ADVANCED FUNCTIONAL MATERIALS
www.afm-journal.de

Inverse Design of Inflatable Soft Membranes Through Machine Learning

Antonio Elia Forte, Paul Z. Hanakata, Lishuai Jin, Emilia Zari, Ahmad Zareei, Matheus C. Fernandes, Laura Sumner, Jonathan Alvarez, and Katia Bertoldi**

Across fields of science, researchers have increasingly focused on designing soft devices that can shape-morph to achieve functionality. However, identifying a rest shape that leads to a target 3D shape upon actuation is a non-trivial task that involves inverse design capabilities. In this study, a simple and efficient platform is presented to design pre-programmed 3D shapes starting from 2D planar composite membranes. By training neural networks with a small set of finite element simulations, the authors are able to obtain both the optimal design for a pixelated 2D elastomeric membrane and the inflation pressure required for it to morph into a target shape. The proposed method has potential to be employed at multiple scales and for different applications. As an example, it is shown how these inversely designed membranes can be used for mechanotherapy applications, by stimulating certain areas while avoiding prescribed locations.

1. Introduction

2D sheets that can morph from flat into 3D shapes have become a powerful and versatile platform to realize deployable systems,[1–3], frequency shifting antennae,[4] active building facades,[5] as well as camouflage devices.[6] Several avenues have been successful in achieving shape changing capabilities. Origami principles have enabled transformation of flat sheets into elaborate shapes via folding;[7–9] morphable sheets have been realized by combining materials that can expand and contract by different amounts in response to external stimuli such as temperature, humidity, or pH;[4,10] inflating membranes reinforced with stiff components have shown promise for the realization of shape changing surfaces.[3,6,11–14] Focusing specifically on inflatable membranes, these are either made in a complex deflated shape and out of a single homogeneous material[3,11–15] or in a simple rest shape by optimizing the material locally to guide the deformation upon inflation.[6,16,17] However, irrespective of their fabrication method, programming 2D sheets to obtain a target 3D shape is a non-trivial task that typically requires the use of optimization algorithms.[16,18–21] These include gradient-free algorithms[16,22] as well as gradient-based methods.[18,21]

Here, we consider inflatable membranes comprising soft and stiff domains and show how machine learning tools can be used to design configurations of such domain that result in target shapes upon inflation. While machine learning methods have been classically employed for image recognition[23] and language processing,[24] they have also recently emerged as powerful tools to solve mechanics problems.[25–38] Building on these recent successes, we demonstrate that these tools can be extended to study the nonlinear mechanics of inflatable systems. By using neural networks (NNs) trained on finite element (FE) simulations, we are able to solve the inverse design problem. This allows us to prescribe a target 3D shape in input and obtain a spatially defined 2D design for a soft membrane, comprising soft and stiff elastomeric pixels, as output. Such a designed membrane is then inflated to an optimal pressure—also instructed by the model—and morphs into the desired shape. The platform hereby introduced, despite being presented in a macroscale framework, is highly scalable and holds potential for many fields of science and engineering, enabling applications such as morphable surfaces for architecture, soft sensors, ergonomic garments, and medical devices. As an example, we show how these membranes can be used in mechanotherapy for wound healing.

2. Our Platform

Our platform consists of a square sheet of elastomeric material with edges of 10 cm mounted on an acrylic chamber which

A. E. Forte, L. Jin, E. Zari, A. Zareei, M. C. Fernandes, J. Alvarez, K. Bertoldi
J.A. Paulson School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138, USA
E-mail: antonio.forte@kcl.ac.uk; bertoldi@seas.harvard.edu
A. E. Forte, E. Zari
Department of Electronics
Information and Bioengineering
Politecnico di Milano
Milan 20133, Italy
A. E. Forte
Department of Engineering
King's College London
London WC2R 2LS, UK
P. Z. Hanakata
Department of Physics
Harvard University
Cambridge, MA 02138, USA
L. Sumner
Independent researcher
The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/adfm.202111610.

DOI: 10.1002/adfm.202111610

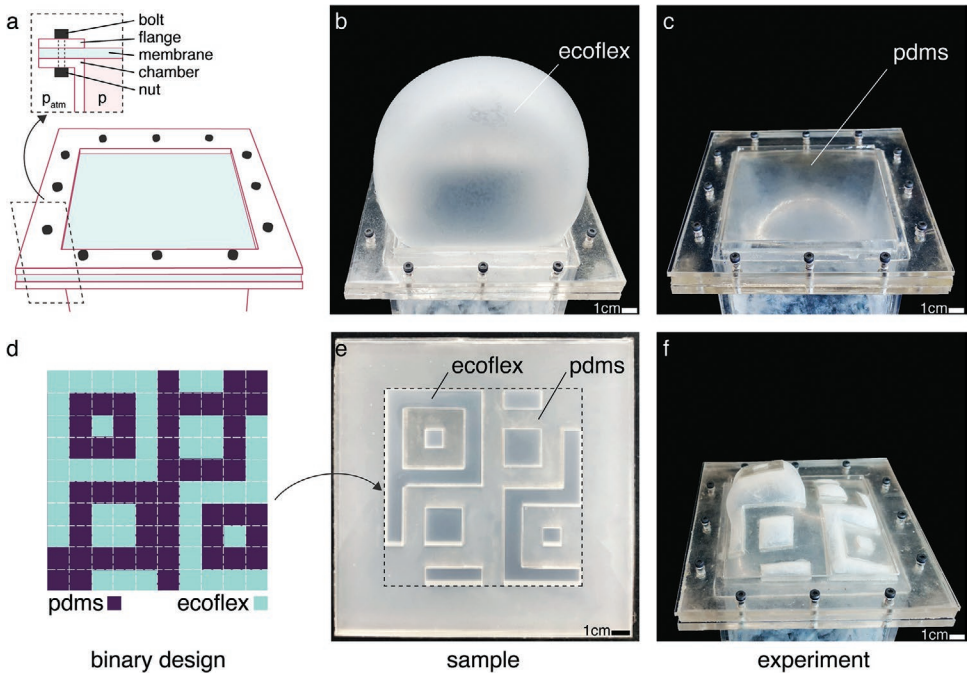

Figure 1. Bi-material membrane unlocks complex deformations. a) Inflation set up: the membrane (in cyan) is clamped between the acrylic chamber's edge underneath and the square flange aligned on top, using bolt and nuts. b) A flat membrane made out of Ecoflex morphs in a simple spherical shape upon inflation. c) If the same membrane is made out of a stiffer material (i.e. PDMS), the deformation is simply scaled down. d) We use a regular grid to divide the membrane's domain into subdomains named pixels, to which either material can be assigned, creating a binary design. e) The membrane can be fabricated as a continuous material. f) Upon inflation, the bi-material membrane assumes complex deformation, depending on the mutual position of stiff and soft pixels.

is pressurized (**Figure 1**a). As expected, when pressurized, the elastomeric membrane deforms out-of-plane achieving a dome-like shape with a height that depends both on the stiffness of the material and the thickness of the sheet. For example, under a pressure p = 1.7 KPa, a membrane made out of Ecoflex (Ecoflex 00-30 with initial shear modulus μ_{Eco} = 0.023 MPa) with thickness h_{Eco} = 1 mm and initial flexural rigidity of $D_{Eco} = \frac{E_{Eco}h_{Eco}^3}{12(1-\nu_{Eco}^2)} = 7.67\times10^{-6}$ Pa·m^3 (where $E_{Eco} = 2\mu_{Eco}(1+\nu_{Eco})$ and ν_{Eco} = 0.5 are the Young's modulus and Poisson's ratio of Ecoflex) reaches a height of about 9 cm (Figure 1b). Differently, a membrane with thickness h_{PDMS} = 7 mm made out of PDMS (SYLGARD 184 with initial shear modulus μ_{PDMS} = 0.85 MPa), for which the initial flexural rigidity is $D_{PDMS} = 9.72\times10^{-2}$ Pa·m^3, reaches a height about 1 cm when subjected to p = 10 KPa (Figure 1c).

Whereas homogeneous membranes always lead to dome-like shapes, it has been shown that the range of achievable shapes for both elastomeric membranes and tubes can be enriched by incorporating stiffer components, such as fibers and sheets.[6,16,39–41] Therefore, to achieve more complex configurations, we realize our membranes out of a combination of stiff and flexible pixels. To demonstrate the concept, we partition the membrane with a 10×10 array of squares (all with edges of 1 cm) and assign to each pixel either a 7 mm thick layer of PDMS or a 1 mm thick layer of Ecoflex (Figure 1d). Such membranes are manufactured using a multistep molding procedure. First, we create a mold (with depth of 7 mm) with the negative shape of the stiff pixels of the binary design and fill it with PDMS. Then, before the PDMS is completely cured (after 1.5 h), we remove 1 mm acrylic sheet from the mold to leave behind a 1 mm deep pocket corresponding to the soft pixels that we fill with Ecoflex (see Video S1, Supporting Information). Note that after curing a continuous membrane is obtained, due to the fact that the two elastomeric networks bind upon contact while curing (Section S1, Supporting Information Appendix) . In Figure 1e, we show the fabricated membrane corresponding to the binary design reported in Figure1d. As shown in Figure 1f for p = 5 kPa, upon inflation, this membrane undergoes a complex transformation, bulging out of plane in a nonintuitive fashion. The non-linear behaviour of such a membrane is strongly governed by the location of the soft and stiff pixels on the square grid and their interactions. As such, understanding the relation between binary design and the resultant 3D inflated shape is non-trivial and requires an efficient inverse design strategy.

3. Inverse Design via Neural Networks

Inspired by recent works that have successfully employed machine learning methods to inverse-design complex physical systems, including mechanical[26,42] and optical[43] metamaterials as well as chemical compounds,[44,45] we employ fully connected NNs to identify binary design of pixelated membranes and associated pressure levels at which such membranes should be inflated to reach a target 3D shape.

To efficiently generate the large amounts of data for the NNs to be trained on, we conduct non-linear FE simulations within

ABAQUS 2019/Standard. In all our simulations, we discretize the PDMS and Ecoflex pixels with four-node general-purpose shell elements (S4R element type) and four-node membrane elements (M3D4 element type), respectively. Further, we i) model the response of both elastomers using an incompressible Gent material, ii) fix all nodes located on the four edges of the models, iii) apply a pressure p (with $p \in [0, 3.5]$ kPa) directly on the bottom surface, and iv) solve for the deformation using the dynamic implicit solver, while monitoring the kinetic energy to ensure quasi-static conditions. We then export the deformed configurations of the membrane at p =1.5, 2.5, and 3.5 kPa and use the method of voxelization[46] to represent them. Specifically, we start with a rectangular cube with dimensions $15 \times 15 \times 15$ cm, which contains all inflated membranes for the three pressure levels considered and split the domain into smaller cubes known as voxels (**Figure 2**a). We then assign a value of 1 if the voxel's centroid falls below the inflated membrane (Figure 2a, internal pixels), and 0 otherwise (Figure 2a, external pixels). Since the height of the majority of the inflated designs is lower than 8 cm, we use voxels with dimensions $1 \times 1 \times 0.25$ cm to cover the height up to $z = 8$ cm and $1 \times 1 \times 1.5$ cm for $z > 8$ cm (Figure 2a). Further, since the corners of the considered rectangular cube are never reached by the membranes upon inflation, to reduce the size of our domain, we remove the corresponding voxels from the analysis, resulting in 9220 voxels.

