however we also experienced loss of symmetry in our previous study using linear elasticity [22], which makes it seem likely that the cause is at least partly due to numerical round-off errors and not entirely due to asymmetrical nonlinear deformation modes.

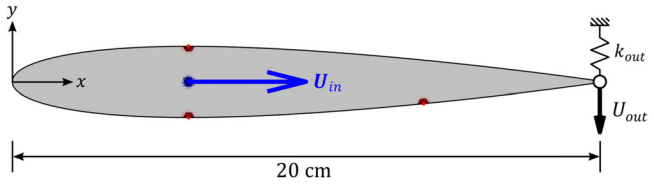

FIGURE 8 | The initial conditions for the bistable airfoil problem.

6.2 | Bistability

For the bistable structure design problem, we base our test case on the example of a bistable morphing airfoil by Bhattacharyya et al. [30]. The idea of the design problem is to generate a monolithic aileron structure and mechanism that, once actuated, snaps through and passively maintains a high-camber configuration with no additional load input.

Topology optimization of snap-through or bistable structures performs best when using a finite element analysis solution method that is capable of tracing load–displacement curves with snap-back trajectories, such as the arc-length method by Bruns et al. [53]. However, the arc length method as presented in reference [53] is load-controlled and is not able to solve for multiple displacement components, which we need for the guided actuator formulation of this work. Instead, we use our displacement-control algorithm while placing optimization constraints on the load–displacement curve to avoid snap-back behaviors that would otherwise cause the solver to diverge.

TABLE 3 | Optimization parameters: Bistable airfoil.

Parameter	Symbol	Value
Element Size	h	1 mm
Number of Elements	N_e	7,680
Domain Thickness	t	1 cm
Support Thickness	t_s	1 cm
Density Filter Radius	$r_{\min}$	4 mm
Super-Gaussian Radius	r	2 mm
Input Displacement Length	$\|\boldsymbol{U}_{\text{in}}\|$	2.5 mm
Output Spring Stiffness	k_{out}	100 N/m
Energy Interpolation Sharpness	β	2000
Move limits: ρ		0.05
Move limits: $\boldsymbol{X}_s, \boldsymbol{Y}_s$		0.5 mm
Move limits: X_f, Y_f		0.5 mm
Move limit: θ		1°

The design domain for the problem is a NACA 0012 airfoil with a chord length of 20 cm, shown in Figure 8. The output point is at the trailing edge, with the goal of making it move downwards a distance of 5 mm against the resistance of a spring. The initial boundary conditions are those selected by Bhattacharyya et al. [30], which represent the best configuration the authors were able to find by manual trial and error. Two supports are placed at the top and bottom edges, at a distance of 6 cm from the leading edge of the airfoil. The actuator is placed on the chord line between the supports and is pointed directly to the right. A third support functions as the aileron hinge and is placed at the bottom edge, 6 cm away from the trailing edge.

The parameters used for the optimization are shown in Table 3. A larger density filter radius is used to avoid slender structures and reduce buckling-related oscillations. The move limits are

also reduced to obtain smoother convergence with less severe oscillations.

The optimization problem consists of solving for multiple points along the load–displacement curve and minimizing the force of the last point, with the goal of achieving a negative value. Other input forces are constrained to tailor the rest of the force-displacement curve to avoid snap-back, and a constraint is used on the output displacement to cause the deflection of the aileron. Solving for eight evenly distributed displacement points, the formulation we use is written as:

$$\begin{aligned}
&\underset{\zeta}{\text{minimize}} : F_{\text{in}}^{(8)} \\
&\text{subject to} : V_f < 0.4 \\
&\qquad U_{\text{out}}^{(8)} > 5 \text{ mm} \\
&\qquad F_{\text{in}}^{(1)} > 2 \text{ N} \\
&\qquad F_{\text{in}}^{(m)} < 15 \sin\left(\pi \frac{m}{6}\right) + 5 \text{ N}, \qquad m = 1, 2, \ldots, 6 \\
&\qquad -5 \text{ N} < F_p^{(m)} < 5 \text{ N}, \qquad m = 1, 2, \ldots, 8
\end{aligned} \tag{64}$$

The upper and lower limits on the load and support locations are set such that they may move outside of the design domain if doing so improves the objective function. This allows the optimizer to completely remove unnecessary supports. The sine wave upper-bound on the input forces $F_{\text{in}}^{(m)}$ causes the load–displacement curve to turn downwards earlier than it otherwise would, eliminating a snap-back behavior that caused the displacement-control algorithm to fail.

The resulting designs and their convergence plots are shown in Figure 9. The number of iterations to convergence is 470 for fixed boundary conditions, but only 380 for variable boundary conditions. The rate of convergence in the first 100 iterations also appears to be significantly faster, showing that allowing the algorithm to adjust the boundary conditions can improve convergence in some cases. Since a snap-through response is itself a buckling behavior, we encountered oscillations in the objective functions of both cases. In the study on topology optimization for snap-through mechanisms by Bruns and Sigmund [54], they also report these oscillations and attribute them to the optimization switching between designs that do have a snap-through response and designs that do not. Since the stopping criteria is based on average changes in element density, the optimization is still able to converge with these small objective function oscillations.

The load–displacement curves of each design are shown in Figure 10, which were obtained in a post-analysis where the input displacement was solved in small increments up to a longer stroke length of 4 mm. The applied actuator load–displacement curve for the fixed boundary condition design shows that it fails to achieve bistability, with a positive force of 1.0 N at the eighth displacement control point, while the variable boundary condition design successfully reaches a negative force of −4.6 N. With the ability to adjust the load and support positions, the optimization increased the geometric advantage of the mechanism by moving the actuator closer to the upper support. With the shorter moment arm around the support, the actuator also must apply a greater force to create the given input displacement, resulting in a stronger snap-through effect. The center point of the upper support also moves slightly outside of the design domain, reducing the size of the supported area and allowing the link to rotate about it more easily. Again, for this design problem more constraints on force have become active with variable boundary conditions compared to fixed boundary conditions.

