Model-Based Design of a Soft 3-D Haptic Shape Display

Margaret Koehler, *Student Member, IEEE*, Nathan S. Usevitch, *Student Member, IEEE*, and Allison M. Okamura, *Fellow, IEEE*

***Abstract*—Haptic shape displays provide compelling touch interactions by allowing users to freely explore a rendered surface. However, these displays are currently limited to 2.5-D surfaces due to the space requirements of their actuation. Building on previous work in haptic jamming, we developed a novel, soft 3-D shape display. A fully 3-D display that a user can grasp and hold allows for improved interactions for applications such as medical palpation training and virtual reality experiences. The shape display is implemented as an inflatable silicone membrane with embedded particle jamming cells that change stiffness and soft pneumatic actuators that control the distance between points on the surface. The device was modeled as a mass-spring system, and this model is used to develop a control sequence for a device to match a target shape. Due to constraints in actuation imposed by the 3-D geometry of the device, we developed an automatic design algorithm for the display, so that a display can be custom-designed to reach a set of target shapes using a relatively small number of actuators.**

***Index Terms*—Haptics and haptic interfaces, modeling, control, and learning for soft robots, simulation and animation, soft robots materials and design.**

I. Introduction

HAPTIC shape displays provide touch feedback to a user by changing shape to render different environments. In contrast to typical kinesthetic haptic devices in which the user interacts with a virtual model through a held stylus, a shape display allows for multipoint interactions and can provide simultaneous kinesthetic and cutaneous feedback. A haptic shape display is an encountered-type haptic device with which a user can actively and directly explore an environment, rather than being constrained to interactions through a tool [1]. By allowing for these complex interactions, shape displays can provide a rich sense of realism and convey haptic information related to the shape of the surface itself and the variable stiffness along that surface.

In this article, we present a fully three-dimensional (3-D) shape display, in contrast to most shape displays, which are 2.5-D (able to render surfaces that can be represented by a planar contour map) [2]. One application for such a shape-changing haptic device is in medical training. Palpation and haptic feedback are important tools for medical evaluation and diagnosis. A shape changing device that mimics an organ could be incorporated into a medical simulation mannequin to model different shape or stiffness changes in the organ. This rapid and automated change of the properties of model organs would be an improvement over current static mannequins that can only render a single patient state. Being fully 3-D would allow trainees to grasp a model organ or manipulate it with two hands, rather than just pressing a surface from the top. Additionally, a haptic shape display could be used in virtual and augmented reality applications, such as product design, online shopping, or gaming, to allow a user to reach out to a virtual object and grab the shape display that approximates the shape of the virtual object. The shape display could act as a form of digital clay, rapidly providing physical feedback on the design of an object or recreating the physical form of an object for assessment prior to a purchase [3].

Building on previous work in haptic jamming, which used particle jamming cells and a silicone membrane in a 2.5-D display to create large deformation of a continuous surface [4], we developed a fully 3-D design, shown in Fig. 1. In this design, we embed particle jamming cells in a silicone membrane to form a shell around an internal pressure cavity. Soft pneumatic actuators inside that cavity control the distance between two surface points. We do not use the particle jamming cells to explicitly control the stiffness of the device, instead using them as part of the overall shape control. While the stiffness is not controlled independently from the shape, the soft components produce an overall device stiffness in the appropriate range for our target application of model organs for palpation training.

Most 2.5-D shape displays depend on a large bank of actuators behind the surface [4]–[7]. In a fully 3-D device, the space behind the surface is limited to the interior of the device, leaving very little space in which to place the actuators. This permits the use of only a small number of actuators, which necessarily limits the versatility of a particular device. To compensate for the small number of actuators, we developed an automatic model-based design algorithm to design custom devices that can reach small sets of target shapes with relatively few actuators. This automatic design algorithm takes advantage of the flexibility of modern manufacturing techniques that can easily produce components

Manuscript received December 26, 2019; accepted February 8, 2020. Date of publication April 7, 2020; date of current version June 4, 2020. This paper was recommended for publication by Associate Editor G. Gu and Editor M. Yim upon evaluation of the reviewers' comments. This work was supported in part by the U.S. Army Medical Research and Materiel Command under Grant W81XWH-15-C-0091, in part by the National Science Foundation under Grant 1830163, and in part by the National Science Foundation Graduate Research Fellowship Program. *(Corresponding author: Margaret Koehler.)*

The authors are with the Department of Mechanical Engineering, Stanford University, Stanford, CA 94305 USA (e-mail: mkoehler@stanford.edu; usevitch@stanford.edu; aokamura@stanford.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2020.2980114

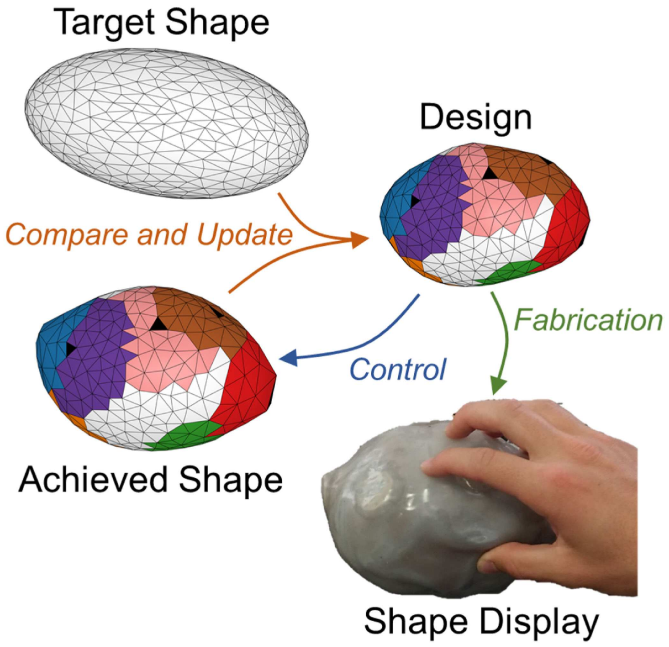


Fig. 1. Iterative design algorithm takes a small set of target shapes as inputs and provides a design (rest shape and actuator configuration) and control sequence as outputs. The algorithm uses a mass-spring model to predict the achieved shape for a given design and control sequence. The design can, then, be realized in hardware, and the shape display actuated according to the control sequence.

of arbitrary shape. As with most soft robotic devices, the use of soft components makes the device complex to model and control. In particular, the particle jamming cells lead to path-dependent behavior in which the final shape is dependent on the whole control sequence, not a single equilibrium result [8]. The automatic design algorithm takes this path-dependency into account. Our approach is outlined in Fig. 1.

This article presents two main contributions. First, we developed the design, modeling, and control of a fully 3-D haptic shape display using actuation techniques from soft robotics and modeling techniques from physics-based computer simulation. Second, we created an iterative automatic design algorithm that designs a shape display to reach a small set of predetermined target shapes with a small number of actuators. This design algorithm works by iteratively controlling the device to match each target shape and adding a corrective actuator where the error in shape matching is the greatest.

II. Background

This article builds on previous research related to two areas: haptic shape displays and computational modeling and design.

A. *Shape Displays and Haptic Interfaces*

Shape displays have been the focus of much work in the fields of haptics and human–computer interaction. Most of these displays are 2.5-D devices consisting of a planar array of actuators moving into a third dimension. Pin arrays are a common paradigm of shape display design across scales [9], [10], and they simulate a continuous surface by either densely packing many pins [5], [11] or by mounting a flexible membrane over the pins [6], [12], [13]. Pin arrays are easy to control to match a particular surface, but require a large number of actuators in a predetermined layout. Some pin arrays include force control of the pins for variable stiffness rendering [7].

Other shape displays have used shape memory alloys [14], [15] or rigid components arranged in a discrete mesh also described as a "deformable crust" [16], [17]. Displays based on pneumatics have also been explored, including flexible tactile arrays [18], [19]. Yao *et al.* [20] described a variety of pneumatically actuated user interface devices that can change shape. A single pneumatic actuator changes the shape and stiffness of a deformable holdable haptic device in [21]. Pneumatics have also been used to create variable size haptic displays, as in [22]. The digital clay design described in [23] uses many balloons to act as a volumetric digital clay interface.

Our device design is based on previous work in haptic jamming displays that incorporate particle jamming cells in a surface to enhance shape control and provide variable surface stiffness [24], [25]. Haptic jamming displays have been primarily focused on 2.5-D displays in which a flat sheet of cells is inflated from below. This work has included analysis of the mechanical properties of a jammed and unjammed shape display [24], as well as computational methods for modeling and control of the device [4], [8]. Stanley and Okamura [8] showed that a large number of actuators were required to achieve good general matching of target height maps and that surface matching was sensitive to individual actuator placement particularly when few actuators were used. This article also indicated that particle jamming cells are critical for controlling the shape, not just in determining mechanical properties.

Various haptics applications for particle jamming are described in [26], including a volumetric particle jamming cell that can maintain a deformed shape that a user inputs. Other work that uses particle jamming in 3-D includes the universal jamming gripper [27], which does not have active shape control but instead conforms passively to its environment, and the jamming skin soft robot, which uses selective jamming during pressurization to locomote and change shape [28]. The display described in [29] combines a pressurization chamber and particle jamming with an excess of a flexible cloth surface to allow the user to sculpt 3-D shapes, and the haptic jamming sphere presented in [30] can change size and stiffness but not shape. Particle jamming has also been proposed for organ design for palpation training [31]. These 3-D jamming surfaces, unlike the 2.5-D haptic jamming displays and unlike our device, do not include actuation other than the jamming cells and internal pressurization, limiting the potential for active shape change. The pressurized balloon in [32] can similarly change size and stiffness but not shape, although the authors do propose an array of such balloons that could change overall shape.

B. *Modeling and Computational Design*

To design a shape display capable of reaching a particular set of target shapes, we must be able to predict how the display will deform under actuation and use this prediction to determine a good rest shape and placement of the actuators. Since our device is soft, we can look to the field of soft robotics. Simulation and design automation tools have been recognized as two of the main challenges in soft robotics [33]. Due to the complex mechanical properties of soft robots, intuition can fail to provide a good design that deforms as desired. Design automation has been used in some soft devices and robots. In [34], the rest shape required to achieve a final balloon shape for a particular inflation pressure was determined using a constrained optimization and an augmented Lagrangian method. Using similar techniques, deformable shapes were made to match particular poses by optimizing the location and direction of applied forces in conjunction with material distribution in [35]. Gradient-based methods were also used in [36] to optimize shape matching from a given rest shape to target shapes based on inflation. In that work, Ma *et al.* [36] optimized the distribution of material in a frame embedded in the surface of a deformable object. Different air chambers were designed in order to allow a single rest shape to match more than one target shape. In contrast to our device, the device in [36] focuses primarily on surface-based actuation rather than linear actuation that is internal to the device. Genetic algorithms have seen widespread success in design optimization [37]. For soft robots in particular, genetic algorithms have been used to design a robot that locomotes based on a predetermined actuation sequence with multiple material elements [38]. Baykal *et al.* [39] combined a stochastic global optimization algorithm with a sampling-based motion planner to determine the kinematic parameters of a deformable robot with the goal of reaching particular target poses. The high dimensionality of robot design, particularly for deformable robots, makes these optimization problems challenging.