Next, we flatten our 10×10 binary designs onto a 100-dimensional vector $\mathbf{X}$ containing 0s and 1s in correspondence to the soft and stiff pixels, respectively, and the numerically obtained inflated shape at the three considered levels of pressure onto three 9220-dimensional vectors $\mathbf{Y}$s containing 0s and 1s in correspondence to the internal and external voxels, respectively (Figure 2b). We then train the NNs to perform a mapping $\mathbf{Y} \rightarrow \overline{p} \oplus \overline{\mathbf{X}}$, where the overbar is used to represent the quantities predicted by the NNs and $\oplus$ denotes concatenation (Figure 2c). In this study, we use fully connected NNs with two hidden layers having identical number of neurons, N_{neu}, and iteratively update the neuron weights and biases to make the output conform to the true $\mathbf{X}$ and p by minimizing[26]

$$\mathcal{L}_{NN} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} | \mathbf{X}_i - \overline{\mathbf{X}}_i |^2 + \lambda | p_i - \overline{p}_i |^2 \quad (1)$$

where N_{train} is the number of training datapoints and λ is an adjustable hyperparameter that controls the relative weight between the mean squared distance in $\mathbf{X}$ and p. Note that, in order to obtain binary values for $\overline{\mathbf{X}}$, we employ a sigmoid function as an output filter on the output layer. Additional details about the NNs' hyperparameters (e.g., learning rate) are reported in Section S5, Supporting Information Appendix.

In order to obtain accurate predictions in the inverse design problem, the NNs need i) to have enough training data to capture the whole design space and ii) an optimized architecture (i.e., optimal N_{neu} and λ). The most common strategy to generate large training datasets is based on randomly generated data (in our case, this translates to generating pixelated membrane designs where the locations for the soft and stiff pixels are randomly assigned).[25,28] However, this approach would require an extremely large number of simulations, since it would be

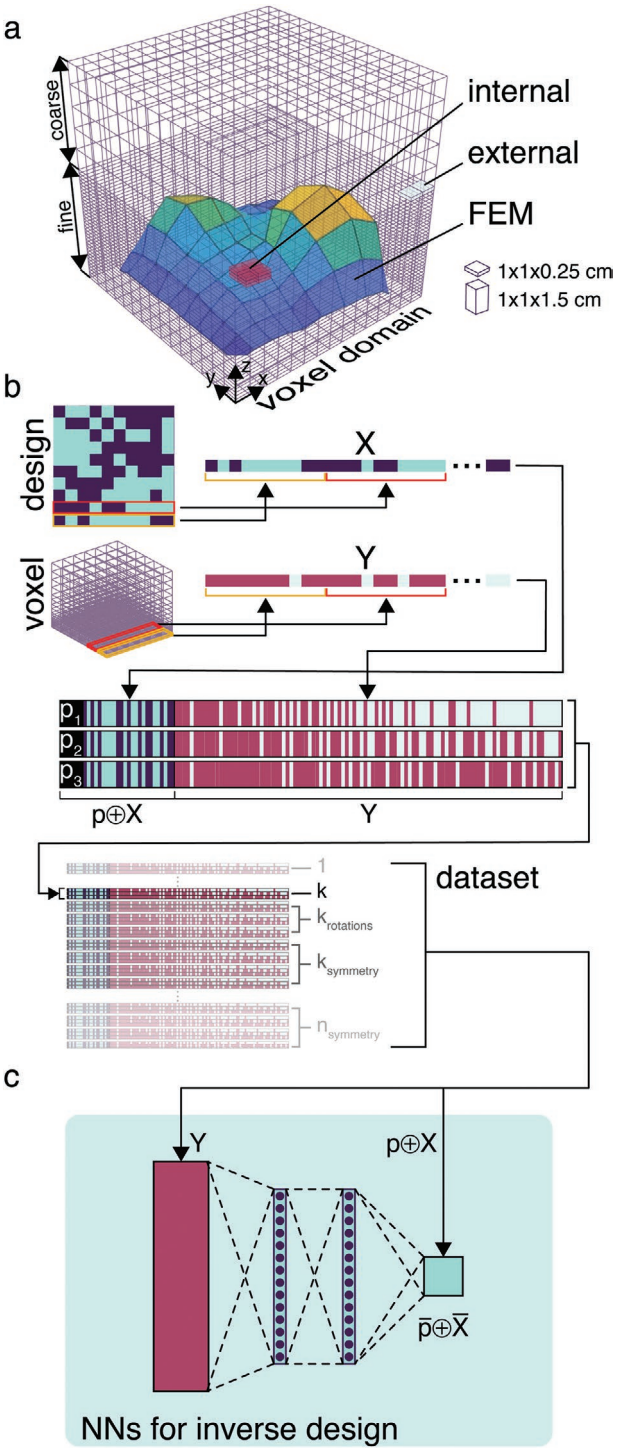

Figure 2. Data voxelization, processing, and structuring. a) The inflated shape from each FEM simulation is mapped into a voxelated domain comprising a fine and a coarse region. The voxels whose centroid falls below the membrane are identified as internal and external otherwise. b) The membrane's binary design and the voxel domain are flattened into two 1D binary arrays named $\mathbf{X}$ and $\mathbf{Y}$, respectively, and concatenated with the corresponding pressure level (p_1, p_2 or p_3), producing three arrays per simulation. The binary design can be rotated and mirrored augmenting the number of datapoints eightfold. c) The $\mathbf{Y}$ and $p \oplus \mathbf{X}$ matrices are fed to the NNs in the input and output layers, respectively.

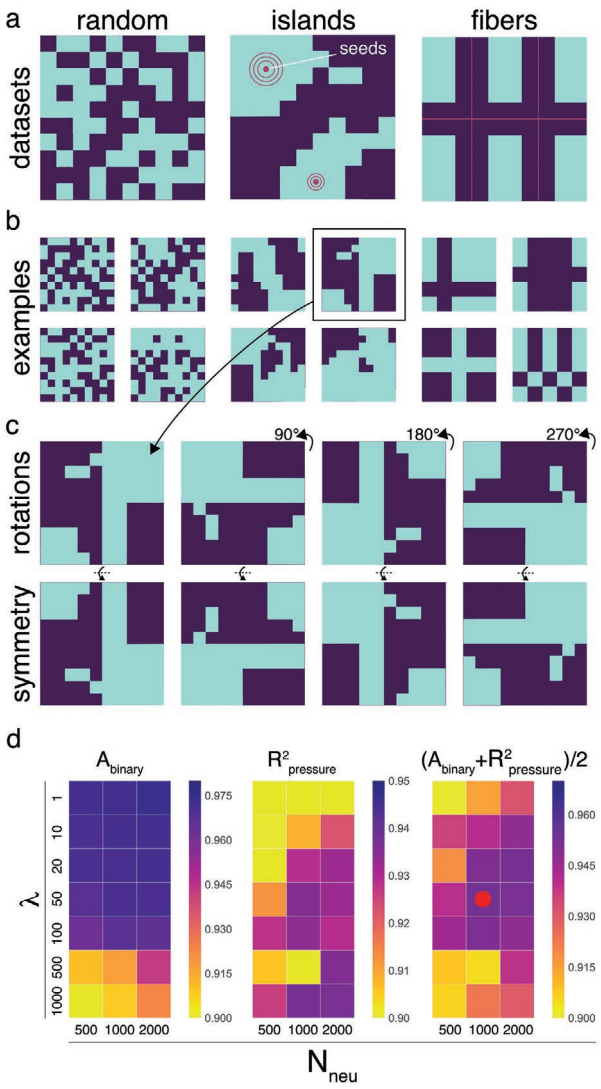

Figure 3. Dataset classes and hyperparameter search. a) To train the NNs, we use three classes of designs: random, islands, and fibers. b) Examples from the three classes are reported to appreciate their topological characteristics. c) Each design can be rotated and mirrored to generate eight different datapoints. d) Effect of the number of neurons in the NNs layers, N_{neu}, and the hyperparameter λ on A_{binary}, $R^2_{pressure}$, and $(A_{binary} + R^2_{pressure})/2$. The red marker indicates the combination of these two parameters that gives the best average accuracy between the two metrics ($(A_{binary} + R^2_{pressure})/2$).

rare to sample data points containing large clusters of soft/stiff pixels or lines of soft/stiff pixels that are known to have a profound effect on the mechanical behavior of the membranes.[6] Therefore, guided by previous studies showing that the performances of NNs improve when the model is trained with diverse (non-redundant) datasets,[47–52] we adopt three different strategies to generate 2D pixelated binary designs whose inflated shapes predicted via FE are used to train the NNs. Specifically, we simulate via FE the behavior upon inflation of i) 2500 membranes in which we randomly assign a value of 0 or 1 to each pixel (**Figure 3**a, left); ii) 2500 membranes in which a few (1–15) pixels ("seeds") located at random positions are allowed to grow in all directions and convert the neighbor pixels from stiff to soft until a critical ratio of soft pixels to all pixels is reached,[38] effectively creating "islands" of soft pixels (Figure 3a, center); iii) 2500 membranes realized employing logical operators to combine row and column vectors of stiff pixels running from and to opposite edges, to create features resembling fibers (Figure 3a, right). We refer to these three datasets as i) random, ii) islands, and iii) fibers and report design examples for each dataset in Figure 3b. Note that, although for each dataset we simulate 2500 membranes, we are able to create a total of 60 000 datapoints per dataset since i) each of these designs can be rotated and mirrored seven times (see Figure 3c) and ii) for each of them, we export the inflated configuration at three different levels of pressure. Additional details about the algorithms used to generate the pixelated membranes are reported in Section S4, Supporting Information Appendix.