The results of this problem show that a relatively minor change in boundary conditions can mean the difference between a failed and a successful design. In the previous work of Bhattacharyya et al. [30], the thickness-to-chord ratio of the airfoil had been doubled to give a larger design domain. Here, we used the true NACA

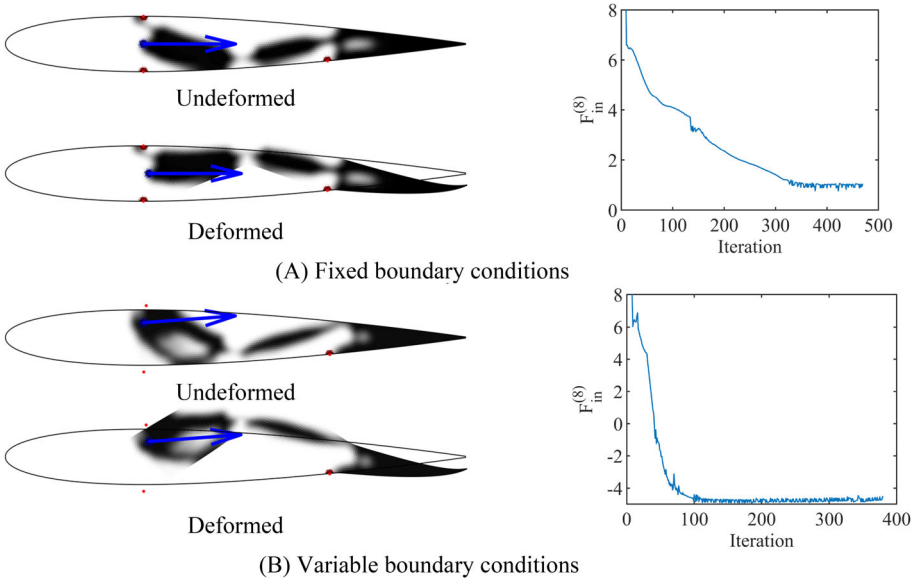

FIGURE 9 | Results of the bistable airfoil problem. Undeformed configurations, deformed configurations, and optimization convergence plots are shown for each design. (a) The design generated with the boundary conditions fixed in their initial configuration. (b) The design generated when allowing the boundary conditions to move from their starting positions.

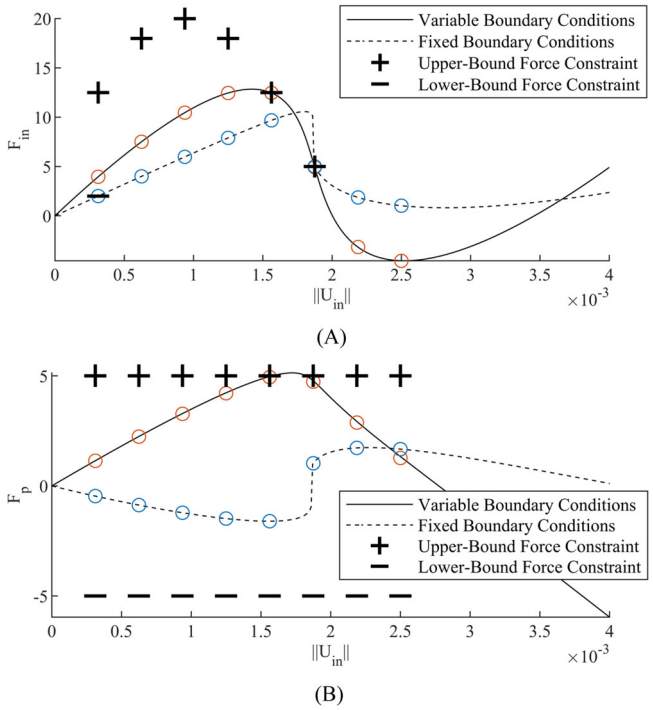

FIGURE 10 | The force-displacement curves of the bistable airfoil designs. The circular markers show the original eight displacement control points that were solved for in the optimization process. (a) Force applied by the actuator, F_{in}. (b) Lateral reaction force on the actuator's guide structure, F_{p}.

0012 airfoil which is a more difficult optimization problem. Using the configuration that had been successful in the previous designs of [30] did not work, and it was necessary to optimize the boundary conditions to achieve bistability.

6.3 | Path Generation

In path generation problems, the entire path of the output point is specified and the mechanism is optimized to follow that motion as closely as possible. We investigate two test cases based on the horizontal line generation problem from the work of Pederson et al. [33] and the morphing wing design problem from the research of Reinisch et al. [34]. The path generation formulations in these studies work by defining a set of m precision points that the output location of the mechanism should move through at each of the m input displacement states. An error function is used for the objective function which minimizes the difference between the actual path and the desired path. Instead of using springs attached to the output points to achieve structurally stiff mechanisms, the path generation problems use multiple load cases where, in each additional load case i, counter forces are applied to the output point to resist its motion. By simultaneously minimizing the output path error in all load cases, fully solid and structurally stiff mechanisms are generated which can perform useful work against external forces.

6.3.1 | Line Generator

The design domain and initial conditions for the horizontal line generator problem are shown in Figure 11. A rectangular design

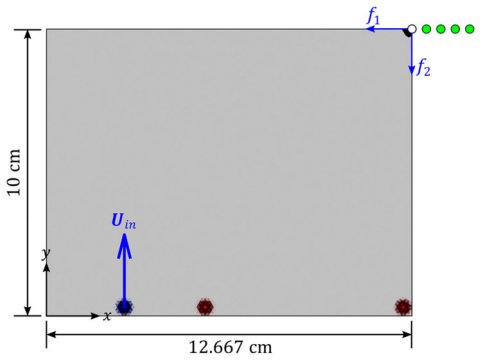

FIGURE 11 | The initial conditions for the horizontal line generator path generation problem. The precision points are shown by the green dots.

TABLE 4 | Optimization parameters: Line generator.