An integral component of all of these design algorithms is the underlying model of the deformable body. Two main approaches are used for deformable modeling [40]. In a finite-element model (FEM) of an elastic surface, the surface is discretized into smaller triangular surfaces and the continuum equations are solved on this smaller surface, with compatability constraints enforced between triangular elements. This can be used, for instance, to approximate a nonplanar surface with many smaller planar surfaces. In a mass-spring model, the surface is again discretized into triangles, but the deformation of the membrane is approximated using elastic elements along the edges of the triangles. Thus, the elastic elements of a FEM of a surface are continuous, 2-D surfaces, while the elastic elements of a mass-spring model of a surface are 1-D springs. Finite element modeling can provide more accurate results, particularly for equilibrium solutions [34], [35] and have been used for the dynamic control of soft robots [41]. Mass-spring models are easier to implement and allow for faster simulation [40]. A mass-spring model was used in [8] to represent a haptic jamming display and analyze the presented control strategies, and it is this approach that we extend in our work.

III. Design and Modeling of Shape Display: System and Components

In this section, we describe the physical components used in the shape display as well as the mass-spring model that was developed for the system. The overall design and hardware components of the shape display are shown in Fig. 2. The main structure of the shape display consists of two nested silicone membranes. The region between the two membranes is filled with coffee grounds and partitioned into independent particle jamming cells. Points on the surface of the inner membrane are connected with soft linear actuators that traverse the inner cavity of the device. The actuators are used to control the distance between their attachment points. The user interacts with the outer silicone membrane. We use a mass-spring model, rather than a FEM, because it allows for fast dynamic simulation and has had sufficient accuracy for previous haptic jamming surfaces [8]. This model is used as a component of the design algorithm that follows. Additionally, the model can be used to explore the range of achievable shapes for a given shape display design.

A. Inflatable Membrane

The rest shape of the shape display is determined by two nested silicone membranes. These silicone membranes are modeled as a triangle mesh of point masses and springs, as seen in Fig. 3. Unlike previous models of haptic jamming surfaces that depend on a specific rectangular geometry [8], this model admits an arbitrary triangle mesh representing the surface. Using the derivation in [40], the spring stiffness for an edge is defined as

$$k_e = \frac{EA(T_e)}{|c|^2} \tag{1}$$

where k_e is the spring stiffness, E is the elastic modulus of the membrane material, $A(T_e)$ is the sum of the initial areas of the two triangles adjacent to the edge, and c is the initial length of the spring. In parallel with each spring is a linear damper. The simulation controls the amount of air inside the main cavity of the shape display, and the corresponding pressure is calculated based on the volume enclosed by the surface, assuming constant temperature. The resulting pressure from that air is, then, applied to each triangle, with the force equally distributed between the three vertices of each triangle. As in [34], the pressure force applied to each vertex i in the mesh is

$$\vec{F}_p^i = \sum_j \frac{pA_j}{3}\hat{n}_j \tag{2}$$

in which p is the gauge pressure in the chamber, A_j is the area of triangle j that has vertex i as a vertex, and $\vec{n}_j$ is the outward normal of triangle j.

The system is simulated using explicit fourth-order Runge–Kutta integration with a time step of 0.001 s. In addition to the linear damper aligned with each spring, a viscous damping force, proportional and opposite to the velocity of each point mass, is applied to each particle in order to reduce oscillations and encourage stability.

B. Inverse Pneumatic Artificial Muscles (IPAMs)

Our system requires actuators composed of soft materials that are capable of high extension ratios. We use IPAMs, modified from [42], to connect points on the surface and control the distance between them. The IPAMs consist of latex rubber tubes enclosed in a sheath of inextensible fabric. The diameter of the

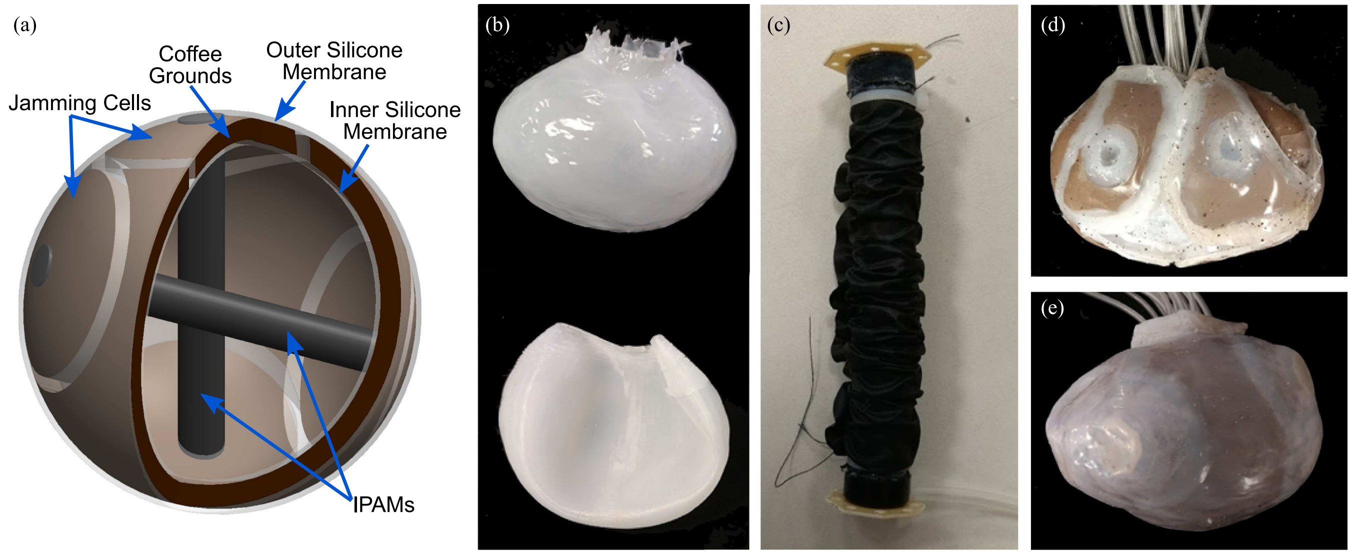
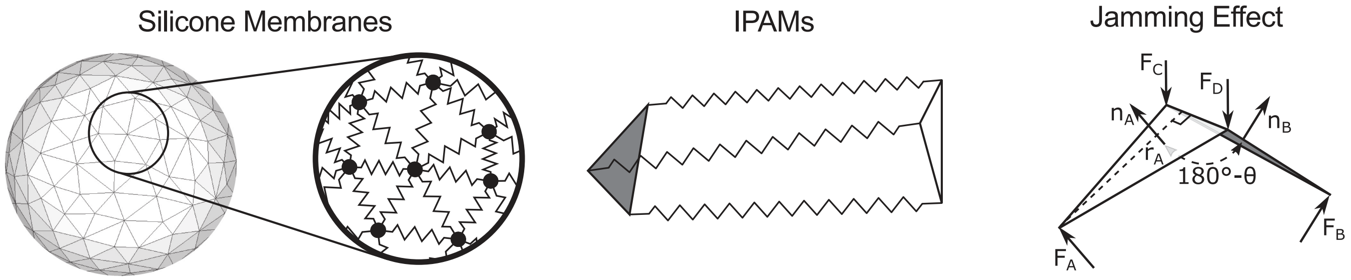


Fig. 2. General shape display device design and its hardware components. (a) Schematic of the general device design, showing how the components are arranged. The spherical rest shape and small number of actuators are shown for clarity. (b) Inner and outer silicone membranes, made by brush molding silicone over a 3-D printed mold, define the rest shape. (c) IPAM consists of a rubber tube encased in a fabric sheath. The fabric is longer than the tube, and unwrinkling of the fabric allows the tube to expand axially but not radially. (d) Jamming cells, consisting of separated regions of coffee grounds, are attached to the inner silicone membrane. Each jamming cell has a separate vacuum line. (e) IPAMs are attached and the outer silicone membrane is stretched over the shape to complete the assembly process. The tubes for the main chamber, IPAMs, and jamming cells are connected externally to their control valves (not shown).



(a) (d) (c)

Fig. 3. Each component of the model is shown. (a) Silicone surface is represented by a triangle mesh and modeled as a mass-spring system with springs along each edge and a particle at each vertex. (b) Each IPAM is modeled as a set of three springs that connects two triangle faces. (c) Dihedral springs along each edge are used to account for the bending stiffness of the surface, which is crucial for modeling the particle jamming effect.

fabric tube is the same as the outer diameter of the rubber tube. The fabric tube is longer than the rubber tube, and each end of the fabric tube is secured to the end of the rubber tube. The fabric is wrinkled so that the full length of the fabric fits on the unstretched rubber tube. As in the case of the fiber-wrapped IPAMs, the fabric sheath stops the rubber tube from expanding radially, forcing it to expand axially when pressurized. IPAMs act as linear springs with variable natural length based on their pressure, and they permit large strain, which is essential for reaching a wide variety of shapes. Unlike typical linear actuators, IPAMs are flexible and can bend around each other, which allows us to avoid problems of actuator intersection. IPAMs have previously been used primarily in tension, but can also apply forces while under compression. While buckling and bowing are concerns when in compression, we have found that the IPAMs can provide substantial pushing force, in part because they are in contact with the other IPAMs in the interior of the device.

An IPAM can be considered as a spring with a controllable natural length. From [42], the axial force produced by an IPAM, F_{IPAM}, is

$$F_{\text{IPAM}} = k(L - L_0) - AP \tag{3}$$

where k is the axial stiffness of the rubber tube, L is the length, L_0 is the free length (assuming no pre-stretch), A is the cross-sectional area, and P is the gauge pressure of the IPAM. The force–length relationship can be written using a natural length L_n as follows:

$$F_{\text{IPAM}} = k(L - L_{\text{n}}) \tag{4}$$

$$L_{\text{n}} = L_0 + \left(\frac{A}{k}\right) P. \tag{5}$$

The tube stiffness, free length, and cross-sectional area are constant. From this equation, it is evident that pressure acts as a linear controller of the natural length of the IPAM. The spring stiffness is dependent on the dimensions and material properties of the rubber tube, and the constant relating pressure to natural length is dependent on this stiffness and the cross-sectional area of the tube.

The IPAMs are included in the mass-spring system according to this linear spring model. To accurately capture the distribution of forces on the surface at the IPAM attachment points [see the IPAM ends in Fig. 2(c) and (e)], each end of the IPAM is attached to a triangle in the mass-spring system, rather than a single vertex. The IPAM is, then, modeled as a set of three identical springs connecting the two triangle ends, each spring connecting a pair of vertices, one from each triangle [see Fig. 3(b)]. The stiffness of each of these springs is one-third of that of the IPAM itself, so that the three springs acting in parallel match the behavior of the IPAM. The physical IPAMs have octagonal endcaps, which are used to attach them to the silicone membrane. We do not explicitly model this attachment shape, instead use a single triangle from the mesh as a simplification. This assumes that the size of the triangle in the mesh is similar to the size of the endcap. Furthermore, this means that we do not design for features smaller than the size of these endcaps or the mesh triangles. As with the surface springs, a linear damper is included in parallel with each of the IPAM springs. The pressure inside each IPAM is controlled.

C. Particle Jamming Cells

The space between the two silicone membranes is filled with dry coffee grounds. This coffee ground layer is partitioned into different airtight regions, each of which is a particle jamming cell. Each jamming cell is connected through a valve to a vacuum line. Applying the vacuum to a cell (jamming the cell) causes the silicone layers to pull together, increasing the friction force between the coffee grounds and causing the jamming cell to stiffen. This is based on the principle of particle jamming, and the properties of this silicone and coffee ground design have been characterized in [24].