To optimize our NNs, we vary the number of neurons N_{neu} and the hyperparameter λ. We first merge the generated 180 000 datapoints into a single dataset. We then use 80% of such dataset to run multiple training sessions, and for each session, we vary the number of neurons N_{neu} and the hyperparameter λ. To attest the performance of each trained model, we use 10% of the datapoints as validation set and the remaining as test set, and introduce two metrics: an accuracy on the predicted binary design, A_{binary}, and an accuracy on the predicted pressure level, $R^2_{pressure}$. Specifically, A_{binary} is evaluated by counting the number of correctly identified pixels and therefore defined as

$$A_{binary} = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \frac{N^i_{correct}}{N_{pixels}} \tag{2}$$

where $N^i_{correct}$ is the total number of correctly predicted pixels for the i-th binary design, N_{pixels} is the total number of pixels in the membrane (i.e., 100), and N_{test} is the number of datapoints used for testing (i.e., 18 000). Differently, since the pressure is a continuous variable, we define the accuracy on the predicted pressure level $\overline{p}$ as

$$R^2_{pressure} = 1 - \frac{\sum_{i=1}^{N_{test}} | p_i - \overline{p}_i |^2}{\sum_{i=1}^{N_{test}} | p_i - \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} p_i |^2} \tag{3}$$

By systematically investigating the effect of λ and N_{neu} on the two metrics, we find that larger values for N_{neu} generally lead to a better test accuracy as the model capacity is increased (Figure 3d). Note that, since increasing the number of neurons often leads to overfitting,[53–55] we use early stopping rules[53] to determine how many iterations can be run before the NNs begin to over-fit. An increase in λ results in larger $R^2_{pressure}$, but lower A_{binary} (Figure 3d). Hence, we use the average accuracy between the two metrics, $(A_{binary} + R^2_{pressure})/2$, to determine the optimized NN architecture, which we find to be characterized by N_{neu} = 1000 and λ = 50–values that are fixed for the next analyses. Additionally, it is worth noticing that the NNs trained on a combination of all datapoints outperform the same model trained on a single class of data, confirming that a diverse dataset leads to better performances (more information on NNs' performances with different training are reported in Section 5, Supporting Information Appendix).

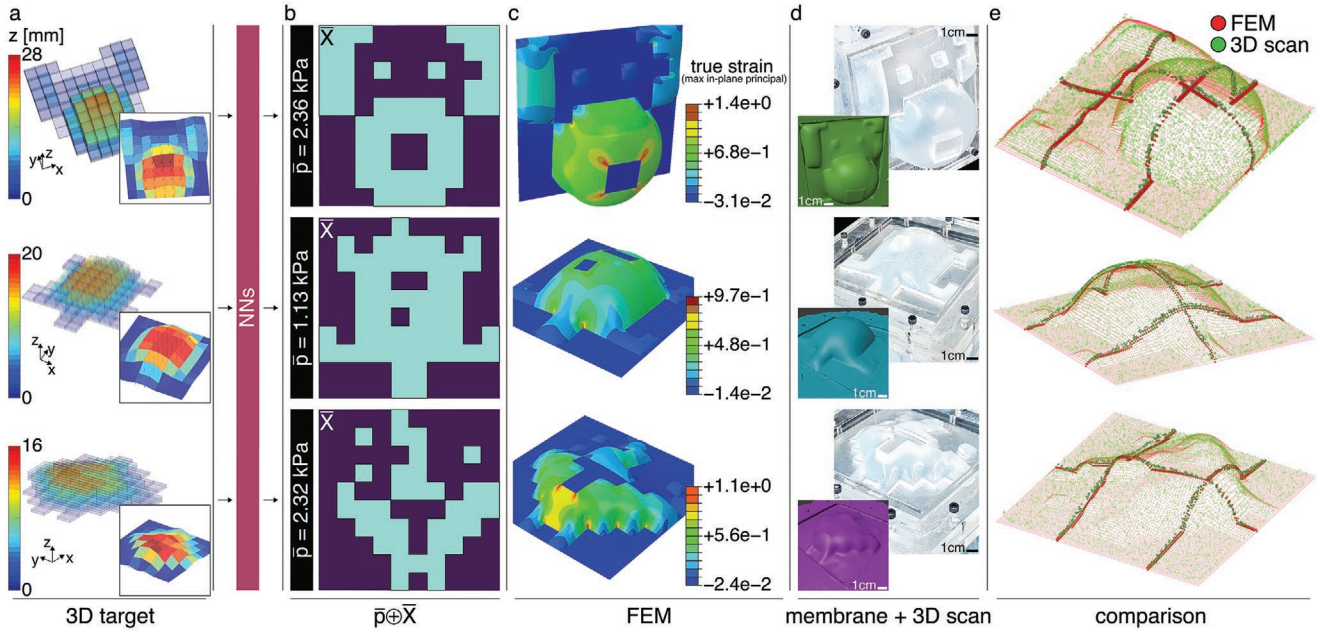

Figure 4. Inverse design of target 3D shapes. a) Target 3D shapes that resemble a dog face (top), a turtle (center), and a manta ray (bottom) are fed in the NNs. b) The NNs provide optimal inflation pressure, $\bar{p}$, and binary designs, $\bar{X}$, as outputs. c) The binary designs are inflated at the corresponding pressure though FE. The colors indicate maximum in-plane principal true strains. d) The designs are fabricated, inflated at the corresponding pressure, and 3D-scanned. e) The clouds of points from the FE simulations (red dots) and the 3D-scans (green dots) are overlapped and compared. Solid markers are highlighted along cutting planes to better show the overlapping.

4. Inverse Design of Soft Membranes

Having verified the accuracy of the NNs on a test set of unseen designs generated through three different algorithms, we then employ them to inverse-design target 3D shapes. Specifically, we feed a 3D shape as input to our trained NNs and obtain as output a 2D binary design for the soft membrane, along with the pressure necessary to reach the target shape upon inflation. To demonstrate the process, we select shapes that resemble a dog face, a turtle, and a manta ray (**Figure 4**a). Each target shape is flattened onto a 9220-dimensional vector, $\mathbf{Y}^{target}$, which is fed into the NNs. As output, for each design we obtain a pixelated membrane design $\bar{\mathbf{X}}$ and inflation pressure $\bar{p}$ (Figure 4b).

To measure the accuracy of the designs identified by the NNs, we start by employing FE to simulate the inflation of the binary design $\bar{\mathbf{X}}$ until the pressure reaches $\bar{p}$. As shown in Figure 4c, the numerically obtained inflated shapes qualitatively match the 3D target ones. To better quantify the similarity between the two sets of shapes, we flatten each numerically obtained shape into a vector $\mathbf{Y}^{FE}$ and compare all its internal voxels to the corresponding one in $\mathbf{Y}^{target}$. We then use the ratio between the number of voxels correctly predicted, N_{correct}, and the number of internal voxels in the target shape, N_{target} as the accuracy metric

$$A_{membrane} = \frac{N_{correct}}{N_{target}} \tag{4}$$

Using Equation (4), we obtain $A_{\text{membrane}} = 0.975, 0.945, 0.996$ for the dog face, turtle, and manta ray, respectively. Such high accuracy values are indicative of effective NNs that provide pixelated designs and pressures leading to 3D shapes extremely close to those targeted.

Next, we physically fabricate the designs identified by the NNs. In Figure 4d, we report snapshots of the physical membranes inflated to the optimal pressure provided by the NNs. As is noticeable, despite the unavoidable imperfections introduced during fabrication and testing, the 3D shapes obtained upon inflation are clearly recognizable. To compare such shapes to those obtained via FE simulations, we use a hand-held 3D scanner (Artec Space Spider, Artec Studio 14.1.1.75) and record the experimentally obtained surface profiles at $\bar{p}$ (insets in Figure 4d). As shown in Figure 4e, we find excellent agreement between the numerically predicted and experimentally obtained inflated shapes, confirming the validity of our approach: (more information on the experiments are reported in Section S2 and S3, Supporting Information Appendix).

Having demonstrated that our NNs, trained with a combination of three different datasets, can be used to identify soft membranes capable of mimicking target 3D shapes upon inflation, we then explore how these can be harnessed for applications. Specifically, since it is known that the application of compressive loading around a wound site can reduce healing time and formation of hypertrophic scars,[56,57] we design soft membranes that apply pressure in targeted areas while avoiding contact with sensitive locations.

To demonstrate our approach, we focus on the two scar profiles highlighted in orange in **Figure 5**a and aim at realizing membranes that upon inflation have their maximum elevation (along the z direction) in the areas surrounding the scars and minimum elevation in the areas where the scars lie. We expect such membranes to apply compressive loading to the region

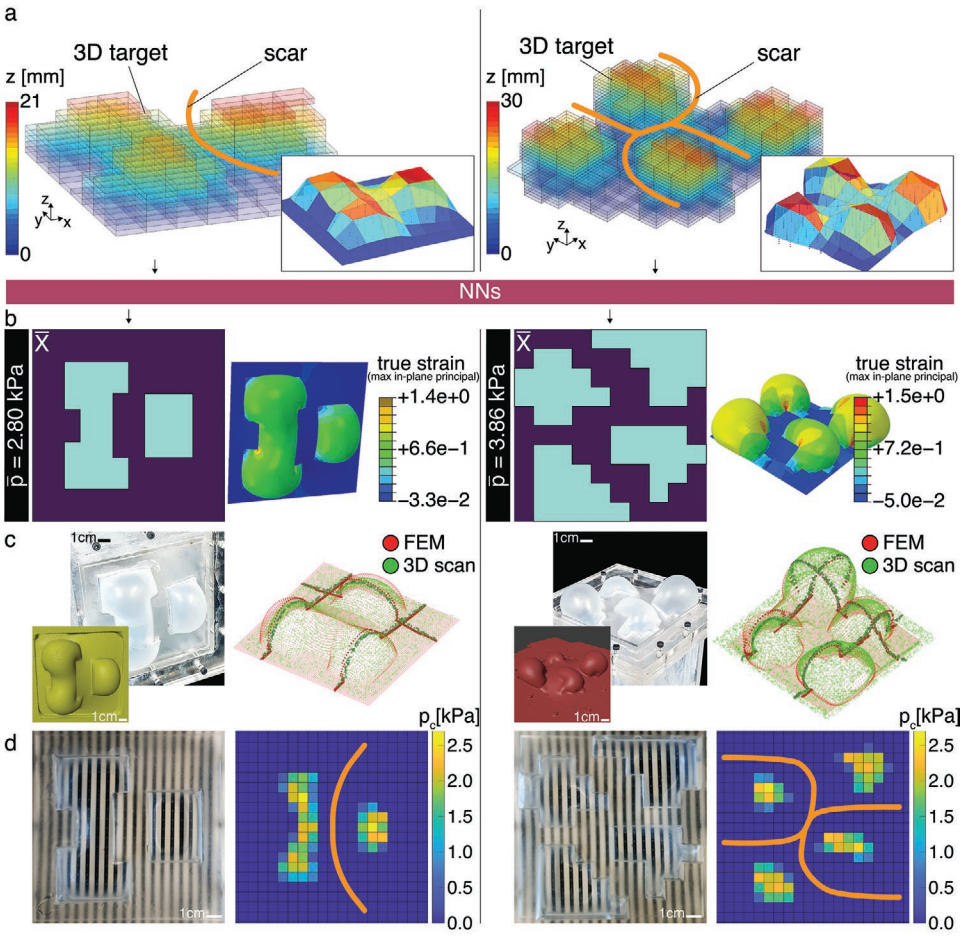

Figure 5. Inverse design of target 3D shapes for mechanotherapy: a) Target 3D-shapes that can stimulate the tissue around pre-defined scar profiles during inflation are fed in the NNs. b) The NNs provide optimal inflation pressure, $\overline{p}$, and binary designs, $\overline{X}$, as outputs, which are inflated at the corresponding pressure though FE. Numerical snapshots of the inflated membranes are shown, with the color indicating the maximum in-plane principal true strain. c) The designs are fabricated, inflated at the corresponding pressure, and 3D-scanned. The clouds of points from the FE simulations (red dots) and the 3D-scans (green dots) are overlapped and compared. Solid markers are highlighted along cutting planes to better show the overlapping. d) The membranes are fixed upside down at a 15 mm distance from a pressure mat and imaged from the top through a transparent pressure chamber. The measured contact pressures are reported, along with their locations and the pre-defined scar profile.