Parameter	Symbol	Value
Element Size	h	1.2 mm
Number of Elements	N_e	20,290
Domain Thickness	t	1 cm
Support Thickness	t_s	1 cm
Density Filter Radius	$r_{\min}$	2.4 mm
Super-Gaussian Radius	r	3 mm
Energy Interpolation Sharpness	β	500
Input Displacement Length	$\|\boldsymbol{U}_{\text{in}}\|$	1 cm
Horizontal Counter Force	f_1	5 N
Vertical Counter Force	f_2	5 N
Move limits: ρ		0.2
Move limits: $\boldsymbol{X}_s, \boldsymbol{Y}_s$		3 mm
Move limits: X_f, Y_f		3 mm
Move limit: θ		2°

domain is used with the output point at the upper-right corner. Four precision points are used to define a straight output path moving 2 cm to the right. Two counter load cases are used, where the first applies a horizontal force f_1 pointing to the left and the second applies a vertical force f_2 pointing downwards. The boundary conditions are initialized at the bottom edge with the configuration used in [33]. Solid non-design regions are placed around the output point to avoid placing the counter forces on low-density elements, which would cause the nonlinear analysis to diverge. The specific parameters for the problem are listed in Table 4.

The optimization problem formulation for the horizontal line generator is

$$\begin{aligned}
\underset{\zeta}{\text{minimize}}: \quad & \sum_{i=1}^{3}\sum_{m=1}^{4}\left[\left(X_{\text{out}}^{(m,i)} - X_{\text{out}}^{*(m)}\right)^2 + \left(Y_{\text{out}}^{(m,i)} - Y_{\text{out}}^{*(m)}\right)^2\right] \\
\text{subject to}: \quad & V_f < 0.2 \\
& F_{\text{in}}^{(m)} < 20\ \text{N}, \qquad m = 1,2,3,4 \\
& -5\ \text{N} < F_p^{(m)} < 5\ \text{N}, \qquad m = 1,2,3,4
\end{aligned} \tag{65}$$

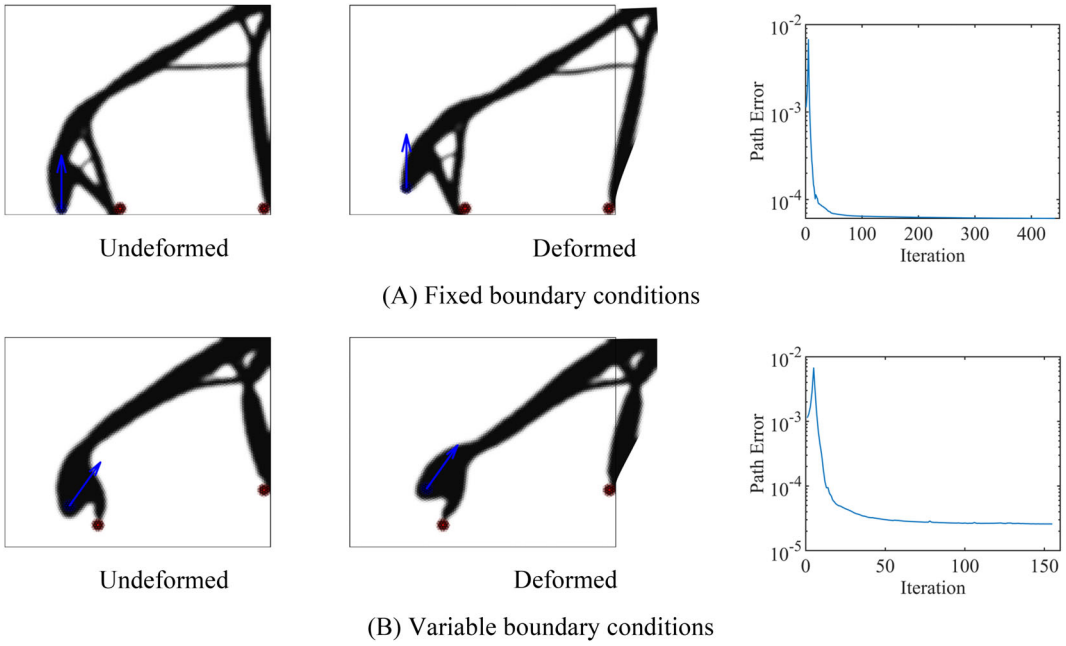

FIGURE 12 | Results of the horizontal line generator problem. The undeformed and deformed configurations are shown for each design. (a) The design generated with the boundary conditions fixed in their initial configuration. (b) The design generated when allowing the boundary conditions to move from their starting positions.

where $X_{\text{out}}^{(m,i)}, Y_{\text{out}}^{(m,i)}$ are the positions of the output point at each input displacement m, for each load case i, and $X_{\text{out}}^{*(m)}, Y_{\text{out}}^{*(m)}$ are the corresponding coordinates of the precision points. The load and support locations are also constrained to stay within the design domain, at least a distance r from the edges.

Running the problem for the fixed and variable boundary condition cases results in the designs shown in Figure 12. The number of iterations to convergence is 442 and 155 for fixed and variable boundary conditions, respectively, again showing that in some cases fewer design iterations may be required. The fixed boundary condition problem generates a design similar to the line generator in the study this problem is based on [33]. For the variable boundary condition design, the actuator and each support move upwards by several centimeters, the actuator and the nearby support move closer to one another horizontally, and the actuator rotates clockwise by 35°. These changes in the boundary conditions lead to a design which more closely follows the straight path of the precision points, giving a value of the path error objective function that is only 43% the value of the fixed boundary condition design's. Using the average distance of the output point from the precision points as a performance metric, the variable boundary condition design's path compared to the fixed boundary condition design's is, on average for all three load cases, only 68% of the distance to the precision points, or 1.47 times closer.

To show the improvement visually, the actual output paths of the mechanisms are plotted next to the precision points in Figure 13 by running the finite element analysis on the final designs using small displacement steps. The path of the variable boundary condition design is closer to the straight line of the precision points in the second and third load cases, but is slightly farther for the first load case where the average distance of the output point along the path from the four precision points is 0.55 mm for the fixed boundary condition design and 0.63 mm for the variable boundary condition design. For the second load case, these average distances are 3.1 and 1.9 mm, and for the third load case they are 2.2 and 1.5 mm. The load–displacement curves are shown in Figure 14.