To model the bending stiffness, torsional springs were added to each edge that apply a torque proportional to the difference in dihedral angle relative to an initial dihedral angle, as used in [43]. The dihedral angle of an edge, θ, is the angle between the normal vectors of the two triangles containing that edge. The torsional spring applies forces on four points, the two points on the edge containing the torsional spring and the two other vertices of the triangles that contain that edge as shown in Fig. 3(c). The torque and forces are computed as follows:

$$\tau = -k_t(\theta - \theta_n) \tag{6}$$

$$\vec{F}_A = \frac{\tau}{\|\vec{r}_A\|}\hat{n}_A \tag{7}$$

$$\vec{F}_B = \frac{\tau}{\|\vec{r}_B\|}\hat{n}_B \tag{8}$$

$$\vec{F}_C = \vec{F}_D = -\frac{1}{2}\left(\vec{F}_A + \vec{F}_B\right) \tag{9}$$

where τ is the torque, k_t is the torsional spring constant, θ_n is the natural angle of the spring, $\vec{F}$ are forces on the different vertices of the triangles that contain the edge, and $\|\vec{r}\|$ is the distance from the vertex to the edge. The dihedral spring stiffness is set to the same value for all of the edges, not taking into account the effect of edge length on bending stiffness.

We use jamming as an ON–OFF actuator, which modulates the stiffness of the cell. When a cell is jammed, the increase in stiffness is simulated using the following changes in the model, similar to those used to model jamming in [8]. First, the natural lengths of the linear springs and the natural angles of the torsional springs for the edges of triangles within that cell are set to the lengths and angles of each spring in the current configuration. The linear spring stiffnesses are scaled up by a factor of 10, and the dihedral spring stiffnesses are scaled up by a factor of 30. When unjammed, the coffee grounds can slide past each other easily, but when the cell is jammed, the coffee grounds lock together to resist stretching and bending. The increase in bending stiffness is due to the fact that the jamming cells have substantial thickness. While the silicone maintains it original properties, the jamming effect of the coffee grounds has the predominant stiffness effect in both linear and torsional spring stiffness when jammed.

Since the modeled properties of the jamming cell, in particular, the dihedral and linear spring natural angles and lengths, are dependent on the shape of the device when the cell is jammed, a full history of the actuation of the device is needed. Thus, a static equilibrium model is not sufficient to determine the final shape of the shape display. This requires a forward dynamic model in the simulation and also limits the possibilities for control and design optimization, leading us to an iterative method based on a heuristic controller and design updates.

IV. Control to Target Shape

Our approach is to develop a heuristic feedback control algorithm that transforms the shape display from its rest shape to a shape that approximates a given target shape. Using the model presented in the previous section, we first determine an actuation sequence that actuates a given design of a shape display to approximate the target shape. We use this controller repeatedly to evaluate different designs in simulation, and, then, finally use the controller to control a hardware version of the shape display. In this section, we first describe how we compare two meshes with different connectivity (a necessary component for the control and design algorithms) and, then, describe the control algorithm itself.

A. Shape Representation and Comparison

An important component of both the control and design algorithms is the representation of the shape and how that representation can be used to evaluate the error in shape matching in order to inform actuation and design updates. As described in the previous section, the shape display is modeled as a mass-spring system, represented by a triangle mesh. The target shapes are also represented as triangle meshes. In order to determine the error between a given mesh and a target mesh, we need some way to compare the two. Furthermore, in contrast to [35], our target meshes do not have the same connectivity as each other or

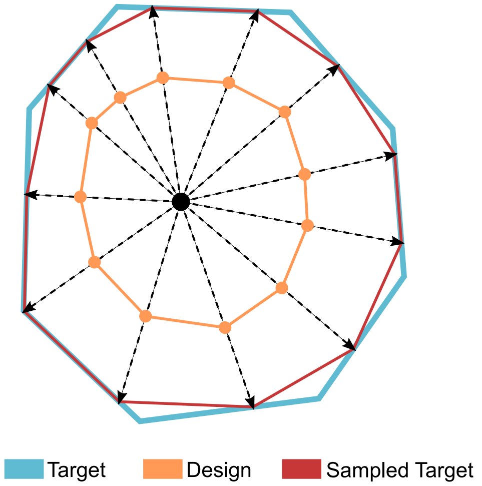


Fig. 4. 2-D example of the radial sampling technique. The target mesh is sampled radially based on the positions of the design mesh vertices in order to make a sampled target mesh. The sampled target mesh can, then, be easily compared to the design mesh, since their connectivity is the same. Sharp corners in the target shape can be missed in the sampled target.

as the design mesh, so simple position correspondences between vertices is not an option. Thus, for error computations we use a radial sampling technique similar to that described in [44] in which we build a sampled target mesh for each target shape that has the same connectivity as the design mesh. This sampled target mesh is updated at each control step and depends on the current state of the design mesh.

A 2-D version of the radial sampling technique is shown in Fig. 4. To make the sampled target mesh, each vertex in the design mesh is used to define a ray from the origin through that vertex. The intersection point between that ray and the target mesh becomes a vertex in the sampled target mesh. The vertices in the sampled target mesh are connected in the same way as the design mesh. This sampled target mesh allows for straightforward comparisons between the design and the target shapes that are used to inform the control and design.

This sampling technique is based on the idea that inflation of the main chamber can be approximated by radial scaling. This is true for a sphere, but not for more complex shapes. However, as the design mesh gets closer to the target mesh (as a result of the actuation), the correspondence between the target mesh and the sampled target mesh will improve. When the two meshes are very different in size or shape, the initial sampled target mesh may not correspond as well to the target mesh because radial expansion is not a perfect representation of inflation and since we specifically change the shape with actuators. Thus, an incremental control algorithm is required. Particularly, sharp edges or features may be lost when the vertices of the design mesh do not align with them. This could be accounted for by sampling both the design and target meshes with a set of sampling rays that are more highly concentrated on those particular areas of interest. However, such a system would have to maintain the physical correspondence between the mesh and the actuator placement as well. Finally, this radial sampling technique limits the shapes that we can use to those that are star-shaped or visible from the origin (shapes for which there is only one intersection point along any ray from the origin). It is important to note that this limitation is due to the particular means of shape comparison that we have chosen to implement; it is not inherent in the design of the display or the control and design algorithms.

We define a metric for postanalysis that gives a single value to capture the shape similarity between the achieved shape (the shape at the end of the control sequence) and the target shape. This metric is not used during the control or design phase because it does not give sufficient information on how to improve the shape matching through actuation or actuator placement. We use the sampled target mesh and the design mesh to compare the similarity of the two shapes using the Pearson Correlation Coefficient (PCC), a normalized cross-correlation commonly used in template matching in images [45] and previously used to compare shape display surfaces [8]. The PCC is computed as

$$P = \frac{\sum_i (r_i - \bar{r})(R_i - \bar{R})}{\sqrt{\sum_i (r_i - \bar{r})^2 \sum_i (R_i - \bar{R})^2}} \tag{10}$$

where r_i is the radius of a vertex in the design mesh, R_i is the radius of the corresponding vertex in the sampled target mesh, $\bar{r}$ is the average radius of the vertices in the design mesh, and $\bar{R}$ is the average radius of the vertices in the sampled target mesh. This metric compares the shape of the sampled target mesh and the design mesh by determining the strength of the correlation between the radii of the vertices in the two meshes. The PCC ranges from -1 to 1, with 1 corresponding to perfect positive correlation, and -1 corresponding to perfect negative correlation. Since this metric only accounts for correlation between the radii, the absolute size of the meshes is not accounted for, so comparing a pair of radially scaled meshes will result in a PCC of 1. With the assumption that the two shapes have the same volume, however, we can be assured that the PCC is a reasonable representation of the shape similarity. Our controller achieves this required volume matching well.

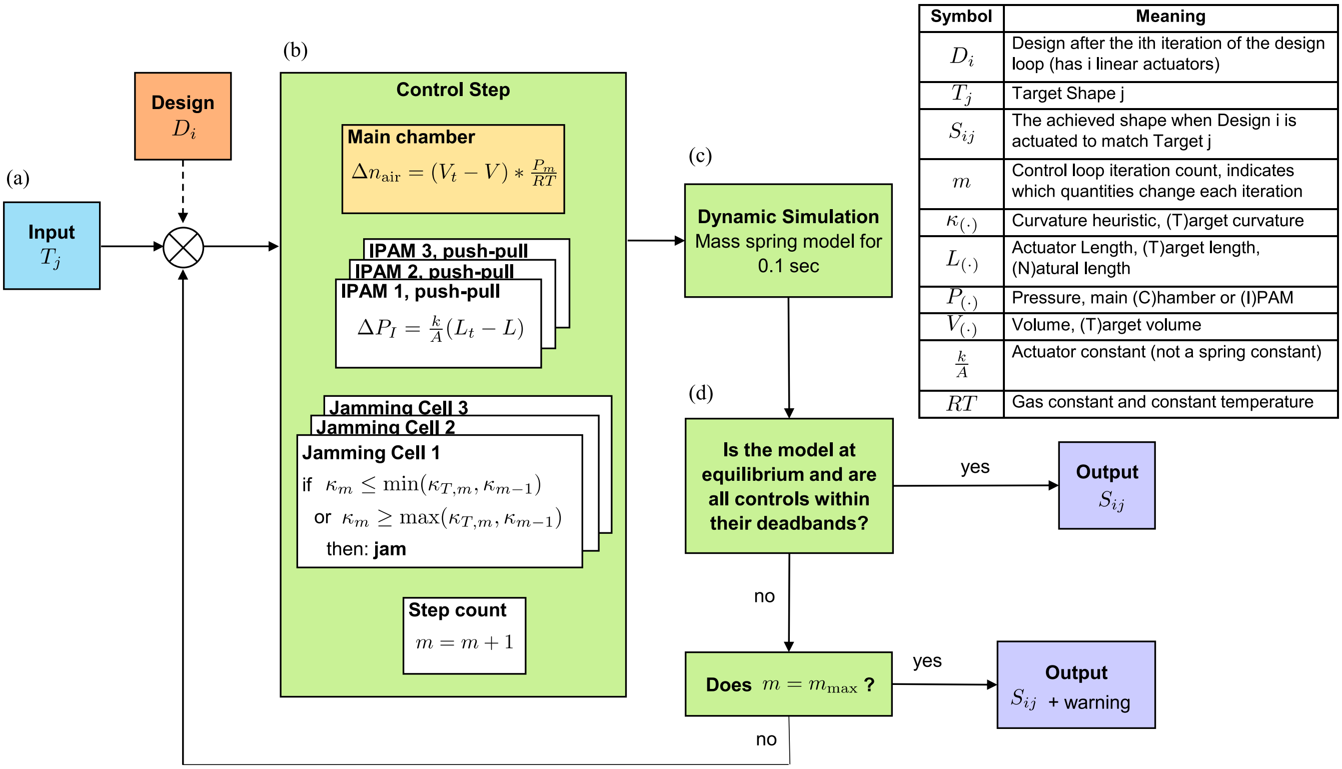
B. Controller

The control algorithm is described in Fig. 5. Each actuator is controlled independently but simultaneously. The control begins with an initial inflation to the target volume, in which only the main chamber inflation is actuated, prior to starting the control of the other actuators. The control variables are as follows: the amount of air in the main chamber, the pressure in each IPAM, and the jamming state of each jamming cell. The amount of air and the IPAM pressures are continuous control variables, while the jamming cell states are binary control variables. Furthermore, we limit the jamming cells to switch a maximum of one time from unjammed to jammed. The rate of main chamber inflation and IPAM inflation are limited in each control step.