around the wound when inflated against the skin and, therefore, to promote healing. To obtain pixelated membrane designs resulting in the target shapes shown in Figure 5a upon inflation, we flatten their voxelated shapes onto 9220-dimensional vectors Y^{target} and feed them into the trained NNs. As for the membranes shown in Figure 3, we then use FE to simulate the behavior of the designs identified by our NNs and find very good agreement between the target and numerically simulated inflated shapes (Figure 5b) testified by $A_{membrane}$=0.949 and 0.932 for the *c* and χ-scars, respectively. Further, we build the physical membranes, and also in this case found that they nicely match the target shapes (Figure 5c). Finally, to evaluate the pressure locally applied by the inflated membranes around the two considered scars, we position the deflated membranes at 15 mm from a pressure mat (Tekscan - Model 5250). When inflated at $\overline{p}$, the membranes come in contact with the pressure mat which records the locally applied contact pressure, p_c. In Figure 5d, we show the top view of the inflated membrane pushing against the mat (visible through the transparent pressure chamber) as well as the recorded pressure distribution overlaid with the scar profiles. As clearly visible, the inflated 3D shapes optimized through our NNs are able to apply pressure around the prescribed areas without touching the scars.

5. Conclusion

In summary, to realize membranes that can morph into pre-programmed shapes upon inflation, we have employed NNs that are trained to identify a pixelated membrane design and inflation pressure leading to the desired 3D shape. The data required to train the NNs were obtained by simulating the membrane inflation through FE, and to guarantee the creation of a diverse dataset, three different algorithms have been used to produce pixelated designs. We have then employed our trained NNs to inverse-design a few user generated 3D shapes and showed how such platform could be used to create patient-specific devices for mechanotherapy routines where it

is important to stimulate the tissue around prescribed areas (scars) to enhance healing and reduce recovery time.

Despite having presented results at the centimeter scale, our methodology is scale independent and can benefit a range of applications where having an inverse-design strategy could facilitate and improve the design process itself. Examples might include ergonomic designing, patient-specific medical devices, architectural components, and shape-morphing acoustic devices. Additionally, we have shown that our NNs trained on only 7500 forward FEM simulations can successfully solve an inverse problem with 2^{100} possible designs. This reinforces previous findings which identified machine learning methods as a valuable complementary tool to established mechanical approa ches.[25–27,29–31,36] The performance of our model can be further improved by applying convolutional neural networks (CNNs) as the filters and pooling layers are efficient in capturing spatial correlation and locality in a sparse data.[25] In particular, 3D CNNs, which are widely used for point cloud labeling in computer vision,[58,59] would be ideal to handle the voxels of arbitrary 3D shapes. Moreover, recent deep learning methods such as conditional generative adversarial neural networks, with image-to-image translation capabilities,[60–62] could also be employed to solve inverse-design problems similar to the one hereby described.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

This research was supported by the NSF grants DMR-2011754, DMREF-1922321, and OAC-2118201. P.Z.H. acknowledges support through the NSF grant DMR-1608501. P.Z.H. thanks Ekin D. Cubuk for helpful discussions.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

Keywords

inverse design, machine learning, membranes, shape morphing, soft matter

Received: November 14, 2021
Revised: December 9, 2021
Published online:

[1] S. J. Callens, A. A. Zadpoor, *Mater. Today* **2018**, *21*, 241.
[2] G. P. Choi, L. H. Dudte, L. Mahadevan, *Nat. Mater.* **2019**, *18*, 999.
[3] E. Siéfert, E. Reyssat, J. Bico, B. Roman, *Nat. Mater.* **2019**, *18*, 24.
[4] J. W. Boley, W. M. van Rees, C. Lissandrello, M. N. Horenstein, R. L. Truby, A. Kotikian, J. A. Lewis, L. Mahadevan, *Proc. Natl. Acad. Sci. U. S. A.* **2019**, *116*, 20856.
[5] L. Tomholt, O. Geletina, J. Alvarenga, A. V. Shneidman, J. C. Weaver, M. C. Fernandes, S. A. Mota, M. Bechthold, J. Aizenberg, *Energy Build.* **2020**, *226*, 110377.
[6] J. Pikul, S. Li, H. Bai, R. Hanlon, I. Cohen, R. Shepherd, *Science* **2017**, *358*, 210.
[7] L. H. Dudte, E. Vouga, T. Tachi, L. Mahadevan, *Nat. Mater.* **2016**, *15*, 583.
[8] L. H. Dudte, G. P. Choi, L. Mahadevan, *Proc. Natl. Acad. Sci. USA* **2021**, *118*, 21.
[9] S. Felton, M. Tolley, E. Demaine, D. Rus, R. Wood, *Science* **2014**, *345*, 644.
[10] J. Kim, J. A. Hanna, M. Byun, C. D. Santangelo, R. C. Hayward, *Science* **2012**, *335*, 1201.
[11] E. Siéfert, E. Reyssat, J. Bico, B. Roman, *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 16692.
[12] E. Siéfert, J. Bico, E. Reyssat, B. Roman, *J. Mech. Phys. Solids* **2020**, *143*, 104068.
[13] T. Gao, E. Siéfert, A. DeSimone, B. Roman, *Adv. Mater.* **2020**, *32*, 2004515.
[14] E. Siéfert, E. Reyssat, J. Bico, B. Roman, *Soft Matter* **2020**, *16*, 7898.
[15] M. Skouras, B. Thomaszewski, B. Bickel, M. Gross, in *Computer Graphics Forum*, Vol. 31, Wiley Online Library **2012**, pp. 835–844.
[16] L. Jin, A. E. Forte, B. Deng, A. Rafsanjani, K. Bertoldi, *Adv. Mater.* **2020**, *32*, 2001863.
[17] F. Connolly, C. J. Walsh, K. Bertoldi, *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 51.
[18] L. H. Dudte, E. Vouga, T. Tachi, L. Mahadevan, *Nat. Mater.* **2016**, *15*, 583.
[19] M. Konaković, K. Crane, B. Deng, S. Bouaziz, D. Piker, M. Pauly, *ACM Transactions on Graphics (TOG)* **2016**, *35*, 89.
[20] A. S. Gladman, E. A. Matsumoto, R. G. Nuzzo, L. Mahadevan, J. A. Lewis, *Nat. Mater.* **2016**, *15*, 413.
[21] J. Panetta, F. Isvoranu, T. Chen, E. Siéfert, B. Roman, M. Pauly, *ACM Transactions on Graphics (TOG)* **2021**, *40*, 39.
[22] R. Bouzidi, Y. Lecieux, *Acta Astronaut.* **2012**, *74*, 69.
[23] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, *Neural Computation* **1989**, *1*, 541.
[24] Y. Goldberg, *Journal of Artificial Intelligence Research* **2016**, *57*, 345.
[25] P. Z. Hanakata, E. D. Cubuk, D. K. Campbell, H. S. Park, *Phys. Rev. Lett.* **2018**, *121*, 255304.
[26] P. Z. Hanakata, E. D. Cubuk, D. K. Campbell, H. S. Park, *Physical Review Research* **2020**, *2*, 042006.
[27] M. A. Bessa, P. Glowacki, M. Houlder, *Adv. Mater.* **2019**, *31*, 1904845.
[28] G. X. Gu, C.-T. Chen, D. J. Richmond, M. J. Buehler, *Mater. Horiz.* **2018**, *5*, 939.
[29] C. Yang, Y. Kim, S. Ryu, G. X. Gu, *Mater. Des.* **2020**, *189*, 108509.
[30] M. Mozaffar, R. Bostanabad, W. Chen, K. Ehmann, J. Cao, M. Bessa, *Proc. Natl. Acad. Sci. U. S. A.* **2019**, *116*, 26414.
[31] Z. Yang, C.-H. Yu, M. J. Buehler, *Sci. Adv.* **2021**, *7*, eabd7416.
[32] J. K. Wilt, C. Yang, G. X. Gu, *Adv. Eng. Mater.* **2020**, *22*, 1901266.
[33] E. Haghighat, M. Raissi, A. Moure, H. Gomez, R. Juanes, A deep learning framework for solution and discovery in solid mechanics, **2020**.
[34] D. W. Abueidda, M. Almasri, R. Ammourah, U. Ravaioli, I. M. Jasiuk, N. A. Sobh, *Compos. Struct.* **2019**, *227*, 111264.
[35] C.-T. Chen, G. X. Gu, *Advanced Theory and Simulations* **2019**, *2*, 1900056.