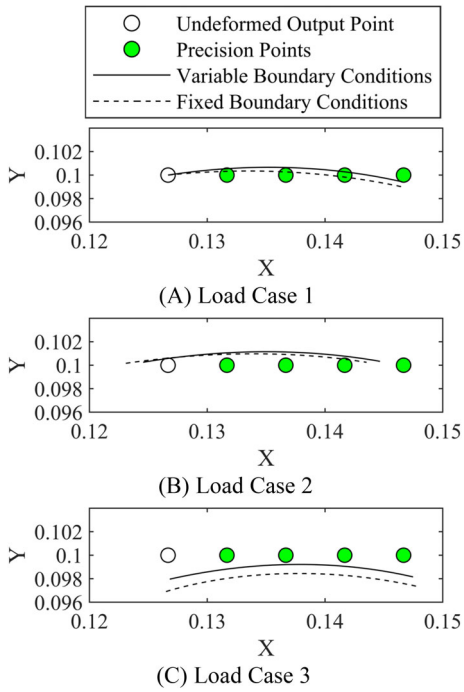

FIGURE 13 | The actual output paths of the horizontal line generator mechanisms. The units of the axes are in meters. (a) The first load case with no counter forces applied. (b) The second load case with the horizontal counter force f_1 applied at the output. (c) The third load case with the vertical counter force f_2 applied at the output.

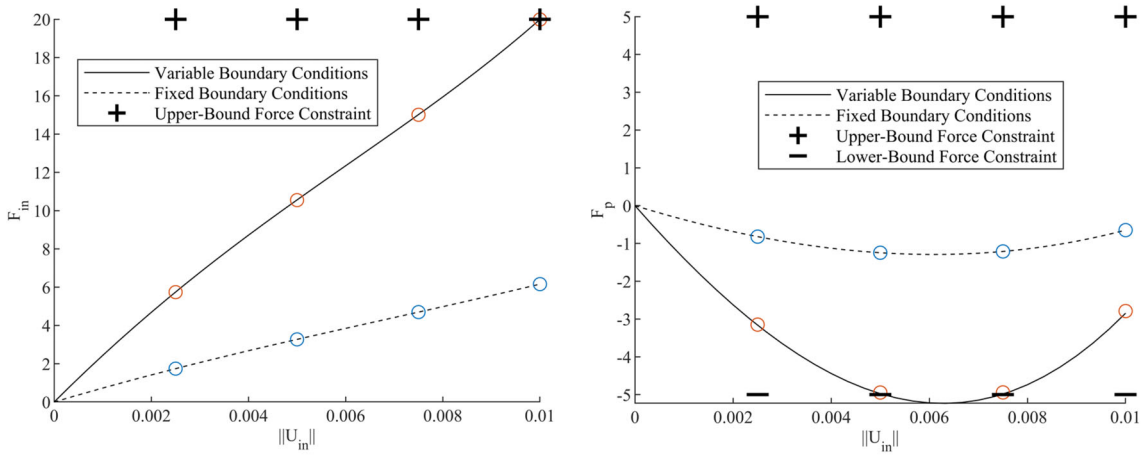

FIGURE 14 | Force-displacement curves for the horizontal line generator designs. (a) Force applied by the actuator, F_{in}. (b) Lateral reaction force on the actuator's guide structure, F_{p}.

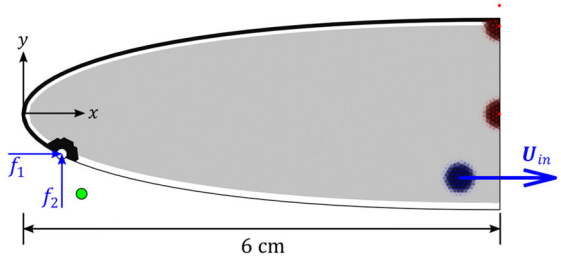

FIGURE 15 | The initial conditions for the morphing wing path generation problem. The precision point is shown by the green dot.

6.3.2 | Morphing Wing

The setup of the morphing wing problem is shown in Figure 15. We use the leading 30% of a 20 cm long NACA 0012 airfoil as the design domain. The output point and boundary conditions are placed based on the configuration chosen in [34], where two supports represent the wing spar and the guided actuator is placed near the bottom surface of the wing pointing towards the rear. A solid non-design region is placed around the upper part of the domain to create the skin that will bend as the wing changes shape. To keep the skin attached to the spar, a third support is placed slightly above the upper-right corner and kept fixed, even in the variable boundary condition case. A void non-design region is placed between the skin and the interior of the wing to ensure the internal mechanism remains separate from the skin, except at the attachment to the output point. A single precision point is placed 2.5 mm behind and 5 mm below the output point, which represents the position this point on the wing should move to in its morphed state. Two counter load cases are used, with a horizontal force pointing to the right to represent a drag force, and a vertical force pointing upwards representing a lifting force. A solid non-design region is placed around the output point. The parameters for this problem are listed in Table 5.

The optimization problem formulation for the morphing wing is written as

TABLE 5 | Optimization parameters: Morphing wing.

Parameter	Symbol	Value
Element Size	h	0.5 mm
Number of Elements	N_e	10,828
Domain Thickness	t	1 cm
Support Thickness	t_s	1 cm
Density Filter Radius	$r_{\min}$	1 mm
Super-Gaussian Radius	r	2 mm
Energy Interpolation Sharpness	β	500
Input Displacement Length	$\|\boldsymbol{U}_{\text{in}}\|$	2 mm
Horizontal Counter Force	f_1	1 N
Vertical Counter Force	f_2	1 N
Move limits: ρ		0.2
Move limits: $\boldsymbol{X}_s, \boldsymbol{Y}_s$		1 mm
Move limits: X_f, Y_f		1 mm
Move limit: θ		5°

$$\begin{aligned} \underset{\zeta}{\text{minimize}}: \quad & \sum_{i=1}^{3}\left[\left(X_{\text{out}}^{(i)} - X_{\text{out}}^{*}\right)^2 + \left(Y_{\text{out}}^{(i)} - Y_{\text{out}}^{*}\right)^2\right] \\ \text{subject to}: \quad & V_f < 0.3 \\ & F_{\text{in}} < 20 \text{ N} \\ & -5 \text{ N} < F_p < 5 \text{ N} \end{aligned} \tag{66}$$

In this problem, the load and support locations are completely unconstrained and may move outside of the design domain if the optimizer finds it advantageous.