1) Chamber Inflation Control: The controller that regulates the amount of air added to the main chamber is

$$\Delta n_{\text{air}} = (V_t - V) * \frac{P}{RT} \tag{11}$$

where V_t is the target volume, V is the current volume, P is the current absolute pressure, R is the gas constant, and T is the temperature, which is assumed to be constant. Under the


(a)
Input T_j
Design D_i
(b)
Control Step
Main chamber
$\Delta n_{\text{air}} = (V_t - V) * \frac{P_m}{RT}$
IPAM 3, push-pull
IPAM 2, push-pull
IPAM 1, push-pull
$\Delta P_I = \frac{k}{A}(L_t - L)$
Jamming Cell 3
Jamming Cell 2
Jamming Cell 1
if $\kappa_m \leq \min(\kappa_{T,m}, \kappa_{m-1})$
or $\kappa_m \geq \max(\kappa_{T,m}, \kappa_{m-1})$
then: jam
Step count
$m = m + 1$
(c)
Dynamic Simulation
Mass spring model for 0.1 sec
(d)
Is the model at equilibrium and are all controls within their deadbands?
yes
Output S_{ij}
no
Does $m = m_{\max}$?
yes
Output S_{ij} + warning
no

Symbol	Meaning
D_i	Design after the ith iteration of the design loop (has i linear actuators)
T_j	Target Shape j
S_{ij}	The achieved shape when Design i is actuated to match Target j
m	Control loop iteration count, indicates which quantities change each iteration
$\kappa_{(\cdot)}$	Curvature heuristic, (T)arget curvature
$L_{(\cdot)}$	Actuator Length, (T)arget length, (N)atural length
$P_{(\cdot)}$	Pressure, main (C)hamber or (I)PAM
$V_{(\cdot)}$	Volume, (T)arget volume
$\frac{k}{A}$	Actuator constant (not a spring constant)
RT	Gas constant and constant temperature

Fig. 5. Control algorithm drives a design to a target shape in simulation. (a) Inputs to the controller are the design and the target shape. (b) In each control step the main chamber inflation, IPAM pressurization, and jamming cell state are controlled based on a comparison between the current shape and the target shape. During the initial inflation stage, only the main chamber (yellow) is actuated until the volume closely matches the target volume. (c) The model of the designed shape is dynamically simulated to reflect the impact of the actuators. (d) The control algorithm finishes when the main chamber volume and IPAMs are within their deadbands or when the maximum number of control steps has been reached, outputting the achieved shape.

assumption that the pressure will remain constant over the next step in the control algorithm, the amount of air is incremented such that the volume matches the target volume. The target volume is determined by computing the volume of the sampled target shape. If pressure, rather than the amount of air in the chamber, is controlled, unstable inflation could occur, since, as the surface expands, the area over which the pressure is applied increases, thereby causing a higher force to be applied. Since surface area grows as a square of the linear distance on the surface (which is directly related to the spring length and, therefore, the resistive force in the springs), a pressure-controlled inflation can become unstable in simulation. Controlling the amount of air in the chamber avoids this possibility and more accurately reflects the physical system in which air must be added to the chamber to increase the pressure.

2) *IPAM Control:* The IPAMs are controlled similarly to the main chamber inflation. The target length of each IPAM is determined based on the distance between the centers of the two triangles to which it is attached in the sampled target shape. By assuming that the force on the IPAM from the combination of the chamber pressure and the outer membrane remains constant, the IPAMs are controlled to match their target length as follows:

$$\Delta P_I = \frac{k}{A}(L_t - L) \tag{12}$$

where P_I is the IPAM gauge pressure, L_t is the target length, L is the current length of the IPAM, and k/A is the constant relating pressure to natural length for the IPAM.

3) *Particle Jamming Cell Control:* The control of the particle jamming cells is based on a measure that approximates the curvature. In our design, each particle jamming cell is centered around an IPAM end that is a triangle in the surface mesh. The curvature metric is defined as follows:

$$\kappa = r_{\text{IPAM}} - \bar{r}_{\text{JC}} \tag{13}$$

where κ is the curvature of the jamming cell, r_{IPAM} is the radius to the center of the IPAM attachment triangle, and $\bar{r}_{\text{JC}}$ is the average radius of all triangles in the jamming cell. Here, the radius of a triangle is the distance from the origin to the centroid of the triangle. This formulation assumes that chamber inflation will move the whole cell radially inward or outward, and the IPAM actuation will move the IPAM attachment triangle in or out. Therefore, the particle jamming cell is used to fix the relative position of the IPAM attachment triangle and the rest of the cell. In contrast to the IPAMs and main chamber inflation, the jamming cells are passive actuators in that they cannot be controlled to make the cell achieve a specific curvature; rather they can only stop the curvature from changing. To control this, we track the curvature of the cell over time. If the current curvature is between the last curvature and the target curvature, it is assumed that the curvature of the cell will continue to approach the target curvature, and the cell is not jammed. Mathematically, this is expressed as

$$\min(\kappa_{T,m}, \kappa_{m-1}) \leq \kappa_m \leq \max(\kappa_{T,m}, \kappa_{m-1}) \tag{14}$$

where κ_m is the current curvature, κ_{m-1} is the last curvature, and $\kappa_{T,m}$ is the current target curvature, which can change over the actuation sequence as the radial sampling develops. It is necessary to use the maximum and minimum because we do not know whether the jamming cell curvature will approach the target curvature from above or below. If the current curvature is not between the last curvature and the current curvature, the cell is jammed. This occurs when either of these conditions are met

$$\begin{aligned} \kappa_m &\leq \min(\kappa_{T,m}, \kappa_{m-1}) \\ \kappa_m &\geq \max(\kappa_{T,m}, \kappa_{m-1}). \end{aligned} \tag{15}$$

Thus, a cell is jammed after the first step in which it starts moving away from the target curvature. Once jammed, a jamming cell is never unjammed.

C. Simulation Results

A control sequence for an example shape display and target shape is shown in Fig. 6. The design, or rest shape and actuator placement of the shape display, is a sphere with five IPAMs, and the target shape is a cube. Four of the IPAMs are aligned with the corners of the cube, while one IPAM connects the centers of two opposite faces. The control algorithm starts by inflating the sphere to match the volume of the target cube. Due to this inflation, the surface pressure pulls on the IPAMs, increasing their length. The jamming cell curvatures decrease, since the IPAMs pull in on the surface, resulting in surface indentations. After the preinflation, the IPAMs are actuated to match their target lengths and the jamming cells are jammed near their target curvature, while the main chamber inflation maintains the shape display at the target volume.

The PCC decreases during the initial inflation step before increasing monotonically when the IPAMs are actuated. This global shape matching improvement is not guaranteed by our algorithm, which uses only the local information pertinent to each actuator. However, in our experience, this local actuation approach provides a good heuristic for global shape matching. The final fit is dependent on the initial alignment of the target shape and the design, since the control algorithm does not allow for rotation of the shape display. In this case, the control algorithm works well because the IPAMs are aligned with the corners of the target cube to start.

The target jamming cell curvatures and IPAM lengths change during the control algorithm. This is because the sampled target shape changes over the course of the actuation, as the simulated surface comes in closer alignment to the target shape. These changes are dependent on the actuation sequence. For the initial sphere and target cube in this example, in which the radial sampling stays fairly consistent throughout the control sequence, these changes in target length and curvature are minor, but for other shapes, the target values for each IPAM and jamming cell change more significantly. Similarly, even though the target shape is symmetric, the target IPAM lengths and jamming cell

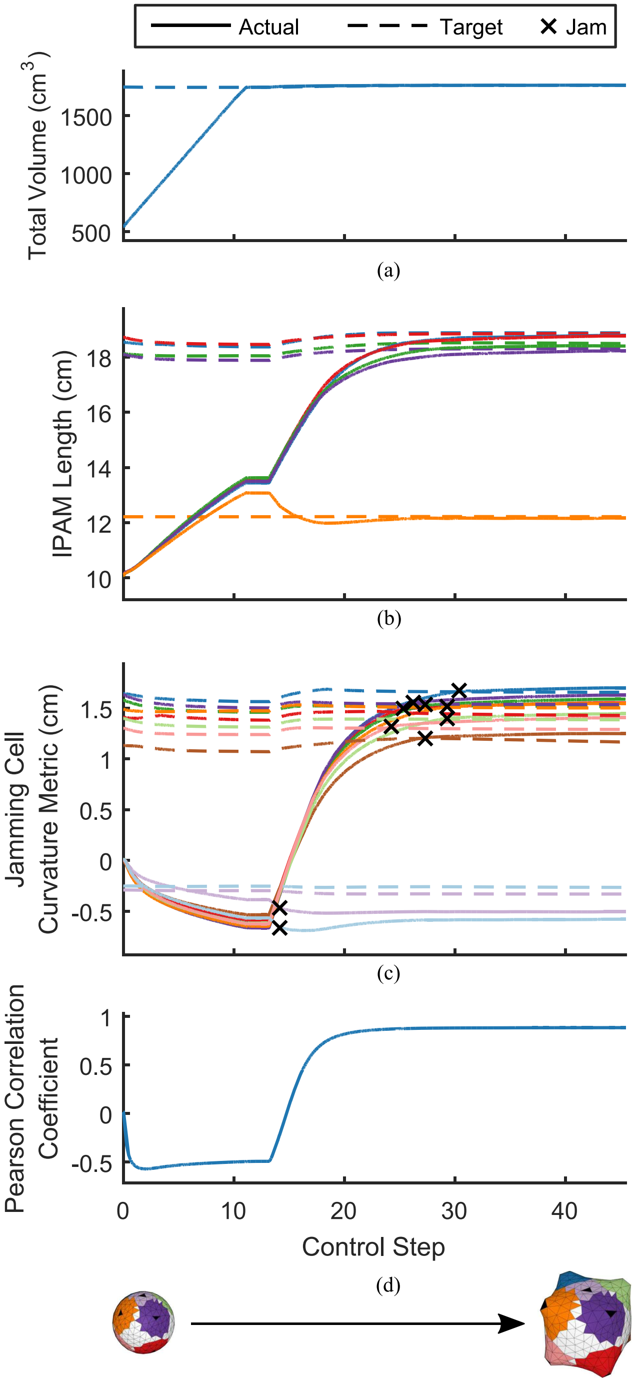


Fig. 6. Example control sequence for controlling a shape display with a spherical rest shape into a cube shape. (a) Main chamber pressure is controlled so that the target shape and the final shape match volume. Very little change in total volume is required after the initial inflation step. (b) IPAM length is controlled to match the distance between the two triangles on the sampled target mesh. Each color corresponds to an IPAM, with the target value for each IPAM shown as a dashed line. The orange line corresponds to the IPAM attached to the face, while the other four correspond to IPAMs attached to opposite corners. (c) Jamming cell curvature for each jamming cell is shown during the control sequence. The jamming cells are jammed when their curvature metric starts moving away from the target curvature, after the initial inflation. The color of the line matches the jamming cell shown on the design and achieved shape at the bottom. The target lengths of the IPAMs and curvatures of the jamming cells change over time as the alignment between the current shape and the target shape shifts, altering the sampled target mesh. (d) After the initial inflation, the control algorithm leads to improved shape matching over the control sequence according to the PCC, though no global matching parameter informs the control.

curvatures are not the same for all four corner IPAMs because the simulation mesh, which is used to both sample the target shape and simulate the shape display, is not symmetric. Thus, the precise alignment of each IPAM with the corner is not identical. The initial inflation step is also useful for the control of curvature achieved by the jamming cells because it creates a larger range of curvatures, which each jamming cell passes through, allowing them to fix their curvature closer to their target curvature. In the cube, this is particularly the case for the two jamming cells associated with the face-centered IPAM.

From the simulation, we can also obtain the force and strain requirements for the linear actuators throughout the control sequence. These data can be used to inform actuator design by detailing the required specifications for the actuators for a particular shape display. For the cube, the four IPAMs attached to the corners are in compression, while the IPAM attached to the faces is in tension. The compression force required from the corner IPAMs is greater than the tension force required from the face IPAM.