[36] C. Ma, Z. Zhang, B. Luce, S. Pusateri, B. Xie, M. H. Rafiei, N. Hu, *npj Computational Materials* **2020**, *6*, 40.
[37] M. Bessa, S. Pellegrino, *Int. J. Solids Struct.* **2018**, *139*, 174.
[38] Y. Mao, Q. He, X. Zhao, *Sci. Adv.* **2020**, *6*, eaaz4169.
[39] F. Connolly, P. Polygerinos, C. J. Walsh, K. Bertoldi, *Soft Rob.* **2015**, *2*, 26.
[40] F. Connolly, C. J. Walsh, K. Bertoldi, *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 51.
[41] P. Polygerinos, Z. Wang, J. T. Overvelde, K. C. Galloway, R. J. Wood, K. Bertoldi, C. J. Walsh, *IEEE Transactions on Robotics* **2015**, *31*, 778.
[42] S. Kumar, S. Tan, L. Zheng, D. M. Kochmann, *npj Computational Materials* **2020**, *6*, 73.
[43] Z. Liu, D. Zhu, S. P. Rodrigues, K.-T. Lee, W. Cai, *Nano Lett.* **2018**, *18*, 6570.
[44] R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams, A. Aspuru-Guzik, *ACS central science* **2018**, *4*, 268.
[45] E. Putin, A. Asadulaev, Y. Ivanenkov, V. Aladinskiy, B. Sanchez-Lengeling, A. Aspuru-Guzik, A. Zhavoronkov, *J. Chem. Inf. Model.* **2018**, *58*, 1194.
[46] A. Kaufman, D. Cohen, R. Yagel, *Computer* **1993**, *26*, 51.
[47] P. Y. Simard, D. Steinkraus, J. C. Platt, et al., in *Icdar*, vol. 3, Citeseer **2003**.
[48] A. Torralba, A. A. Efros, in *CVPR 2011*, IEEE, Piscataway, NJ **2011**, pp. 1521–1528.
[49] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q. V. Le, in *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2019**, pp. 113–123.
[50] K. Vodrahalli, K. Li, J. Malik, *arXiv preprint arXiv:1811.12569* **2018**.
[51] V. Birodkar, H. Mobahi, S. Bengio, *arXiv preprint arXiv:1901.11409* **2019**.
[52] S. Paul, J. H. Bappy, A. K. Roy-Chowdhury, in *2016 IEEE International Conference on Image Processing (ICIP)*, IEEE, Piscataway, NJ **2016**, pp. 494–498.
[53] I. V. Tetko, D. J. Livingstone, A. I. Luik, *Journal of Chemical Information and Computer Sciences* **1995**, *35*, 826.
[54] C. M. Bishop, *Machine Learning* **2006**, *128*, 9.
[55] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, *Journal of Machine Learning Research* **2014**, *15*, 1929.
[56] L. A. Barnes, C. D. Marshall, T. Leavitt, M. S. Hu, A. L. Moore, J. G. Gonzalez, M. T. Longaker, G. C. Gurtner, *Advances in Wound Care* **2018**, *7*, 47.
[57] G. C. Gurtner, R. H. Dauskardt, V. W. Wong, K. A. Bhatt, K. Wu, I. N. Vial, K. Padois, J. M. Korman, M. T. Longaker, *Ann. Surg.* **2011**, *254*, 217.
[58] D. Maturana, S. Scherer, in *2015 IEEE Int. Conf. on Robotics and Automation (ICRA)*, IEEE, Piscataway, NJ **2015**, pp. 3471–3478.
[59] J. Huang, S. You, in *2016 23rd Int. Conf. on Pattern Recognition (ICPR)*, IEEE, Piscataway, NJ **2016**, pp. 2670–2675.
[60] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, J. Choo, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2018**, pp. 8789–8797.
[61] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2017**, pp. 1125–1134.
[62] M. Mirza, S. Osindero, *arXiv preprint arXiv:1411.1784* **2014**.

ADVANCED FUNCTIONAL MATERIALS

Supporting Information

for *Adv. Funct. Mater.,* DOI: 10.1002/adfm.202111610

Inverse Design of Inflatable Soft Membranes Through Machine Learning

Antonio Elia Forte, Paul Z. Hanakata, Lishuai Jin, Emilia Zari, Ahmad Zareei, Matheus C. Fernandes, Laura Sumner, Jonathan Alvarez, and Katia Bertoldi**

Inverse design of inflatable soft membranes through machine learning

*Antonio Elia Forte Paul Z. Hanakata Lishuai Jin Emilia Zari Ahmad Zareei Matheus C. Fernandes Laura Sumner Jonathan Alvarez Katia Bertoldi**

Dr A. E. Forte, Dr L. Jin, E. Zari, Dr A. Zareei, Dr M. C. Fernandes, J Alvarez, Prof K Bertoldi
J.A. Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02138, USA
bertoldi@seas.harvard.edu
Dr A. E. Forte, E. Zari
Department of Electronics, Information and Bioengineering, Politecnico di Milano, Milan, 20133 Italy
Dr A. E. Forte,
Department of Engineering, King's College London, London, WC2R 2LS, UK
Dr P. Z. Hanakata
Department of Physics, Harvard University, Cambridge, MA 02138, USA
Dr L Sumner
Independent researcher

1 Fabrication

As shown in Figure 1d in the main text, our soft membranes comprise a 10×10 array of squares (all with edges of 1 cm). Each of these can be either made out of stiff (PDMS) or soft (Ecoflex) silicone material. The stiff and soft pixels are 7 mm and 1 mm thick, respectively.
To demonstrate the concept, we partition the membrane with a 10×10 array of squares (all with edges of 1 cm) and assign to each pixel either a 7 mm thick layer of PDMS or a 1 mm thick layer of Ecoflex

1.1 Materials: Ecoflex 00-30 and Polydimethylsiloxane

Polydimethylsiloxane (PDMS, Sylgard® 184) and Ecoflex 00-30 (Smooth-On) are selected as materials to realize the stiff and soft pixels, respectively.
Ecoflex 00-30 comes in two parts that must be mixed 1A:1B by weight or volume and cured at room temperature for 4 hours. The technical properties of Ecoflex 00-30 (as provided by the vendor) are listed in Table 1.

Property	Unit	Result
Mixing Ratio		1A:1B
Color		Translucent
Mixed Viscosity	cPs	3000
Specific Gravity		1.07
Working Time at 25°C (Pot Life)	minutes	45
Cure Time at 25°C	hours	4
Cure Time at 65°C	minutes	10

Table 1: Ecoflex 00-30 properties.

PDMS Sylgard® 184 Silicone Elastomer is supplied as a two-part liquid kit consisting of a polymeric base (A) and a curing agent (B). The two components are mixed together with a 10A:1B mixing ratio which is recommended by the vendor. Technical information provided by the vendor for the PDMS are listed in Table 2. Note that PDMS can be cured at any temperature between room temperature and 200°C. The higher the temperature, the stiffer the resulting material and the shorter the curing time.

Temperatures exceeding 200°C will cause thermal decomposition of the material [1, 2]. At room temperature PDMS has a cure time of 48 hours. In order to speed up the PDMS curing time and the membrane fabrication procedure, we interpolated the data given by the vendor in the range of temperature 25°C - 150°C (see Table 2) by fitting it using the power law reported in Figure 1. For all our samples we choose a curing time of about 3 hours at 60°C.

Property	Unit	Result
Mixing Ratio		10A:1B
Color		Colorless
Mixed Viscosity	cP	3500
Specific Gravity		1.03
Working Time at 25°C (Pot Life)	hours	1.5
Cure Time at 25°C	hours	48
Cure Time at 100°C	minutes	35
Cure Time at 125°C	minutes	20
Cure Time at 150°C	minutes	10

Table 2: PDMS Sylgard ® 184 properties.

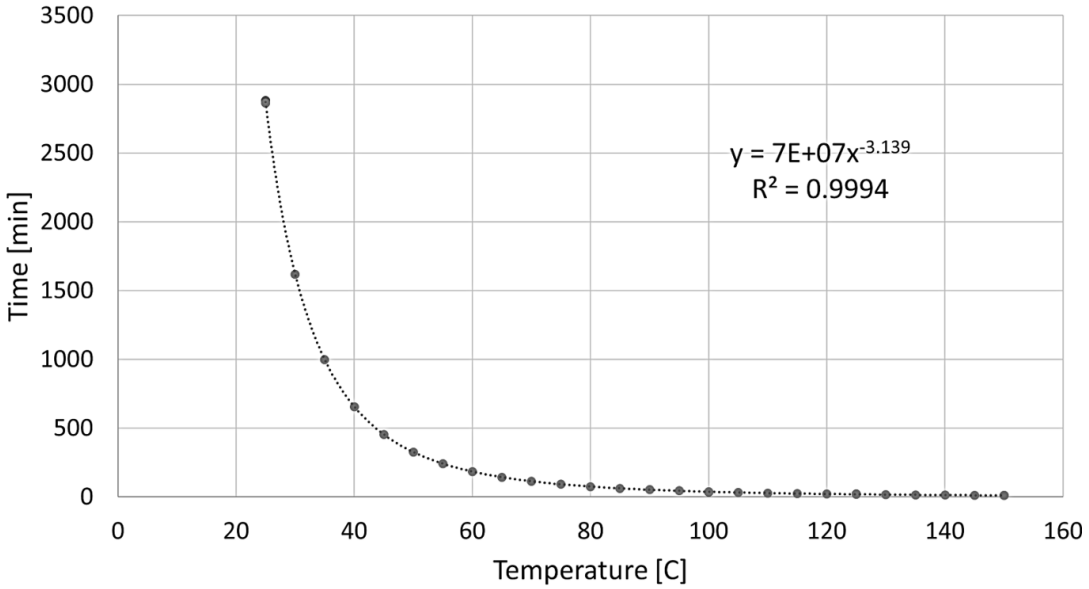

Figure 1: PDMS Cure Time vs Curing Temperature.

Finally, we note that when pouring PDMS into the acrylic mold air-bubble formation occurs which significantly increase the risk of material failure under stress (upon inflation). To remove the air-bubbles, a degassing procedure in vacuum is adopted:

1. Before curing, the PDMS sample is placed into the vacuum chamber.
2. The vacuum pump is turned on for about 4 minutes in order to create vacuum. After this time, a significant amount of air bubbles appear on the surface of the sample.
3. The pump is turned off and the vacuum valve is opened to bring the sample back to atmospheric pressure. The pressure shock bursts the air-bubbles.

Steps 2 and 3 are repeated 5 times, until all bubbles are removed from the sample. The whole process takes about 20 minutes.

1.2 Bonding between Ecoflex and PDMS

To ensure good bonding between the two materials, we investigate eight different curing procedures. All experimental tests are conducted using a dog-bone specimen (whose dimensions are reported in Figure 2), filled with both PDMS and Ecoflex. A release agent (Ease Release 200 - Smooth-On) is sprayed on the dog-bone acrylic mold before pouring the first material. The eight procedures are detailed below:

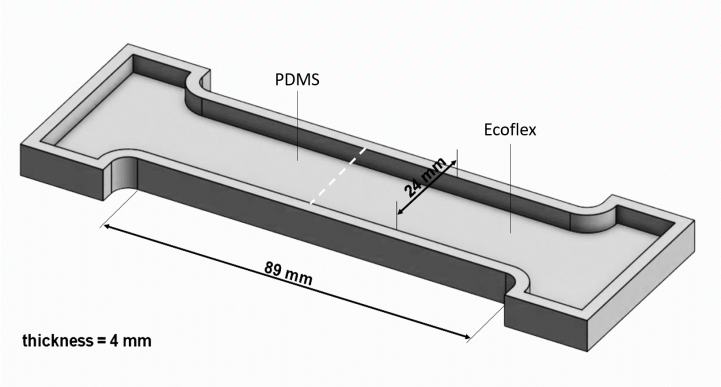

Figure 2: Dog-bone specimen dimensions.