The results of the optimization for fixed and variable boundary conditions are shown in Figure 16, and the output paths of the optimal designs for each load case are shown in Figure 17. Load–displacement curves are shown in Figure 18. Once again, the fixed boundary condition design is similar to the result of the study the problem is based on [34]. For the variable

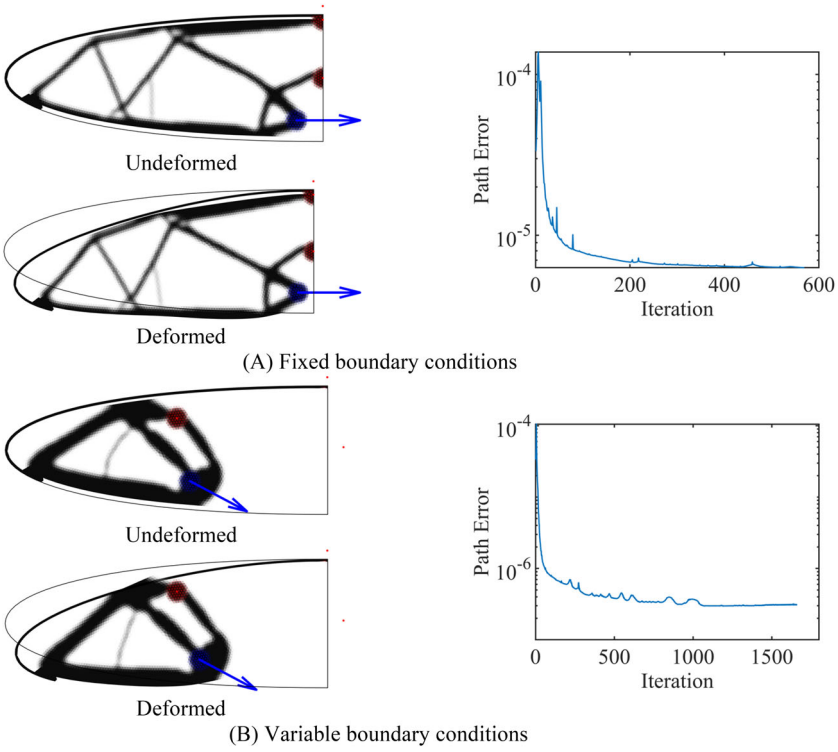

FIGURE 16 | Results of the morphing wing leading edge problem. (a) The design generated with the boundary conditions fixed in their initial configuration. (b) The design generated when allowing the boundary conditions to move from their starting positions.

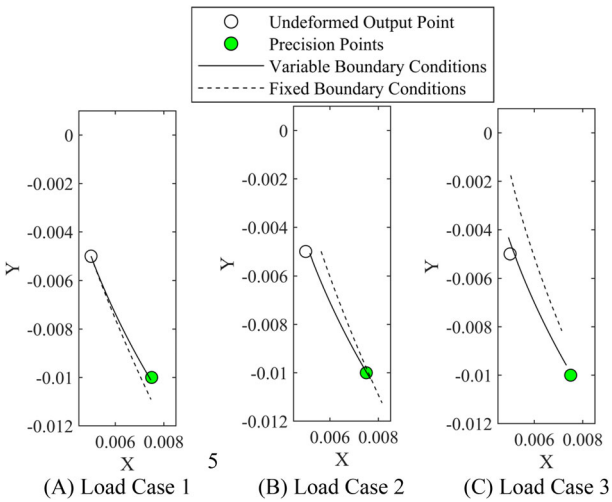

FIGURE 17 | The actual output paths of the morphing wing leading edge mechanisms. The units of the axes are in meters. (a) The first load case with no counter forces applied. (b) The second load case with the horizontal counter force f_1 applied at the output. (c) The third load case with the vertical counter force f_2 applied at the output.

boundary condition design, the actuator and one of the two supports move closer to the output point, while the other support is moved out of the domain allowing its material to be used in the mechanism's structure instead. The output path passes 4.8 times closer to the precision point, at 21% the distance of the fixed boundary condition design averaged across the three load cases. The paths plotted in Figure 17 show that in the first load case with no counter force the variable boundary condition design morphs to a distance of only 0.11 mm from the target position, while the fixed boundary condition design misses it by eight times that distance, at 0.91 mm from the precision point. In the counter-loaded cases, the variable boundary condition design still comes close to the target position at distances of 0.31 mm for load case two and 0.45 mm for load case three, while the fixed boundary condition design is pushed further away by the counter forces and only comes to 1.4 mm (4.5 times further) and 1.9 mm (4.2 times further) from the precision point in the second and third load cases, respectively.

In this case, a significantly higher number of iterations are required for variable boundary conditions, at 1660 compared to 569 for fixed boundary conditions which may be due to the boundary condition points traveling a large distance across the design domain with small move limits.

For both the line generator and morphing wing variable boundary condition examples, the increases in performance under minimal external loading are due to the better placement of the supports and actuator for achieving the target motions. Comparing the counter-loaded cases to the first load case, the larger increases in distance from the precision points for the fixed boundary condition designs show that the changes in the load and support layouts have also allowed for stiffer optimal structures. These results thus clearly demonstrate the importance of boundary condition placement for both the kinematic and structural aspects of compliant mechanism performance.

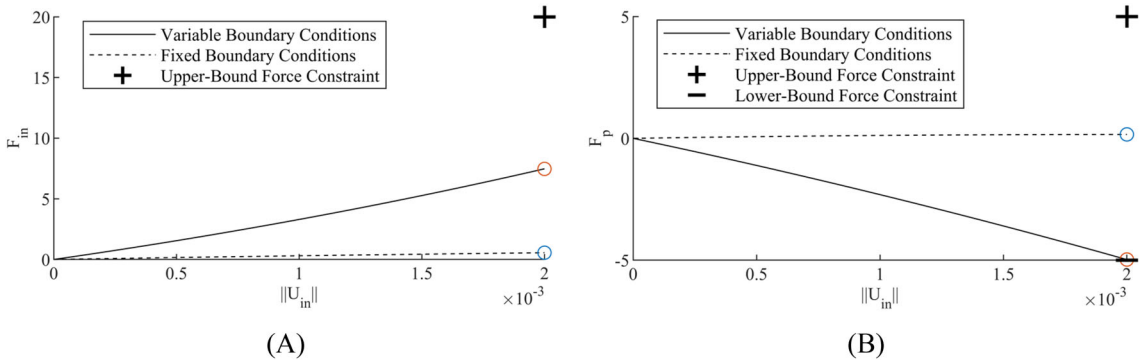

FIGURE 18 | Force-displacement curves for the morphing wing designs. (a) Force applied by the actuator, F_{in}. (b) Lateral reaction force on the actuator's guide structure, F_p.