V. Design Algorithm

Ideally, one shape display would be able to match a wide variety of target shapes. However, such a versatile display requires a large number of actuators, which is not feasible due to space constraints in the physical system. Rather than creating a single, completely general shape display, we instead customize the design of the shape display to reach a set of desired target shapes. This allows us to reduce the number of actuators required to match those shapes well, so our shape display is feasible to fabricate. To customize the shape display, we developed an automatic design algorithm that determines a shape display design consisting of the rest shape of the membrane and the locations of the linear actuators and jamming cells. Because the jamming cells lead to path-dependent behavior, a complete control simulation must be run to determine how well a given design can match a target shape. Thus, we cannot use gradient-based optimization algorithms. Furthermore, due to the high number of degrees of freedom in our device, a genetic algorithm would be too time intensive. Instead, we implement a greedy algorithm for actuator placement that adds actuators iteratively to correct the worst error. In this section, we describe the algorithm and, then, demonstrate its use for both single and multiple shape matching.

A. General Framework

The design algorithm is shown in Fig. 7 and in Algorithm 1. The algorithm takes a base mesh, B, target shapes, $T_1, \ldots, T_n$ and the number of actuators to be added, p. First, it creates an initial design, D_0, defining the rest shape of the silicone membrane. The initial design is controlled in simulation to match each target as well as possible, using the control algorithm described in the previous section. The final state of the controlled design for a given target shape is called the achieved shape, S_{ij}. Then, a set of actuators consisting of a linear actuator and two jamming cells is added to the design where the error between the target shape and achieved shape is greatest, resulting in a new design, D_{i+1}. The process, then, repeats using the new design, adding actuators iteratively until a user-specified number of actuators has been reached.

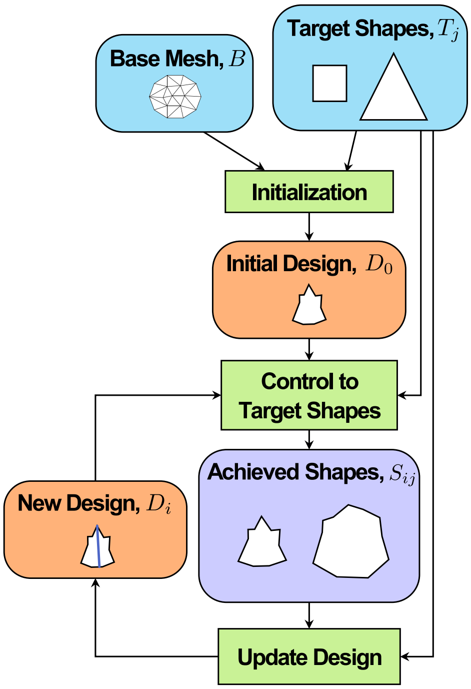


Fig. 7. Design algorithm consists of three processes: Initialization, control to target shapes, and update design. Together these processes determine the design (consisting of the rest shape and actuator configuration) and control sequences so that the shape display can match each target shape.

Algorithm 1: Shape Display Design Algorithm.

```
function DESIGN($B, T_1, \ldots, T_n, p$)
  $D_0 \leftarrow$ initialize($B, T_1, \ldots, T_n$)
  for $i \leftarrow 0, p-1$ do                    ▷ $p$ is number of actuators
    $e_{\max} \leftarrow 0$                           ▷ maximum scaled error
    for $j \leftarrow 1, n$ do
      $S_{ij} \leftarrow$ control($D_i, T_j$)
      $(e_j, t_j) \leftarrow$getMaxScaledError($T_j, S_{ij}$)
      if $e_j \geq e_{\max}$ then
        $e_{\max} \leftarrow e_j$
        $t_{\max} \leftarrow t_j$               ▷index of max error triangle
        $j_{\max} \leftarrow j$                   ▷index of max error target
      end if
    end for
    $t_{\text{opp}} \leftarrow$getOtherActuatorEnd($S_{ij_{\max}}, T_{j_{\max}}, t_{\max}$)
    $D_{i+1} \leftarrow$ addActuatorSet($D, t_{\max}, t_{\text{opp}}$)
  end for
  return $D_p$
end Function
```

1) Initialization: In the initialization step (initialize() in Algorithm 1), the algorithm first centers each target on the origin. Using the radial sampling technique described in Section IV-A, the algorithm constructs an initial design mesh by radially sampling each target shape using the coordinates of the vertices in the base mesh. The radius at each vertex is set to the average of the sampled target radii after compensating for the difference in volume between targets. Finally, this rest shape is scaled down to have a volume less than the smallest shape. This initialization is based on the assumption that the membrane will expand uniformly, radially outward upon inflation, without taking into account the complex deformations that will actually occur. The final downscaling is included to guarantee that the surface is in tension, avoiding wrinkling, which is challenging to model and not a feature of any of our target shapes. This rest shape, with no actuators other than main chamber inflation, is the initial design.

2) Control to Target Shapes: Following the initialization and using the control algorithm described in Section IV-B, the initial design is controlled to match each target shape as well as possible reaching the achieved shape, S_{ij} (control() in Algorithm 1). The control step for the first design consists only of inflating the main pressure chamber such that the volume of the achieved shape matches the volume of the target shape. For subsequent designs, this step may include actuating linear actuators and jamming cells.

3) Update Design: The design is updated by adding a linear actuator and two jamming cells to the design according to the following method. Once again using the radial sampling technique of Section IV-A, the error between the sampled target shape radius and the achieved shape radius is computed for the center of each triangle in the mesh and for each target shape (getMaxScaledError() in Algorithm 1). This error is scaled to compensate for differences in volume between the target shapes. The triangle with the greatest absolute error across all target shapes is chosen as one attachment point of the linear actuator.

Next, the other attachment point of the linear actuator is chosen according to the following criteria evaluated only for the target shape that had the maximum error (getOtherActuatorEnd() in Algorithm 1). In order to avoid collisions with the surface, the opposite triangle is constrained to be approximately across from the maximum error triangle. Specifically, the angle between the vectors connecting the origin to the maximum error triangle and the origin to the other triangle must be at least 120°. From the triangles satisfying this constraint, we choose the triangle with the greatest error with the same sign as that of the greatest error, since the linear actuators cannot push on one end while pulling on the other. This triangle becomes the other linear actuator attachment point.

Finally, jamming cells are added to the surface by adding the triangles within a prescribed radius of the linear actuator attachment points excluding triangles that are already part of another jamming cell (addActuatorSet() in Algorithm 1). By adding jamming cells in this way, the jamming cells can be thought of as providing a variable radius of influence for each linear actuator. When jammed, the linear actuator pushes or pulls on the whole jamming cell area, while when not jammed, the linear actuator primarily pushes or pulls only on the triangles to which it is directly attached. New linear actuators are restricted only to areas where there is not already a jamming cell or linear actuator attachment.

The process of controlling the design to match the target shapes repeats. A new maximum-error triangle is found, and the iterative process of adding sets of actuators continues until the number of actuators defined by the user is reached. The PCC is computed for every achieved shape.

B. Single Target Shape

To demonstrate the iterative actuator placement part of the algorithm (not including initialization) in an intuitive way, we can use a predetermined rest shape and a single target shape. For example, in Fig. 8, we show the progression of the algorithm for a cube starting with a spherical rest shape. The algorithm places the actuators one at a time, attaching the linear actuators to triangles aligned with the corners of the cube. This example shows that for simple shapes, the algorithm matches what we would intuitively design. Also, the design algorithm and the control sequence are able to match the shape well, and the achieved shape matches the target shape more closely with each additional actuator set, as we see in Fig. 9.

Similarly, we can run the design algorithm for a single target shape with a predetermined rest shape for other target shapes and evaluate the shape matching for each as shown in Fig. 10. As expected, the algorithm places the actuators in different locations given the different target shapes. For these examples, we can see that targets with sharper points tend to require higher actuator forces when starting with a spherical rest shape.

C. Multiple Target Shapes

The design algorithm can also be used to design a shape display that can reach more than one shape. This case uses the full design algorithm including the initialization step. In this multishape case, when adding a set of actuators, the algorithm finds the largest radial error across all targets after the error is normalized to the target volume. The algorithm adds the set of actuators according to the errors in that shape only, ignoring errors in the other achieved shapes. The linear actuator can theoretically be made to have no impact on the achieved shape for these other target shapes. By pressurizing the linear actuator such that its natural length is the distance between the triangles of the surface without the actuator, the added linear actuator will not apply any forces to the surface. Thus, adding an actuator set, which addresses the maximum error for one target shape, should not negatively influence the shape matching of the other target shapes. However, the control algorithm does not take this approach and instead uses all available actuators in for each target shape, so we do not have this strict guarantee. This approach allows us to take advantage of each actuator in every shape, rather than using only a subset of the actuators to reach each target shape.

We evaluated multiple target shape matching for different combinations of shapes. These results are presented in Fig. 11. The design algorithm was run for each subset of the four target shapes used: oval, pear, heart, and cube. The resultant design and the achieved shape for each set of target shapes are shown. The algorithm used five actuators, so these results can be compared to those in Fig. 10 for single shape matching. With multishape matching, the rest shape of the design is an average of the target shapes. By averaging the target shapes, we achieve reasonably

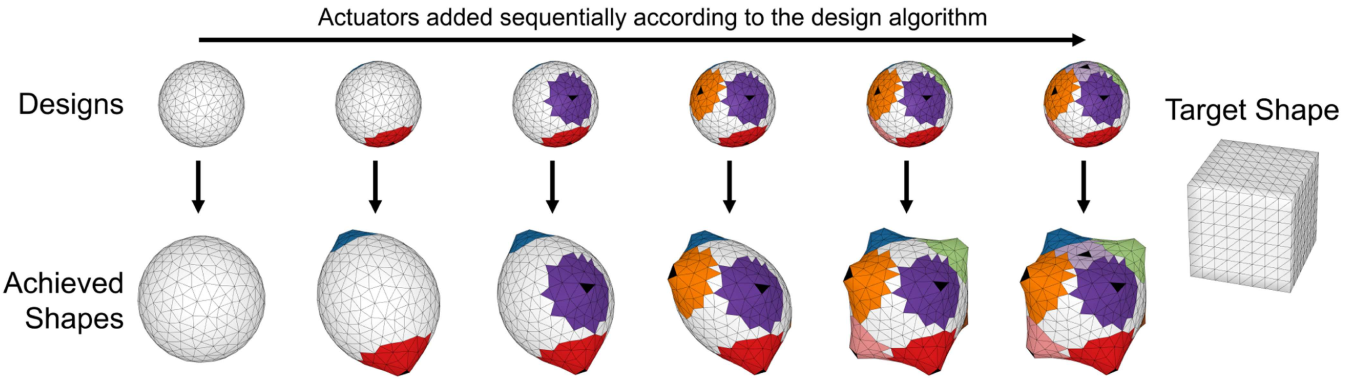

Actuators added sequentially according to the design algorithm
Designs
Achieved Shapes
Target Shape

Fig. 8. Progression of the design algorithm for a single shape. The top row shows how the design changes as actuators are added. For single shape matching, a predefined spherical rest shape is used, while linear actuators (black) are added to the triangle with the greatest radial error. Jamming cells, shown as colored sections, are added around each linear actuator attachment point to control the curvature. The target shape is used to compare the shapes in both the control and design steps to determine how to drive the design to the target by actuating the actuators appropriately or adding actuators where they are most needed.