The eight procedures are detailed below:

- Procedure 1: PDMS is poured and cured at 60°C for 3 hours. Ecoflex is poured and the specimen is cured at 25°C for 4 hours.
- Procedure 2: PDMS is poured and cured at 60°C for 2 hours and 15 minutes. Ecoflex is poured and the specimen is cured at 25°C for 1h. Then the specimen is cured at 60°C for 45 minutes to complete the PDMS cure.
- Procedure 3: PDMS is poured and cured at 60°C for 1 hour and 30 minutes. Ecoflex is poured and the specimen is cured at 25°C for 1h. Then the specimen is cured at 60°C for 1 hour and 30 minutes to complete the PDMS cure.
- Procedure 4: PDMS is poured and cured at 60°C for 45 minutes. Ecoflex is poured and the specimen is cured at 25°C for 1h. Then the specimen is cured at 60°C for 2 hours and 15 minutes to complete the PDMS cure.
- Procedure 5: Ecoflex is poured and cured at 25°C for 4 hours. PDMS is poured and the specimen is cured at 60°C for 3 hours.
- Procedure 6: Ecoflex is poured and cured at 25°C for 3 hours. PDMS is poured and the specimen is cured at 60°C for 3 hours.
- Procedure 7: Ecoflex is poured and cured at 25°C for 2 hours. PDMS is poured and the specimen is cured at 60°C for 3 hours.
- Procedure 8: Ecoflex is poured and cured at 25°C for 1 hour. PDMS is poured and the specimen is cured at 60°C for 3 hours.

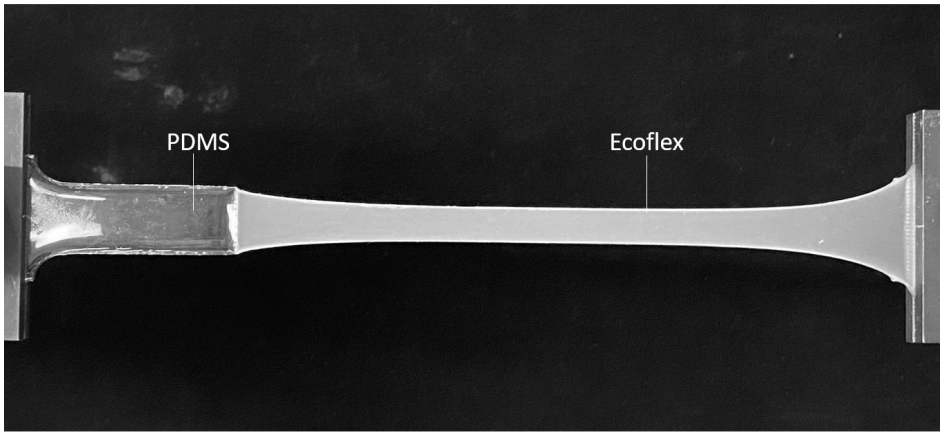

Figure 3: Uniaxial tensile test of a dog-bone specimen to determine the bond strength between PDMS and Ecoflex.

Cured specimens are carefully demolded and a tensile test is performed using an Instron 5969 Universal Test Machine (see Fig. 3). We test the samples until failure and investigate their maximum elongation. The collected experimental data is presented into the force-displacement graph shown in Fig. 4.
We find that the specimen cured following Procedure 1 (which involves pouring the Ecoflex when the PDMS is already completely cured) is the one that withstands the lowest tensile stress. This means that the bond between the two materials is stronger if they bind while curing. Further, the results of Figure 4 suggest that procedures 6 and 7 guarantee the best performance in terms of material bonding, confirmed by the fact that the fracture takes place away from the bonding interface. However, procedures that involve pouring Ecoflex first (i.e. Procedures 5, 6, 7 and 8) are not advisable due to air bubble formation at the interface of the two materials. Finally, specimens cured with Procedures 4 and 8 are excluded because the two materials do not form a clear interface at the junction, as shown in Figure 5. This means that in these two procedures the second material is poured too early, and the materials end up mixing. As a result, we choose Procedure 3, as it involves pouring PDMS first and withstands the highest elongation before breaking.

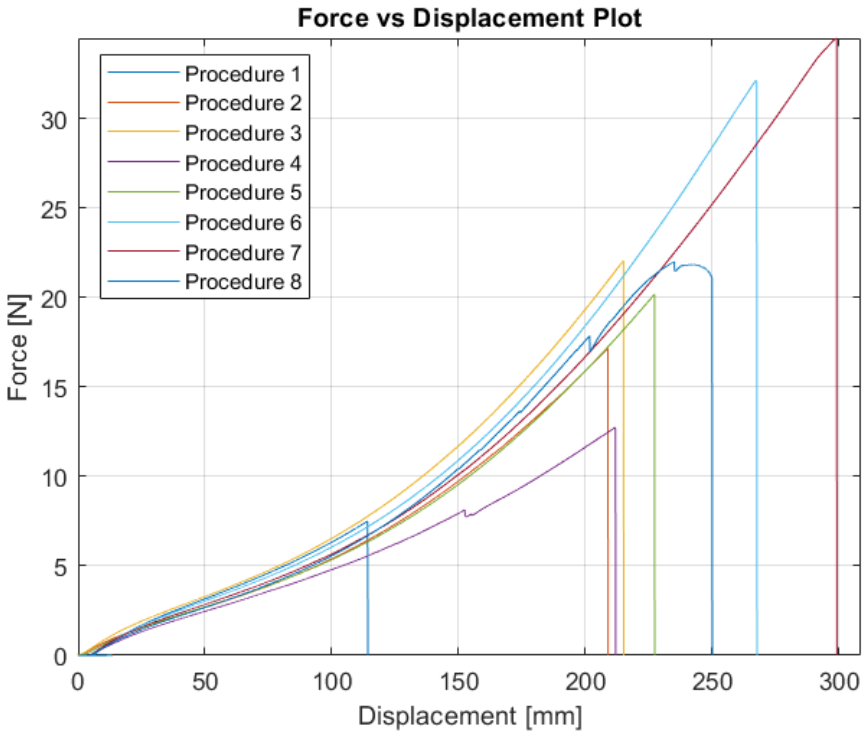

Figure 4: Tensile test results for PDMS-Ecoflex specimens cured following eight different curing procedures.

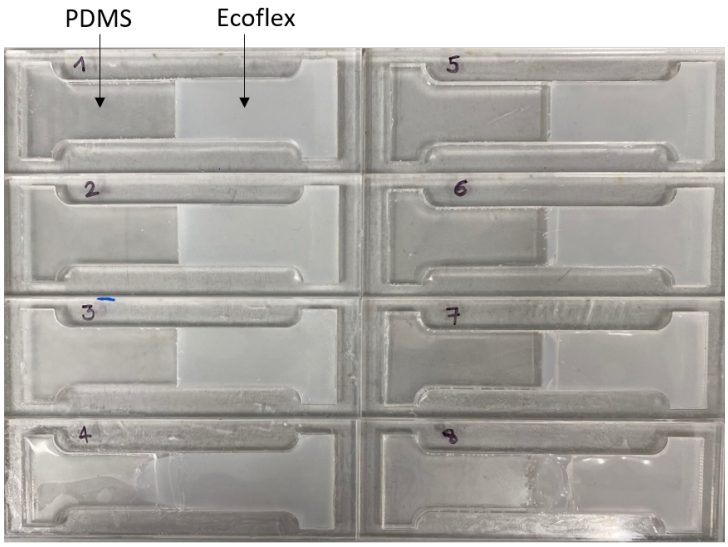

Figure 5: PDMS-Ecoflex dog-bone specimens.

After selecting Procedure 3 for membrane fabrication, we introduce another step at the end of the membrane curing procedure. This step consists of heating the final membrane for 1 hour at 200°C, which is the most elevated curing temperature that does not cause thermal decomposition of PDMS [1]. This step is added because previous studies have shown that the Young's modulus of PDMS increases linearly with the curing temperature [1, 2]. In order to demonstrate the effect of this additional step, we conduct a tensile test on dog-bone specimens made of either PDMS or Ecoflex only, exploring the difference in the samples' behavior when heated at 200°C. In Figure 6 we report the results of 4 PDMS samples. All the samples are cured as in Procedure 3. In addition, specimens 3 and 4 are treated with the addition heating step at 200°C for 1 hour after curing. As expected, the stiffness of PDMS significantly increases when the samples are heated at elevated temperature. Differently, the mechanical properties of Ecoflex (Figure 7) do not change considerably when the samples are heated at 200°C. Thus, by adding a final step of heating at 200°C at the end of the curing procedure, we are able to increase stiffness ratio between the PDMS and Ecoflex, allowing the membrane to achieve more complex configurations when inflated.

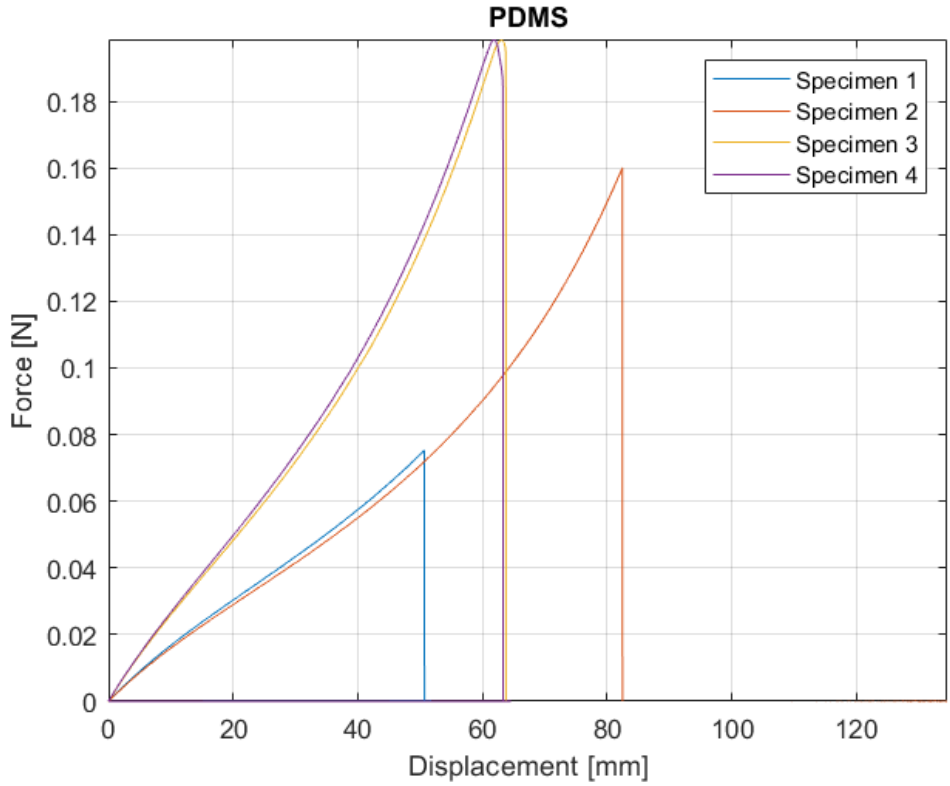

Figure 6: Tensile test comparison between PDMS samples not heated (specimens 1-2) and heated (specimens 3-4) at 200°C after curing.