7 | Conclusions

In this work, we presented a nonlinear elastic, displacement-controlled topology optimization framework that incorporates the spatial configuration of the boundary conditions as design variables. Expanding on work done for variable loads and supports in linear elasticity [22], we formulated the hyperelastic finite element model, nonlinear governing equations, iterative Newton–Raphson solution algorithm, and adjoint sensitivity equations necessary to achieve a continuously variable input displacement location and orientation. Coupling this capability with variable support locations, we tested the method on several types of nonlinear compliant mechanism synthesis problems. In an effort to make an unbiased comparison of the designs our algorithm produced to those that typical topology optimization might generate in the hands of a skilled engineer, we ran each of our test problems both with and without variable boundary conditions. The initial locations of the boundary conditions were set as the configurations chosen by other researchers in previous compliant mechanism optimization studies, which represent what a good engineer might choose using their intuition or trial-and-error based methods. In all examples, using variable boundary conditions produced large improvements over designs obtained using fixed boundary conditions. In the bistable morphing airfoil problem, the fixed boundary condition design failed outright to achieve the design goal, while the variable boundary condition design achieved bistability with a relatively small change to the actuator and support positions. These results highlight the difficulty in relying on human intuition for selecting effective boundary conditions and underscore the need for automated methods like the one developed in this work.

Despite the positive results showing the effectiveness of variable boundary conditions, the framework nevertheless has several limitations that should be addressed in future research. The use of a pure displacement-control algorithm prevented us from effectively solving bistable structural optimizations where snap-back behaviors occurred. A different algorithm that is able to trace load–displacement paths through limit points should be developed or extended to be compatible with a variable actuator position. Another issue is that in some cases the number of iterations to convergence can be computationally limiting, such as for the morphing wing of section 6.3.2 where the optimization completed after 1660 iterations. A worthwhile task for future research would be to find effective ways of reducing the required number of iterations. The combination of position and density design variables in a single optimization formulation may contribute to slow convergence since their sensitivities can span many orders of magnitude. However, the exact cause requires further investigation. There are also issues associated with nonlinear topology optimization in general. The divergence of unstable low-density elements in the Newton–Raphson iterations is not solved completely by the linear-nonlinear interpolation method [44], which limits the size of displacements and forces that can be applied. We also experienced oscillations in the convergence of optimizations where buckling occurs in the structure. A robust method for avoiding these buckling oscillations would be very useful for nonlinear topology optimization methods in general.

The possibilities for new research directions using variable boundary conditions in topology optimization are numerous. In compliant mechanism design, more complex and counter-intuitive mechanisms such as those with multiple degrees of freedom [59, 60] would likely see similar, or potentially greater, benefits than we demonstrated here. Koppen [61] describes these problems as relatively complex, even in some minimum working examples, due to crossing load paths that make different degrees of freedom inherently coupled. It is also discussed by Koppen [61] that these problems are highly constrained, significantly restricting the space of possible optimized designs and the possible design changes that can happen during optimizations. We hypothesize that if the boundary conditions were included as design variables, the expanded design space would free the optimizer to take additional routes on the way to the same, or likely better, optimized designs. Different types of movable mechanical boundary conditions, such as roller supports, have not been attempted yet, which could be applied to design mechanisms with sliding joints. Alternative methods of determining the optimal layouts of the boundary conditions, such as by using free-form design methods instead of the feature-mapping method used here, could be investigated as well. Greater control over positional design variables could be implemented, such as by constraining points to remain a certain distance away from the edges of non-rectangular design domains. Lastly, optimizing variable boundary conditions for each physics discipline in multiphysics problems holds much potential, for example, in the design of electrothermal actuators [6, 7]. If optimal placements of all six types of boundary conditions including elastic supports, applied

forces, applied temperatures, prescribed heat fluxes, applied voltages, and prescribed currents were all simultaneously optimized along with material distribution, it would greatly expand the design space and much better optimal designs may be discovered.

Acknowledgments

This research was funded by the National Science Foundation through Grant No. 1752045.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data and code used in this study is available from the corresponding author upon reasonable request.

References

1. M. P. Bendsoe and O. Sigmund, *Topology Optimization: Theory, Methods, and Applications* (Berlin, Germany: Springer Science and Business Media, 2013).

2. M. P. Bendsøe and N. Kikuchi, "Generating Optimal Topologies in Structural Design Using a Homogenization Method," *Computer Methods in Applied Mechanics and Engineering* 71, no. 2 (1988): 197–224.

3. G. K. Ananthasuresh, S. Kota, and Y. Gianchandani, "A Methodical Approach to the Design of Compliant Micromechanisms," in *Solid-State Sensor and Actuator Workshop* (SC: IEEE, 1994), 189–192.

4. O. Sigmund, "On the Design of Compliant Mechanisms Using Topology Optimization," *Mechanics of Structures and Machines* 25, no. 4 (1997): 493–524, https://doi.org/10.1080/08905459708945415.

5. P. D. L. Jensen, F. Wang, I. Dimino, and O. Sigmund, "Topology Optimization of Large-Scale 3D Morphing Wing Structures," in *Actuators* (Basel, Switzerland: Multidisciplinary Digital Publishing Institute, 2021), 217.

6. O. Sigmund, "Design of Multiphysics Actuators Using Topology Optimization–Part I: One-Material Structures," *Computer Methods in Applied Mechanics and Engineering* 190, no. 49–50 (2001): 6577–6604.

7. O. Sigmund, "Design of Multiphysics Actuators Using Topology Optimization – Part II: Two-Material Structures," *Computer Methods in Applied Mechanics and Engineering* 190, no. 49–50 (2001): 6605–6627, https://doi.org/10.1016/S0045-7825(01)00252-3.

8. Y. Moritoki, T. Furukawa, J. Sun, et al., "3D-Printed Micro-Tweezers With a Compliant Mechanism Designed Using Topology Optimization," *Micromachines (Basel)* 12, no. 5 (2021): 579.