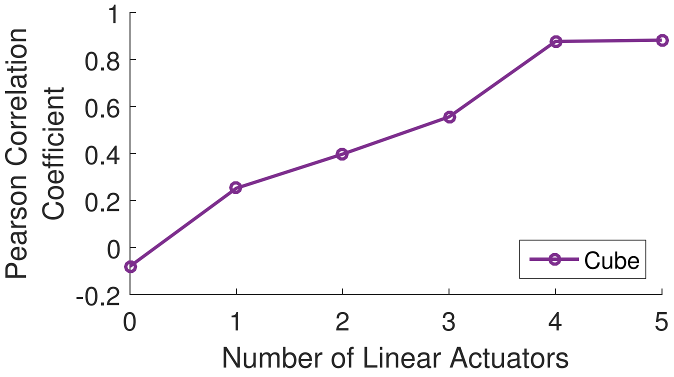


Fig. 9. For the cube example shown above, we can see that PCC, which measures the similarity between the target and the achieved shape, increases with each iteration of the design algorithm loop. As we would expect, each of the four corner actuator sets has a greater impact on the shape matching than the final linear actuator, which attaches to opposite faces of the target cube shape.

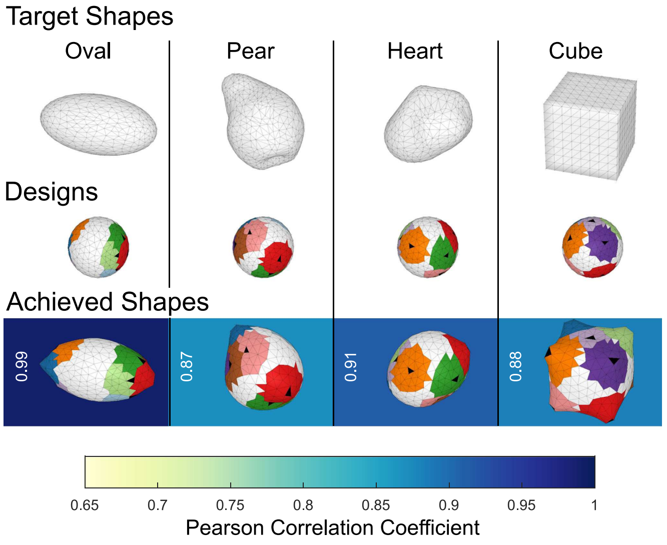


Fig. 10. Single shape matching algorithm results for different shapes. The design for each target shape uses the same spherical rest shape, but the design algorithm places actuators in different positions to achieve good matching for each target shape. The achieved shapes approximate the target shapes well, as shown with the fit as measured by the PCC.

good matching before adding actuators, and the actuators can, then, be used to address differences between the target shapes.

While the number of actuators that can be used in a physical device is limited due to space constraints, we also tested how the design algorithm would perform with more actuators. For three of the multishape matching cases, we ran the design algorithm using 20 linear actuators and 40 corresponding jamming cells. The results are shown in Fig. 12. The area of each jamming cell was reduced by a factor of four relative to that used in Fig. 11, so the total surface area covered by jamming cells was similar in the final configuration (not accounting for overlaps). From these results, we see that more actuators help achieve better matching, although the quality of shape matching levels off after a certain point with only minor improvements after that point. This is reasonable because the major features may be captured by a small number of actuators. There is some actuator sharing, which can be seen from the fact that adding the oval as a target shape in Fig. 12(c) does not require additional actuators for similar fit quality relative to the test shown in Fig. 12(b).

D. Hardware Validation

Finally, we validated the design and simulation by building a physical shape display for a multishape matching case with two target shapes, an oval and a heart. Fig. 13 shows the progression of the design sequence for these shapes as actuators are added. The physical shape display can be constructed according to the design and actuated according to the results of the simulated control algorithm. To manufacture the device, two 3-D printed molds were made based on the design for the inner and outer silicone membranes. The molds were in the shape of the rest shape of the device (scaled radially to allow for the thickness of the jamming cells) and included features at points where IPAMs were attached. The silicone membranes were made by brush molding silicone rubber (Eco-flex 00-30, Smooth-On, Inc., Easton, PA). The boundaries of jamming cells were made of silicone and their locations were approximated using the IPAM attachment points. These boundaries were manually glued onto the silicone membrane using Sil-Poxy (Smooth-On, Inc.).

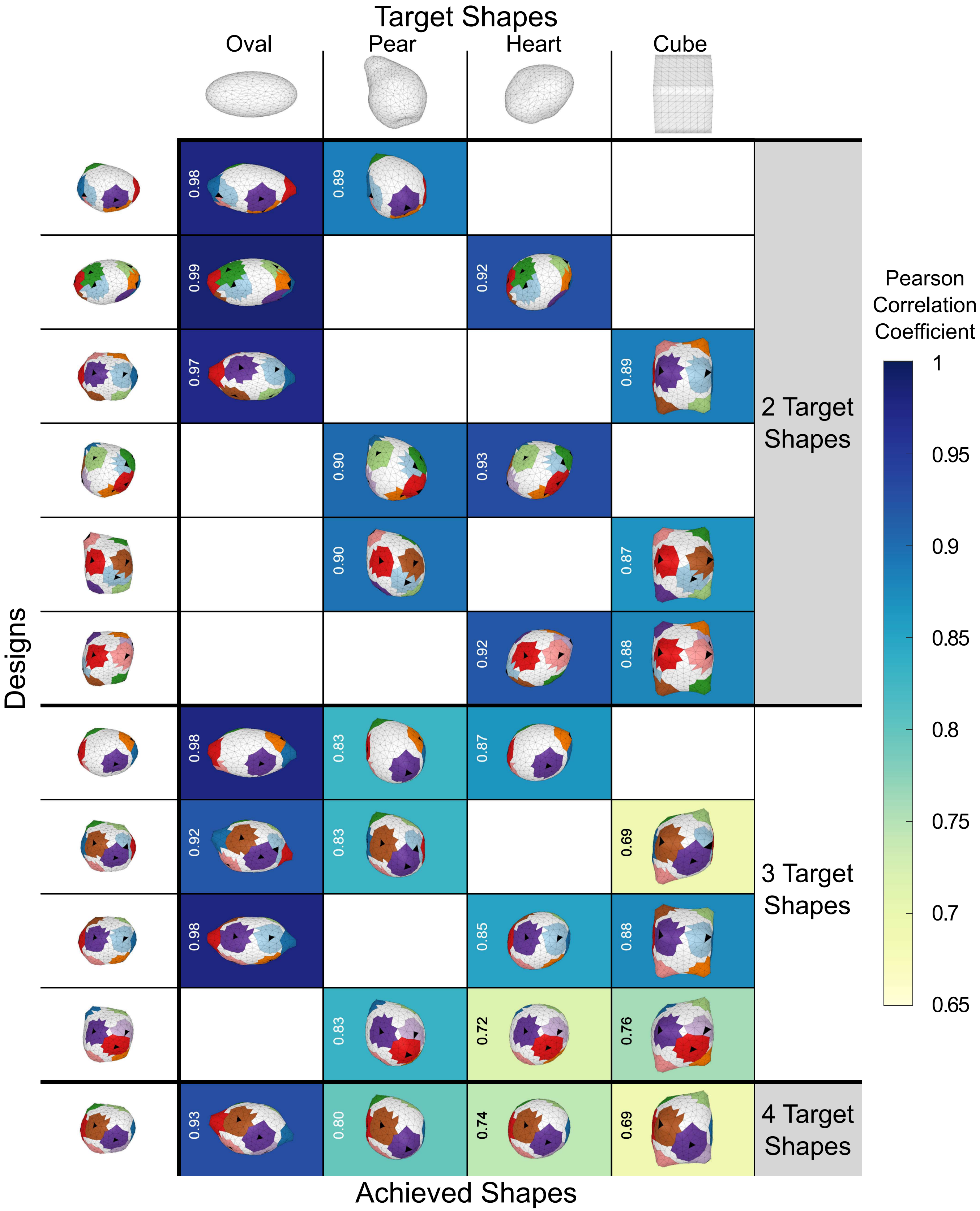
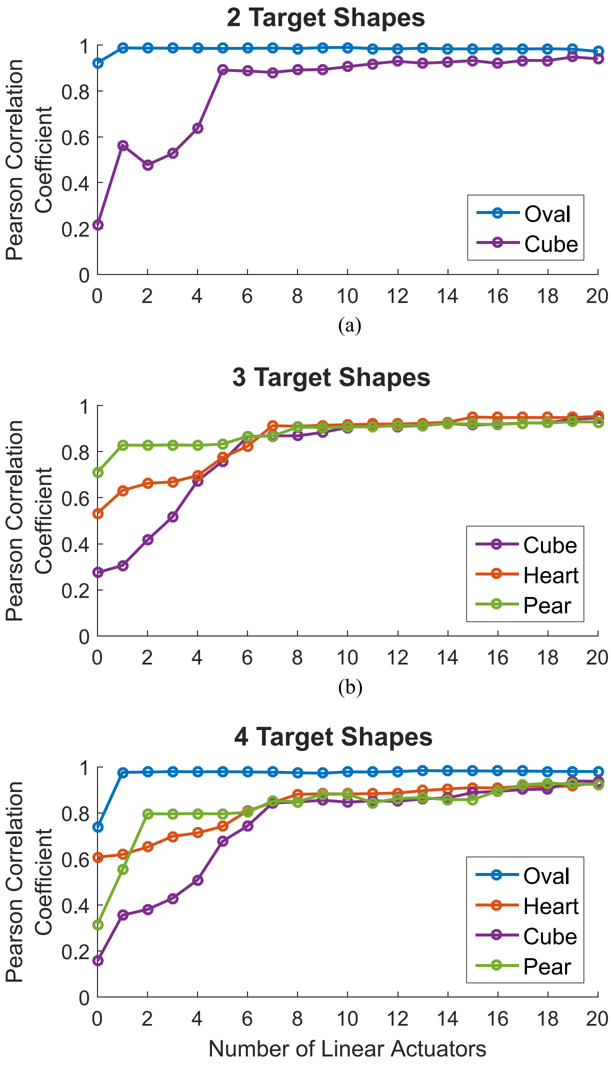
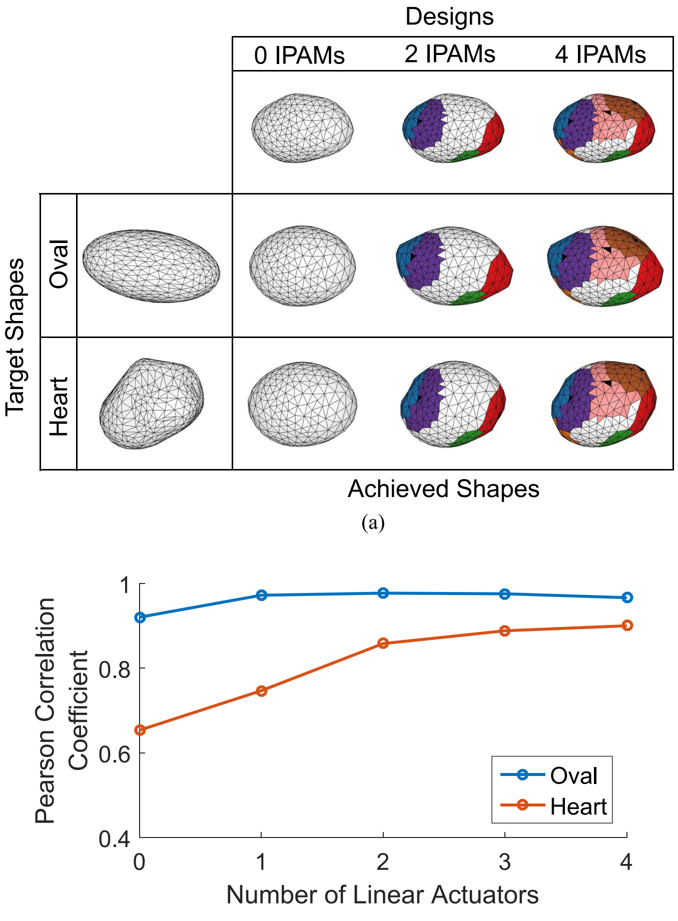


Fig. 11. Design algorithm was used to produce designs to match subsets of the target shapes. Each row of this table shows a design produced by the design algorithm as well as the simulated achieved shapes resulting from that design. Each design places five IPAMs and ten jamming cells. The impact of the initialization, which averages the two target shapes radially, can be seen in the different rest shapes of the designs. As more targets are added, the fit of each target, measured by the PCC and shown as the color behind each mesh, declines. The shapes are shown in the orientation they were in during the design algorithm.