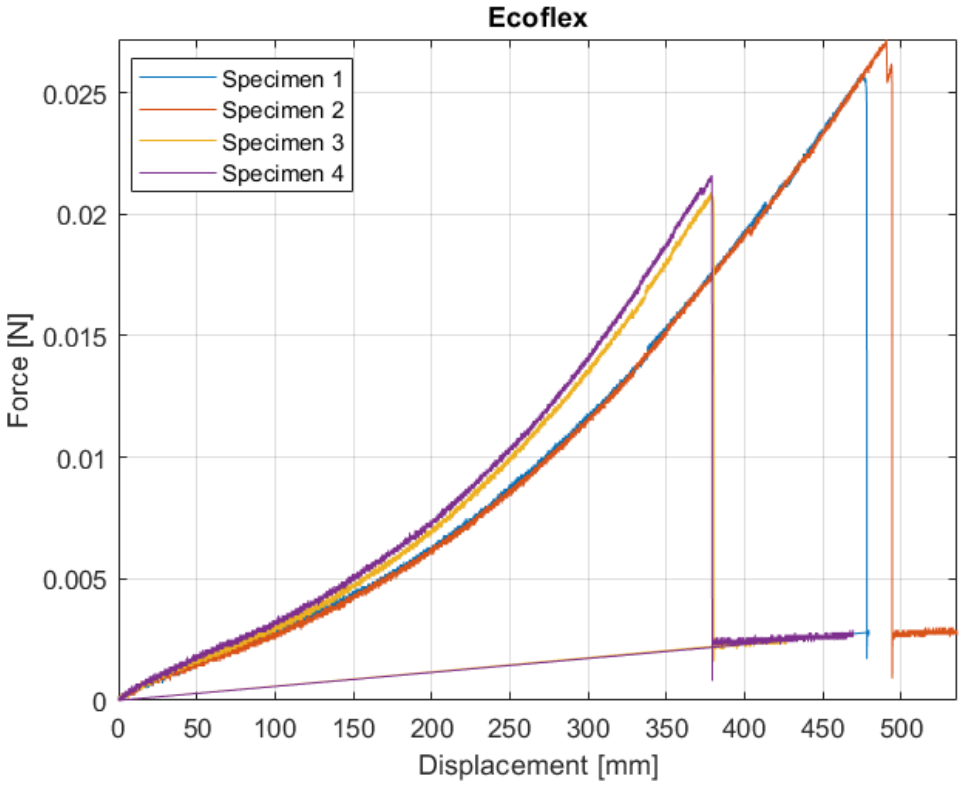

Figure 7: Tensile test comparison between Ecoflex samples not heated (specimens 1-2) and heated (specimens 3-4) at 200°C after curing.

1.3 Fabrication of the pixelated membranes

Our membranes are fabricated using a multi-step molding process. The whole process requires around 7 hours. Note that the membrane binary design occupies a square area of 100x100 mm^2, to which we add a margin of 25 mm on every side (made of PDMS, see Figure 1e in the main manuscript) to fix the membrane on the pressure chamber. Thus, the area of the fabricated membrane is 150x150 mm^2.
Our membranes are fabricated using the following 12 steps (see Figure 8 and the supplementary Movie S1):

1. We start by laser cutting the components of the acrylic mold. We first laser cut a square with length of 170 mm on 3 mm thick acrylic sheet. This will be the mold base (Figure 8a). We also laser cut a square with length 150 mm from double-sided tape (Figure 8b).

2. Then we create a square frame that has the same size of the base, and an internal square hole with length of 150 mm out of a 6 mm thick acrylic sheet. We also cut the binary design out of the same acrylic sheet (Figure 8c) and out of double sided tape (Figure 8d).

3. In order to obtain a 7 mm deep mold, we laser cut both frame and binary design again, this time out of a 1 mm polyester sheet (Figure 8e).

4. We attach the 6 mm frame to the mold base using adhesive tape. We stick the double-sided tape square created in 1. onto the mold base (Figure 8f) after removing its protective layer on one side. We glue the 1 mm frame on the 6mm frame using superglue. By doing so we obtain a 7mm deep mold.

5. By using the laser cutter at low power we cut the binary design directly through the tape without damaging the acrylic mold underneath (Figure 8g). Note that this step is crucial because it acts as a reference to the location of each stiff/soft pixel on the mold base. We remove the second protective layer from the tape (in the areas corresponding to the soft pixels).

6. We stick the 6 mm thick acrylic binary design (from point 2., Figure 8h) on the exposed tape, which corresponds to the location of the soft pixels. By using the binary design cut on double-sided tape (from point 2. , Figure 8i), we stick the 1 mm thick polyester binary design on the 6 mm thick acrylic binary design (Figure 8l). This last layer will be removed when PDMS is half-cured, in order to pour the 1 mm layer of Ecoflex.

7. At this point, the acrylic mold is completed (Figure 8m). A release agent is spread on the mold and the PDMS (already mixed) is poured in the mold forming a 7 mm thick layer corresponding to the stiff pixels (Figure 8n). Air bubbles which form in the samples are removed using the degassing method described previously. The PDMS is cured at 60°C for 1.5 hours (half of its cure time).

8. We take the mold out of the oven. Then we remove the 1 mm thick mold layer and the double-sided tape underneath, leaving a 1 mm deep empty pocket corresponding to the soft pixels (Figure 8o).

9. In order to get a homogeneous 1 mm thick layer of Ecoflex 00-30, we dispense it by weight. We place the mold on a precision scale and pour 0.107 g of Ecoflex (with specific density = 1.07 g/cc) for each soft pixel, which has a volume of 0.1 cm^3 (Figure 8p).

10. The membrane is set on a Thermo Scientific ™ leveling platform and cured at room temperature for 1 hour. The use of a three-point leveling platform ensures the fabrication of a flat layer of Ecoflex. During this time the two elastomeric networks start to bond together.

11. Then, we put the membrane in the oven at 60°C for 1.5 hours in order to complete the curing procedure. At the end of this step PDMS and Ecoflex are completely cured and bonded together.

12. Finally, in order to increase the stiffness ratio between the two materials, we heat the membrane at 200°C for 1 hour.

At the end of the procedure, we carefully remove the membrane from the mold (Figure 8q) and we measure the membrane thickness in order to verify that the PDMS and Ecoflex layers are 7 mm and 1 mm thick respectively. The pressure chamber flange is used to mark the positions of the holes on the perimeter of the membrane (Figure 8r), and we use a hole-puncher to produce them (Figure 8s).
After being fabricated the membranes are mounted on the pressure chamber (Figure 8t) and inflated to the optimal pressure level instructed by the the ML model. The membrane inflation procedure is described in the next section.

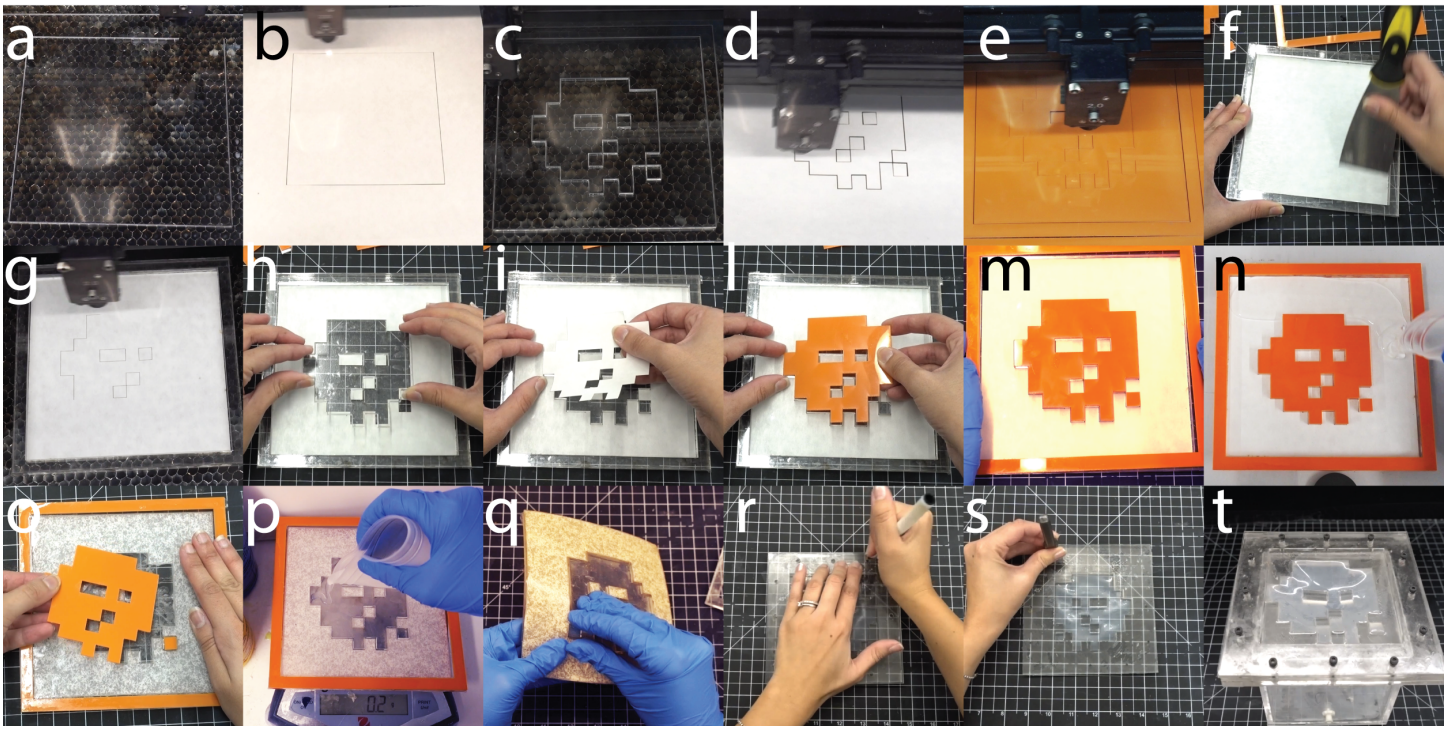

Figure 8: Membrane fabrication procedure via multi step molding approach.

2 Testing

2.1 Inflation

To inflate the membranes we designed a pressure chamber made out of acrylic. This is sealed by applying super-glue on all edges. The membrane has a PDMS frame around the 100x100 mm^2 area (as shown in Figure 1e of the main manuscript) which is perforated with a hole-puncher. Using these holes, the membrane is fixed on the top of the chamber by means of a flange and 12 bolts. Note that the internal edge of the flange that holds the membrane in place has dimensions 100x100 mm. This allows the membrane to deform out-of-plane only in that area.
On one side of the acrylic chamber there is an inlet for pressurized air, to which a tube is attached. The tube is connected to both to a manual pump and to a pressure control system through a 3 way valve. The pressure control system (Figure 9) consists of a pressure sensor (MPX5050DP, NXP USA Inc.), a Data Acquisition device (NI-USB-2009, National Instruments) connected to a laptop, which runs a Matlab script that records the data and displays the pressure level in real time. Air is manually pumped into the chamber in order to reach the desired pressure. Due to the difference in pressure between the inside and outside of the chamber, the membrane deforms out-of-plane assuming the target 3D shape.