9. O. Sardan, V. Eichhorn, D. H. Petersen, S. Fatikow, O. Sigmund, and P. Bøggild, "Rapid Prototyping of Nanotube-Based Devices Using Topology-Optimized Microgrippers," *Nanotechnology* 19, no. 49 (2008): 495503.

10. S. Koppen, M. Langelaar, and F. van Keulen, "A Simple and Versatile Topology Optimization Formulation for Flexure Synthesis," *Mechanism and Machine Theory* 172 (2022): 104743.

11. M. Wallin, A. Dalklint, and D. Tortorelli, "Topology Optimization of Bistable Elastic Structures—An Application to Logic Gates," *Computer Methods in Applied Mechanics and Engineering* 383 (2021): 113912.

12. Q. Chen, X. Zhang, H. Zhang, B. Zhu, and B. Chen, "Topology Optimization of Bistable Mechanisms With Maximized Differences Between Switching Forces in Forward and Backward Direction," *Mechanism and Machine Theory* 139 (2019): 131–143.

13. B. Zhu, X. Zhang, H. Zhang, et al., "Design of Compliant Mechanisms Using Continuum Topology Optimization: A Review," *Mechanism and Machine Theory* 143 (2020): 103622, https://doi.org/10.1016/j.mechmachtheory.2019.103622.

14. L. Rakotondrainibe, G. Allaire, and P. Orval, "Topology Optimization of Connections in Mechanical Systems," *Structural and Multidisciplinary Optimization* 61 (2020): 1–17.

15. L. Rakotondrainibe, J. Desai, P. Orval, and G. Allaire, "Coupled Topology Optimization of Structure and Connections for Bolted Mechanical Systems," *European Journal of Mechanics - A/Solids* 93 (2022): 104499.

16. Q. Xia, M. Y. Wang, and T. Shi, "A Level Set Method for Shape and Topology Optimization of Both Structure and Support of Continuum Structures," *Computer Methods in Applied Mechanics and Engineering* 272 (2014): 340–353.

17. T.-U. Lee and Y. M. Xie, "Simultaneously Optimizing Supports and Topology in Structural Design," *Finite Elements in Analysis and Design* 197 (2021): 103633.

18. J. H. Zhu and W. H. Zhang, "Integrated Layout Design of Supports and Structures," *Computer Methods in Applied Mechanics and Engineering* 199, no. 9–12 (2010): 557–569.

19. T. Buhl, "Simultaneous Topology Optimization of Structure and Supports," *Structural and Multidisciplinary Optimization* 23, no. 5 (2002): 336–346, https://doi.org/10.1007/s00158-002-0194-2.

20. Q. Xia and T. Shi, "Topology Optimization of Compliant Mechanism and Its Support Through a Level Set Method," *Computer Methods in Applied Mechanics and Engineering* 305 (2016): 359–375.

21. Q. Xia, L. Xia, and T. Shi, "Topology Optimization of Thermal Actuator and Its Support Using the Level Set Based Multiple–Type Boundary Method and Sensitivity Analysis Based on Constrained Variational Principle," *Structural and Multidisciplinary Optimization* 57 (2018): 1317–1327.

22. L. Alacoque and K. A. James, "Topology Optimization With Variable Loads and Supports Using a Super-Gaussian Projection Function," *Structural and Multidisciplinary Optimization* 65 (2022): 50, https://doi.org/10.1007/s00158-021-03128-2.

23. T.-U. Lee and Y. M. Xie, "Optimizing Load Locations and Directions in Structural Design," *Finite Elements in Analysis and Design* 209 (2022): 103811.

24. T.-U. Lee and Y. M. Xie, "Finding Globally Optimal Arrangements of Multiple Point Loads in Structural Design Using a Single FEA," *Structural and Multidisciplinary Optimization* 66, no. 4 (2023): 1–16.

25. T. Buhl, C. B. W. Pedersen, and O. Sigmund, "Stiffness Design of Geometrically Nonlinear Structures Using Topology Optimization," *Structural and Multidisciplinary Optimization* 19 (2000): 93–104.

26. T. E. Bruns and D. A. Tortorelli, "Topology Optimization of Non-linear Elastic Structures and Compliant Mechanisms," *Computer Methods in Applied Mechanics and Engineering* 190 (2001): 3443–3459, https://doi.org/10.1016/S0045-7825(00)00278-4.

27. M. Wallin, N. Ivarsson, and D. Tortorelli, "Stiffness Optimization of Non-Linear Elastic Structures," *Computer Methods in Applied Mechanics and Engineering* 330 (2018): 292–307.

28. K. A. James and H. Waisman, "Layout Design of a Bi-Stable Cardiovascular Stent Using Topology Optimization," *Computer Methods in Applied Mechanics and Engineering* 305 (2016): 869–890.

29. A. Bhattacharyya, M. Bashkawi, S. Y. Kim, W. Zheng, T. Saxton-Fox, and K. A. James, "Computational Design and Experimental Testing of a Flexible Bi-Stable Airfoil for Passive Flow Control," in *AIAA Aviation 2021 Forum* (2021), 3087.

30. A. Bhattacharyya, C. Conlan-Smith, and K. A. James, "Design of a Bi-Stable Airfoil With Tailored Snap-Through Response Using Topology Optimization," *CAD Computer Aided Design* 108 (2019): 42–55, https://doi.org/10.1016/j.cad.2018.11.001.

31. W. Li, F. Wang, O. Sigmund, and X. S. Zhang, "Digital Synthesis of Free-Form Multimaterial Structures for Realization of Arbitrary Programmed Mechanical Responses," *Proceedings of the National Academy of Sciences* 119, no. 10 (2022): e2120563119.

32. W. Li, F. Wang, O. Sigmund, and X. S. Zhang, "Design of Composite Structures With Programmable Elastic Responses Under Finite Deformations," *Journal of the Mechanics and Physics of Solids* 151 (2021): 104356.

33. C. B. W. Pedersen, T. Buhl, and O. Sigmund, "Topology Synthesis of Large-Displacement Compliant Mechanisms," *International Journal for Numerical Methods in Engineering* 50, no. 12 (2001): 2683–2705.