(c)

Fig. 12. Design algorithm was tested with a high number of actuators for three multishape matching cases: (a) Two target shapes, the oval and the cube, (b) three target shapes, the cube, the heart, and the pear, and (c) four target shapes, the oval, the heart, the cube, and the pear. The additional actuators improve the shape matching, as measured by the PCC, though the shape matching levels off once a critical number of actuators have been added.



(b)

Fig. 13. Example of multiple shape matching. (a) Table of the target shapes, intermediate and final designs, and intermediate and final achieved shapes. This design, which was used for the hardware validation, is different than that shown in Fig. 11 for the same target shapes because it uses only four IPAMs, and it uses larger radius jamming cells. The shapes are also shown in a different orientation. (b) PCC of the shapes as the algorithm progresses is shown. As more linear actuators (IPAMs) are added to the design, the fits of both target shapes improve, as seen both in achieved shapes and in the PCC.

The outer membrane was attached to the top of the jamming cell boundaries, and the jamming cells were filled with coffee grounds as the outer membrane was attached. A control sequence for the device was created from the simulation consisting of linear actuator lengths and main chamber pressures at time points when the jamming cells were jammed. The resulting shapes corresponding to the oval and heart are shown in Fig. 14.

To validate the model and the design algorithm, we compared the physical, actuated shape to that predicted by the model. We scanned the shape using a Microsoft Kinect and used Microsoft's Kinect Fusion [46] to reconstruct the surface of the shape display. The shape display surface was manually extracted from the scene, and the shape was roughly aligned to the model. Then, an iterative closest point algorithm [47], [48] was used to align the scanned mesh to that of the model. The radial sampling method was used to compare the scanned shape to the simulated achieved shape and to the original target shape for both the oval and the heart. The results are reported in Fig. 14. The region of the surface where the pneumatic tubes exit the interior of the device was not included in the scan. This region and some of the surface surrounding it were not visible during the scan due to occlusions. The percentage of the surface included in the scan is also shown in Fig. 14. To determine the effectiveness of this measurement technique, we also measured an acrylonitrile butadiene styrene (ABS) mold made with a MakerBot Replicator 2 used in the shape display construction and compared it to the model that was used to print it. This method resulted in a PCC of 0.99, a mean absolute radial error (MARE) of 0.09 cm, and a mean percent radial error (MPRE) of 1.6%, indicating that the method produces reliable measurement results.

E. Discussion

Based on the validation results shown in Fig. 14, we see that the physical shape display has good correspondence with the

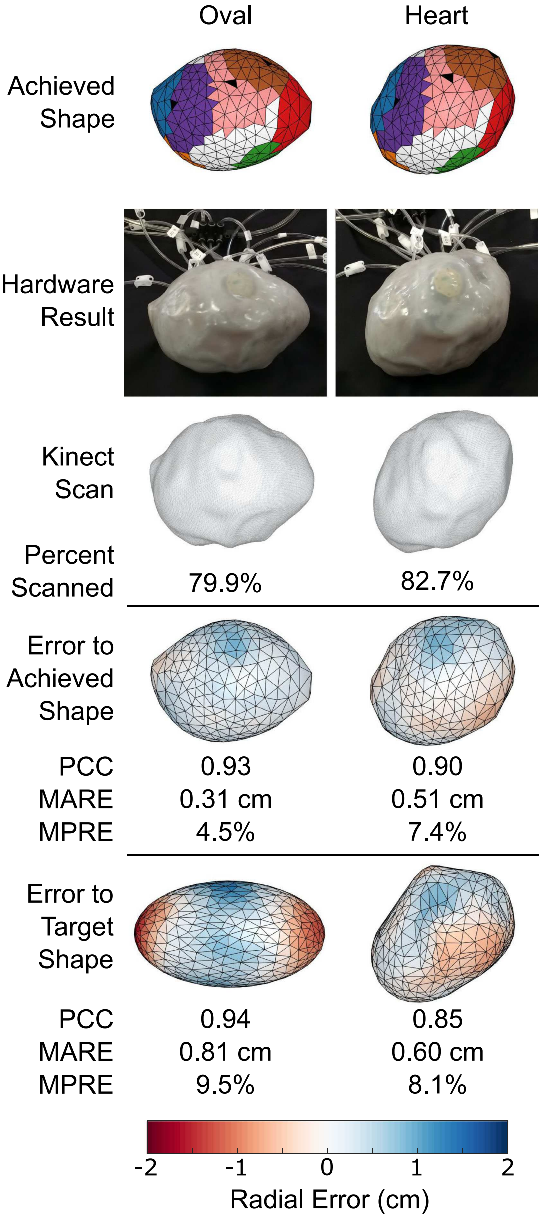


Fig. 14. Example of the hardware designed by the algorithm and actuated using the control sequence determined by the algorithm. The targets and design are those shown in Fig. 13. The achieved shape from simulation show the position of the IPAMs, which can be seen in the picture of the hardware. The hardware is scanned using a Microsoft Kinect, and the amount of the surface that is scanned reported. The error is shown between from the hardware to the simulated achieved shape and to the original target shape are reported, with the PCC, MARE, and MPRE reported.

modeled display, with MAREs between the scan of the device surface and the model averaging less than 10% relative to both the simulated achieved shape and to the original target shape. There are a variety of sources of error possible between the physical device and the model. Much of the manufacturing process was manual, which could have led to errors. For example, the alignment of the outer membrane with the inner membrane and IPAMs was approximate. Also, the jamming cell placement was not as precise as in the simulation. Additionally, the model did not capture the interactions between IPAMs in the interior of the device. Since the IPAMs must bend around each other, they do not always provide the direct pushing or pulling force that was modeled. The model also does not account for the variation of the stiffness across the surface of the device caused by the silicone that separates the jamming cells and the Sil-Poxy required to secure them. These inconsistencies in the surface material properties likely contribute to some of the wrinkles that can be seen on the surface of the Kinect scans, particularly around features such as the IPAM attachment points and jamming cell edges. The part of the device surface where the air tubes exit the interior, which consists of a thick silicone plug of about 5 cm diameter, was also not modeled.

The design algorithm automatically determines a rest shape and places actuators to reach a variety of target shapes. For the shapes with predictable actuator placement, such as the cube, the algorithm matches our intuition for where actuators should be placed. For the multiple target case that was used for physical validation, we can see from the PCC plot (see Fig. 13) that even though an actuator was placed to address error in only one of the target shapes, it could also contribute to the shape matching in the other shape. The first actuator in particular was placed according to error in the oval, but the heart sees great improvement as well from that actuator. Additionally, the initialization improves the initial fit dramatically. This leads to a more effective use of the actuators, since they can address differences between the target shapes, rather than just differences between the rest shape and the target shape. The final fit of the designed display to the target shapes is also sensitive to the initial alignment of the target shapes, since we do not allow rotation prior to running the design algorithm.

The results in Figs. 11 and 12 show the potential and limitations of the design algorithm for matching multiple target shapes, including both similar shapes, such as the oval and heart, and dissimilar shapes, such as the pear and the cube. For shapes with prominent discrete vertices, such as the cube, the position of the actuators is critical. Thus, with only five actuators and more than two target shapes, the cube is not able to align the actuators with its corners, and its shape matching quality declines. However, if more actuators can be used, the cube match improves. Shapes with less prominent vertices seem better able to use added actuators to counter error than other target shapes. Since this hardware design limits the possible number of actuators due to space constraints, it is better suited to shapes with less prominent distinct vertices.

While the IPAM linear actuators can both push and pull, we found pushing to be a more important actuation direction in achieving shapes similar to the target shapes. The highest forces in the linear actuators were in the pushing direction for all of the single shapes tested. This reflects the fact that most of the tested shapes had convex features, rather than concave. The pear does have a concave section on its bottom, and the linear actuator that was automatically placed there was in tension to produce this feature. Both the design algorithm and the metrics used to evaluate the shape matching relied on a radial conception of the device. This limits the potential shapes that can be designed for and achieved. Also, the size of triangle elements in our mesh limit the level of surface detail. For more complex shapes, surface actuation approaches may be more appropriate. A surface-based approach may also be able to better capture specific features of the shape, such as edges or corners, that affect human perception.

The complex control of our device, primarily due to the path-dependent nature of the jamming cells, limits the available options for design optimization. Evaluations of the fit are expensive because they are dependent on both the design of the shape display and the result of the closed-loop controller, and we cannot take a gradient in the design space. Genetic algorithms are appealing for such gradient-free optimization, but a large number of parameters would require a large population and many generations to converge to a good solution, leading to many fitness evaluations and long optimization times. In contrast, our greedy iterative algorithm depends only on the number of target shapes and the number of linear actuators to add. Thus, the algorithm can run much more quickly than would be possible with a genetic algorithm.

VI. Conclusion

In this article, we described a novel 3-D haptic shape display, a mass-spring model of the device, a control algorithm to drive the device to match a given target shape, and an automatic design algorithm that determines the shape and actuator location of the device to make the shape display capable of matching a small set of target shapes with a small number of actuators. The design extended previous work in haptic jamming surfaces, making the device fully 3-D using soft pneumatic actuators. The control algorithm used local actuator information to effect global improvement in shape matching. The design algorithm took advantage of computational modeling to achieve reasonably good shape matching with a small number of internal actuators, though this shape matching was limited for shapes that have many prominent vertices.

Many avenues are open for future work, both in device design and in algorithm development. For the device, alternate actuation techniques could provide the ability to reach a wider variety of shapes. More compact actuation would allow a higher number of actuators to be used, which would make each display more versatile and could make the shape matching more precise. Adding sensing to the hardware, in particular, embedded surface sensing, would allow for feedback control and could enable the device to collect information on the user's interaction with the device. Additionally, the target mechanical properties of the final shape could be taken into account, which would more directly include the particle jamming feature in the design. For design automation, other algorithms may be more appropriate. Search algorithms that explore more complex actuation sequences could exploit the path-dependency of the jamming cells. Alternatively, avoiding this path-dependency could allow for an approach based on a force equilibrium, which could speed up simulation and lead to gradients useful for design optimization. A different parameterization of the shape surface may prove well-suited to genetic algorithms, which would allow for a more thorough search of the design space.

In order to use the device for the proposed applications, a few key developments would be needed. For organ simulation for palpation training, the ability to modulate the stiffness of the device to match particular disease states is critical, because stiffness is an important element for diagnosis via palpation. This stiffness modulation could result from responsive actuation, which would require sensing, or use of the jamming cells for stiffness control instead of shape control. Including target stiffness states in addition to the target shapes would extend the potential usefulness of the design algorithm, but significant modifications would be needed. A user evaluation of the device by expert practitioners would also be essential to develop the shape display into a useful tool, for example, for organ simulation. For virtual and augmented reality applications, in which the device could be used as a form of digital clay, shape and surface sensing would be needed to enable bilateral interaction with a virtual model. It would be important to identify the required versatility of a design that would be sufficient for these applications. Finally, understanding human haptic perception of shape would inform the design requirements for a general shape display.