34. J. Reinisch, E. Wehrle, and J. Achleitner, "Multiresolution Topology Optimization of Large-Deformation Path-Generation Compliant Mechanisms With Stress Constraints," *Applied Sciences* 11, no. 6 (2021): 2479.

35. K. Svanberg, "The Method of Moving Asymptotes—A New Method for Structural Optimization," *International Journal for Numerical Methods in Engineering* 24, no. 2 (1987): 359–373.

36. N. Pollini and O. Amir, "Mixed Projection- and Density-Based Topology Optimization With Applications to Structural Assemblies," *Structural and Multidisciplinary Optimization* 61 (2020): 687–710, https://doi.org/10.1007/s00158-019-02390-9.

37. F. Wein, P. D. Dunning, and J. A. Norato, "A Review on Feature-Mapping Methods for Structural Optimization," *Structural and Multidisciplinary Optimization* 62 (2020): 1597–1638.

38. N.-H. Kim, *Introduction to Nonlinear Finite Element Analysis* (Berlin, Germany: Springer Science and Business Media, 2014).

39. T. Belytschko, W. K. Liu, B. Moran, and K. Elkhodary, *Nonlinear Finite Elements for Continua and Structures* (Hoboken, New Jersey: John Wiley and Sons, 2014).

40. J. N. Reddy, *An Introduction to Nonlinear Finite Element Analysis Second Edition: With Applications to Heat Transfer, Fluid Mechanics, and Solid Mechanics* (OUP Oxford, 2014).

41. S. S. Rao, *The Finite Element Method in Engineering* (Oxford, UK: Butterworth-Heinemann, 2017).

42. T. E. Bruns and D. A. Tortorelli, "An Element Removal and Reintroduction Strategy for the Topology Optimization of Structures and Compliant Mechanisms," *International Journal for Numerical Methods in Engineering* 57, no. 10 (2003): 1413–1430.

43. G. H. Yoon and Y. Y. Kim, "Element Connectivity Parameterization for Topology Optimization of Geometrically Nonlinear Structures," *International Journal of Solids and Structures* 42, no. 7 (2005): 1983–2009.

44. F. Wang, B. S. Lazarov, O. Sigmund, and J. S. Jensen, "Interpolation Scheme for Fictitious Domain Techniques and Topology Optimization of Finite Strain Elastic Problems," *Computer Methods in Applied Mechanics and Engineering* 276 (2014): 453–472.

45. Y. Luo, M. Y. Wang, and Z. Kang, "Topology Optimization of Geometrically Nonlinear Structures Based on an Additive Hyperelasticity Technique," *Computer Methods in Applied Mechanics and Engineering* 286 (2015): 422–441.

46. L. Liu, J. Xing, Q. Yang, and Y. Luo, "Design of Large-Displacement Compliant Mechanisms by Topology Optimization Incorporating Modified Additive Hyperelasticity Technique," *Mathematical Problems in Engineering* 2017 (2017): 4679746.

47. G. L. Bluhm, O. Sigmund, and K. Poulios, "Internal Contact Modeling for Finite Strain Topology Optimization," *Computational Mechanics* 67, no. 4 (2021): 1099–1114.

48. A. Klarbring and N. Strömberg, "Topology Optimization of Hyperelastic Bodies Including Non-Zero Prescribed Displacements," *Structural and Multidisciplinary Optimization* 47 (2013): 37–48.

49. A. Dalklint, M. Wallin, and D. A. Tortorelli, "Structural Stability and Artificial Buckling Modes in Topology Optimization," *Structural and Multidisciplinary Optimization* 64, no. 4 (2021): 1751–1763.

50. Q. Chen, Q. Wen, X. Zhang, Y. Yang, and S. Xiao, "Buckling-Induced Instability in Topology Optimization of Compliant Constant-Force Mechanisms," *Mechanism and Machine Theory* 191 (2024): 105475.

51. M. J. Clarke and G. J. Hancock, "A Study of Incremental-Iterative Strategies for Non-linear Analyses," *International Journal for Numerical Methods in Engineering* 29, no. 7 (1990): 1365–1391.

52. J. Batoz and G. Dhatt, "Incremental Displacement Algorithms for Nonlinear Problems," *International Journal for Numerical Methods in Engineering* 14, no. 8 (1979): 1262–1267.

53. T. E. Bruns, O. Sigmund, and D. A. Tortorelli, "Numerical Methods for the Topology Optimization of Structures That Exhibit Snap-Through," *International Journal for Numerical Methods in Engineering* 55, no. 10 (2002): 1215–1237.

54. T. E. Bruns and O. Sigmund, "Toward the Topology Design of Mechanisms That Exhibit Snap-Through Behavior," *Computer Methods in Applied Mechanics and Engineering* 193, no. 36–38 (2004): 3973–4000.

55. K. Svanberg, "MMA and GCMMA-Two Methods for Nonlinear Optimization," (2007), 1–15, https://people.kth.se/~krille/mmagcmma.pdf.

56. F. Ferrari and O. Sigmund, "A New Generation 99 Line Matlab Code for Compliance Topology Optimization and Its Extension to 3D," *Structural and Multidisciplinary Optimization* 62, no. 4 (2020): 2211–2228.

57. G. A. da Silva, A. T. Beck, and O. Sigmund, "Topology Optimization of Compliant Mechanisms With Stress Constraints and Manufacturing Error Robustness," *Computer Methods in Applied Mechanics and Engineering* 354 (2019): 397–421.

58. G. A. da Silva, A. T. Beck, and O. Sigmund, "Topology Optimization of Compliant Mechanisms Considering Stress Constraints, Manufacturing Uncertainty and Geometric Nonlinearity," *Computer Methods in Applied Mechanics and Engineering* 365 (2020): 112972.

59. C. Alonso, R. Ansola, and O. M. Querin, "Topology Synthesis of Multi-Input–Multi-Output Compliant Mechanisms," *Advances in Engineering Software* 76 (2014): 125–132.

60. B. Zhu, Q. Chen, M. Jin, and X. Zhang, "Design of Fully Decoupled Compliant Mechanisms With Multiple Degrees of Freedom Using Topology Optimization," *Mechanism and Machine Theory* 126 (2018): 413–428.

61. S. Koppen, "Topology Optimization of Compliant Mechanisms With Multiple Degrees of Freedom," (2022).