Acknowledgment

The authors would like to thank E. Hawkes for his guidance in making and modeling IPAMs and A. Stanley for his assistance in modeling and initial design.

References

[1] W. McNeely, "Robotic graphics: A new approach to force feedback for virtual reality," in *Proc. IEEE Virtual Reality Annu. Int. Symp.*, 1993, pp. 336–341.

[2] D. Leithinger, D. Lakatos, A. DeVincenzi, M. Blackshaw, and H. Ishii, "Direct and gestural interaction with relief: A 2.5 D shape display," in *Proc. 24th Annu. ACM Symp. User Interface Softw. Technol.*, 2011, pp. 541–548.

[3] J. Rossignac *et al.*, "Finger sculpting with digital clay: 3D shape input and output through a computer-controlled real surface," in *Proc. Shape Model. Int.*, 2003, pp. 229–231.

[4] A. A. Stanley, K. Hata, and A. M. Okamura, "Closed-loop shape control of a haptic jamming deformable surface," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 2718–2724.

[5] S. Follmer, D. Leithinger, A. Olwal, A. Hogge, and H. Ishii, "inFORM: Dynamic physical affordances and constraintsthrough shape and object actuation," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2013, pp. 417–426.

[6] Y. Cai, J. Wang, L. Zhao, Y. Liu, and Y. Wang, "A novel shape-changing haptic table-top display," in *Proc. Int. Conf. Opt. Instruments Technol.: Optoelectronic Imag./Spectroscopy Signal Process. Technol.*, 2018, pp. 178–182.

[7] K. Nakagaki, D. Fitzgerald, Z. J. Ma, L. Vink, D. Levine, and H. Ishii, "inFORCE: Bi-directional 'force' shape display for haptic interaction," in *Proc. 13th Int. Conf. Tangible, Embedded, Embodied Interact.*, 2019, pp. 615–623.

[8] A. A. Stanley and A. M. Okamura, "Deformable model-based methods for shape control of a haptic jamming surface," *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 2, pp. 1029–1041, Feb. 2017.

[9] T.-H. Yang, S.-Y. Kim, C. H. Kim, D.-S. Kwon, and W. J. Book, "Development of a miniature pin-array tactile module using elastic and electromagnetic force for mobile devices," in *Proc. 3rd Joint EuroHaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst.*, 2009, pp. 13–17.

[10] C. Wagner, S. Lederman, and R. Howe, "A tactile shape display using RC servomotors," in *Proc. 10th Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst.*, 2002, pp. 354–355.

[11] A. F. Siu, E. J. Gonzalez, S. Yuan, J. Ginsberg, A. Zhao, and S. Follmer, "shapeShift: A mobile tabletop shape display for tangible and haptic interaction," in *Proc. Adjunct Publication 30th Annu. ACM Symp. User Interface Softw. Technol.*, 2017, pp. 77–79.

[12] D. Leithinger and H. Ishii, "Relief: A scalable actuated shape display," in *Proc. ACM Int. Conf. Tangible, Embedded Embodied Interact.*, 2010, pp. 221–222.

[13] H. Iwata, H. Yano, F. Nakaizumi, and R. Kawamura, "Project FEELEX: Adding haptic surface to graphics," in *Proc. ACM SIGGRAPH*, 2001, pp. 469–475.

[14] P. Taylor, A. Moser, and A. Creed, "A sixty-four element tactile display using shape memory alloy wires," *Displays*, vol. 18, no. 3, pp. 163–168, 1998.

[15] M. Coelho, H. Ishii, and P. Maes, "Surflex: A programmable surface for the design of tangible interfaces," in *Proc. ACM CHI Extended Abstracts Human Factors Comput. Syst.*, 2008, pp. 3429–3434.

[16] S. Klare and A. Peer, "The formable object: A 24-degree-of-freedom shape-rendering interface," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 3, pp. 1360–1371, Jun. 2015.

[17] A. Mazzone and A. Kunz, "Sketching the future of the SmartMesh wide area haptic feedback device by introducing the controlling concept for such a deformable multi-loop mechanism," in *Proc. 1st Joint Eurohaptics Conf. Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst.*, 2005, pp. 308–315.

[18] C.-H. King, M. Culjat, M. Franco, J. Bisley, E. Dutson, and W. Grundfest, "Optimization of a pneumatic balloon tactile display for robot-assisted surgery based on human perception," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 11, pp. 2593–2600, Nov. 2008.

[19] S. Gallo, C. Son, H. J. Lee, H. Bleuler, and I.-J. Cho, "A flexible multimodal tactile display for delivering shape and material information," *Sens. Actuators A, Phys.*, vol. 236, pp. 180–189, 2015.

[20] L. Yao, R. Niiyama, J. Ou, S. Follmer, C. D. Silva, and H. Ishii, "PneUI: Pneumatically actuated soft composite materials for shape changing interfaces," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2013, pp. 13–22.

[21] B. C. MacMurray *et al.*, "A variable shape and variable stiffness controller for haptic virtual interactions," in *Proc. IEEE Int. Conf. Soft Robot.*, 2018, pp. 264–269.

[22] N. Akbar and S. Jeon, "Encountered-type haptic interface for grasping interaction with round variable size objects via pneumatic balloon," in *Proc. Eurohaptics*, 2014, pp. 363–372.

[23] H. Iwata, H. Yano, and N. Ono, "Volflex," in *Proc. ACM SIGGRAPH Emerging Technol.*, 2005, p. 31.

[24] A. A. Stanley and A. M. Okamura, "Controllable surface haptics via particle jamming and pneumatics," *IEEE Trans. Haptics*, vol. 8, no. 1, pp. 20–30, Jan.–Mar. 2015.

[25] M. Li, T. Ranzani, S. Sareh, L. D. Seneviratne, P. Dasgupta, H. A. Wurdemann, and K. Althoefer, "Multi-fingered haptic palpation utilizing granular jamming stiffness feedback actuators," *Smart Mater. Struct.*, vol. 23, no. 9, 2014, Art. no. 095007.

[26] S. Follmer, D. Leithinger, A. Olwal, N. Cheng, and H. Ishii, "Jamming user interfaces: Programmable particle stiffness and sensing for malleable and shape-changing devices," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2012, pp. 519–528.

[27] J. R. Amend, Jr., E. Brown, N. Rodenberg, H. M. Jaeger, and H. Lipson, "A positive pressure universal gripper based on the jamming of granular material," *IEEE Trans. Robot.*, vol. 28, no. 2, pp. 341–350, Apr. 2012.

[28] E. Steltz, A. Mozeika, N. Rodenberg, E. Brown, and H. M. Jaeger, "JSEL: Jamming skin enabled locomotion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2009, pp. 5672–5677.

[29] N. Aihara, T. Sato, and H. Koike, "Highly deformable interactive 3D surface display," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2012, pp. 91–92.

[30] N. S. Usevitch, R. Khanna, R. M. Carrera, and A. M. Okamura, "End effector for a kinesthetic haptic device capable of displaying variable size and stiffness," in *Proc. Eurohaptics*, 2016, pp. 363–372.

[31] L. He, N. Herzig, S. de Lusignan, and T. Nanayakkara, "Granular jamming based controllable organ design for abdominal palpation," in *Proc. 40th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, 2018, pp. 2154–2157.

[32] N. Takizawa, H. Yano, H. Iwata, Y. Oshiro, and N. Ohkohchi, "Encountered-type haptic interface for representation of shape and rigidity of 3D virtual objects," *IEEE Trans. Haptics*, vol. 10, no. 4, pp. 500–510, Oct.–Dec. 2017.

[33] H. Lipson, "Challenges and opportunities for design, simulation, and fabrication of soft robots," *Soft Robot.*, vol. 1, no. 1, pp. 21–27, 2014.

[34] M. Skouras, B. Thomaszewski, B. Bickel, and M. Gross, "Computational design of rubber balloons," in *Proc. Comput. Graph. Forum (Proc. Eurographics)*, vol. 31, no. 2, 2012, pp. 835–844.

[35] M. Skouras, B. Thomaszewski, S. Coros, B. Bickel, and M. Gross, "Computational design of actuated deformable characters," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 82:1–82:10, 2013.

[36] L.-K. Ma, Y. Zhang, Y. Liu, K. Zhou, and X. Tong, "Computational design and fabrication of soft pneumatic objects with desired deformations," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 239:1–239:12, 2017.

[37] G. Renner and A. Ekárt, "Genetic algorithms in computer aided design," *Comput.-Aided Des.*, vol. 35, pp. 709–726, 2003.

[38] J. Hiller and H. Lipson, "Automatic design and manufacture of soft robots," *IEEE Trans. Robot.*, vol. 28, no. 2, pp. 457–466, Apr. 2012.

[39] C. Baykal, C. Bowen, and R. Alterovitz, "Asymptotically optimal kinematic design of robots using motion planning," *Auton. Robots*, vol. 43, no. 2, pp. 345–357, 2019.

[40] A. V. Gelder, "Approximate simulation of elastic membranes by triangulated spring meshes," *J. Graph. Tools*, vol. 3, no. 2, pp. 21–42, 1998.

[41] C. Duriez, "Control of elastic soft robots based on real-time finite element method," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 3982–3987.

[42] E. W. Hawkes, D. L. Christensen, and A. M. Okamura, "Design and implementation of a 300% strain soft artificial muscle," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 4022–4029.

[43] L. Li and V. Volkov, "Inflatable Models," *J. Comput. Sci. Technol.*, vol. 21, no. 2, pp. 154–158, 2006.

[44] J. Kent, R. E. Parent, and W. E. Carlson, "Establishing correspondences by topological merging: A new approach to 3-D shape transformation," *Graph. Interface*, vol. 91, pp. 271–278, 1991.

[45] J. P. Lewis, "Fast template matching," *Vision Interface*, vol. 95, pp. 120123, 1995.

[46] S. Izadi *et al.*, "KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proc. ACM Symp. User Interface Softw. Technol.*, 2011, pp. 559–568.

[47] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," in *Proc. SPIE 1611, Sensor Fusion IV: Control Paradigms Data Struct.*, 1992, pp. 586–606.

[48] J. Wilm. Iterative closest point. 2013. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/27804-iterative-closest-point

Margaret Koehler (Student Member, IEEE) received the B.S. and M.S. degrees in mechanical engineering in 2014 and 2016, respectively, from Stanford University, Stanford, CA, USA, where she is currently working toward the Ph.D. degree in mechanical engineering.

Her research interests include haptics, medical robotics, simulation, and control for a wide variety of applications.

Nathan S. Usevitch (Student Member, IEEE) received the B.S. degree in mechanical engineering from Brigham Young University, Provo, UT, USA, in 2015 and the M.S. degree in mechanical engineering in 2017 from Stanford University, Stanford, CA, USA, where he is currently working toward the Ph.D. degree in mechanical engineering.

His research interests include soft robotics, haptics, and multirobot systems.

Allison M. Okamura (Fellow, IEEE) received the B.S. degree from the University of California, Berkeley, Berkeley, CA, USA, in 1994, and the M.S. and Ph.D. degrees from Stanford University, Stanford, CA, USA, in 1996 and 2000, respectively, all in mechanical engineering.

She is currently a Professor of Mechanical Engineering with Stanford University. Her research interests include haptics, teleoperation, medical robotics, virtual environments and simulation, neuromechanics and rehabilitation, prosthetics, and engineering education.

Prof. Okamura is the Editor-in-Chief for the IEEE ROBOTICS AND AUTOMATION LETTERS.