2.2 3D scanner

In order to record the 3D shapes assumed by membranes upon inflation we use a hand-held Artec Space Spider 3D scanner based on blue light technology, which captures the geometry of real objects and produces a three-dimensional digital models. The set up for 3D scanning is shown in Figure 10. In order to obtain high quality scans,

1. before scanner acquisition, we spread a chalk spray over the inflated membrane, in order to give it a non-transparent color;

2. we place a large square acrylic sheet covered with patterned wrapping paper as a frame around the membrane, on the chamber flange. By doing so, the colorful geometric pattern designed on the frame defines the plane where the membrane lies before inflation, acting as a reference for the 3D scanner to distinguish the membrane from the background.

After a calibration step, a 360° data acquisition is performed by rotating the chamber on a Lazy Susan table, while the Artec Studio software for 3D scanning and data processing displays the 3D model of the object on the computer. The software also allows to post-process the model, by smoothing and/or filling small missing holes on the object surface. The 3D model of the inflated membrane can be saved and exported as a .ply file.

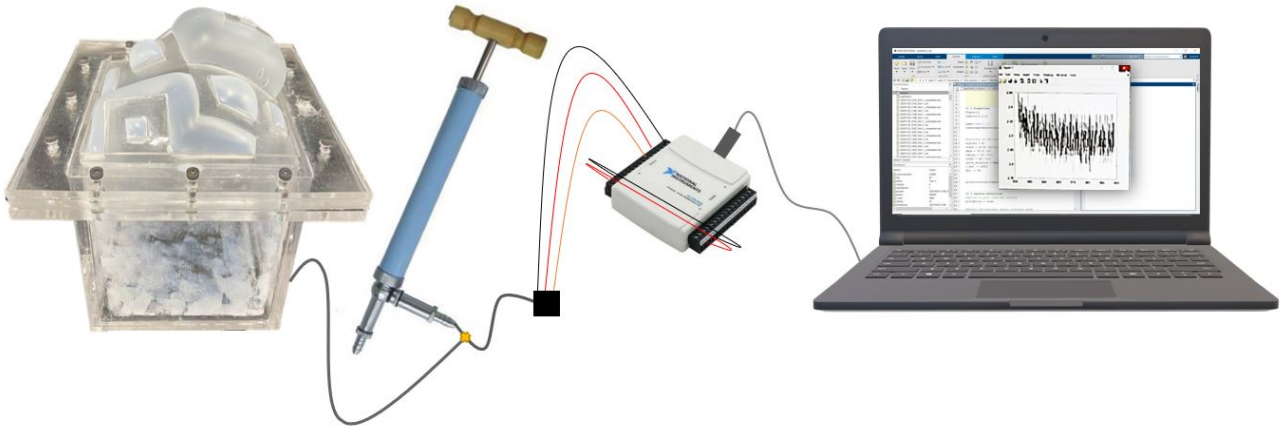

Figure 9: Pressure control system consisting of: a manual pump, a pressure sensor and a micro-controller.

Figure 10: Artec Space Spider 3D scanner acquisition set up.

3 Additional experimental results

In the following, we assess the membrane manufacture repeatability by fabricating and testing five soft membranes with a simple binary design (see Figure 11).
In the first row of Figure 12 we show the inflated membranes corresponding to binary design 1, 2, 3, 4 and 5 (from left to right). Note that each membrane is inflated to a different pressure level. The 3D scanner models of the inflated membranes are shown on the second row of Figure 12.

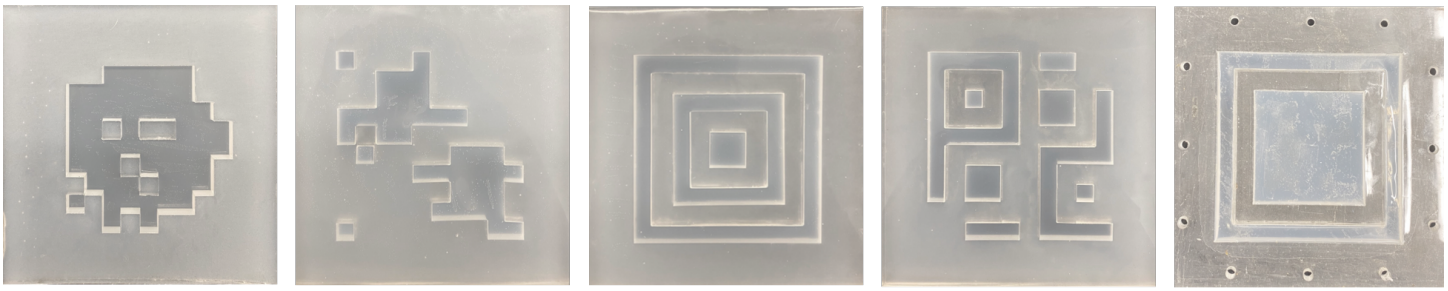

Figure 11: Membranes fabricated for validation of the experiments. Design 1, 2, 3, 4, 5.

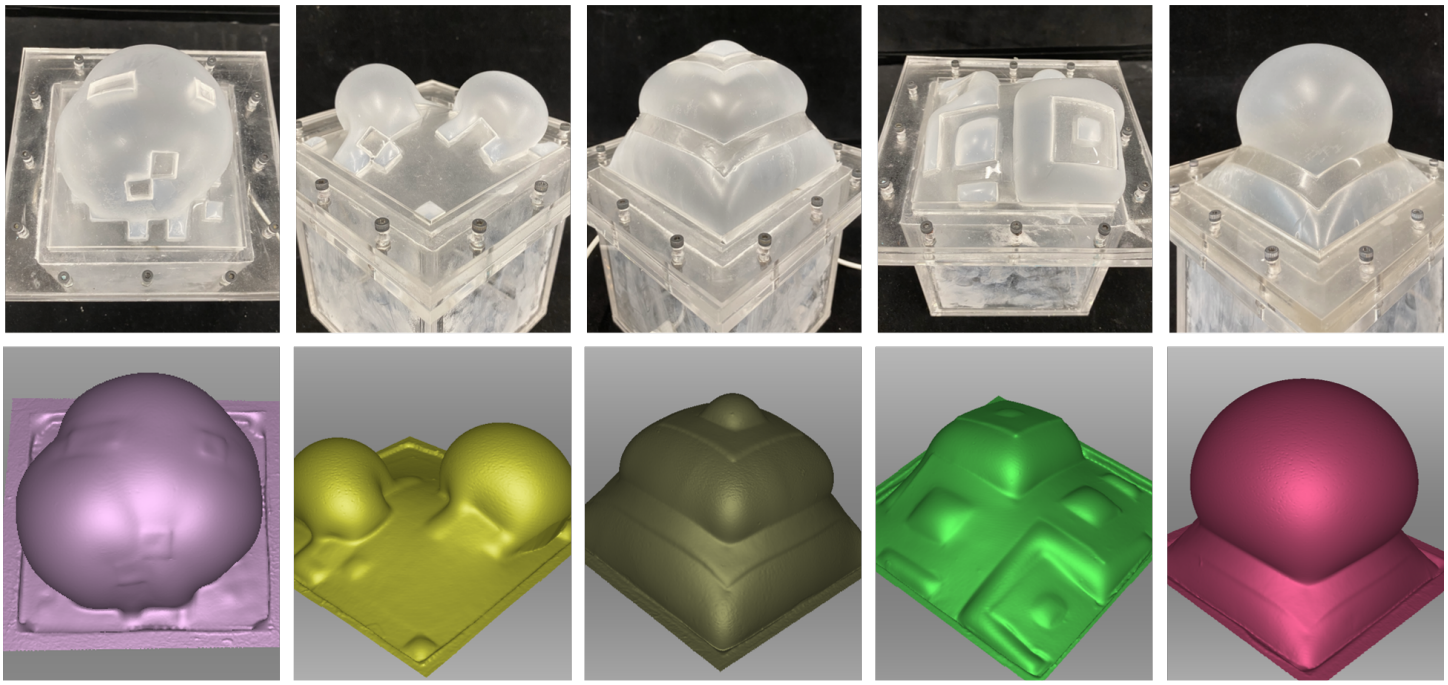

Figure 12: Inflated membranes and 3D scanner models.

4 Finite Element simulations

To gain a deeper understanding of the mechanical response of these membranes and calculate their deformation upon pressurization, we conduct non-linear Finite Element (FE) simulations within ABAQUS 2019/Standard. In all our simulations, we discretize the PDMS and Ecoflex pixels with four-node general-purpose shell elements (S4R element type) and four-node membrane elements (M3D4 element type), respectively. Guided by experimental measurements, the thickness of the PDMS and Ecoflex sections are set as 7 mm and 1 mm, respectively. Further, we model the response of both elastomers using an incompressible Gent material model [3] with strain energy density function W given by

$$W = -\frac{\mu J_{lim}}{2} \ln\left(1 - \frac{I_1 - 3}{J_{lim}}\right), \tag{S1}$$

where μ and J_{lim} represent the small strain shear modulus and a material parameter related to the limiting stretch, and $I_1 = \mathrm{tr}(\mathbf{F}^T\mathbf{F})$, $\mathbf{F}$ being the deformation gradient. We find that the response of PDMS and Ecoflex is accurately captured using $(\mu, J_{lim}) = (850 \text{ kPa}, 2.8)$ and $(23 \text{ kPa}, 24)$, respectively . An in-house ABAQUS user subroutine (UHYPER) is used to define the hyperelastic material behavior given by Eq. [S1] in the FE simulations.

To remove rigid body translations and rotations, we fix all nodes located on the four edges of the membranes, and apply a pressure p (with $p \in [0, 3.5]$ kPa) directly on the bottom surface. We then solve for the deformation using the dynamic implicit solver (using a density of $\rho = 1000$ kg/m^3 for the PDMS and Ecoflex) and monitor the kinetic energy to ensure quasi-static conditions. We then export the deformed configurations of the membrane at p =1.5, 2.5 and 3.5 kPa and use the method of voxelization described in the main text to represent them.

4.1 Validation

In order to validate the FE model we compare the acquired 3D scanner model of the inflated membranes (represented by the green point cloud in Figure 13) with the FE results of the corresponding binary designs (represented by the purple point cloud in Figure 13) inflated at the pressure level used in the experiments. More specifically, we import the two point clouds into Matlab, overlap them and visually compare. Figure 13 shows an accurate overlapping of the two point clouds for all the geometries considered in Figure 12.

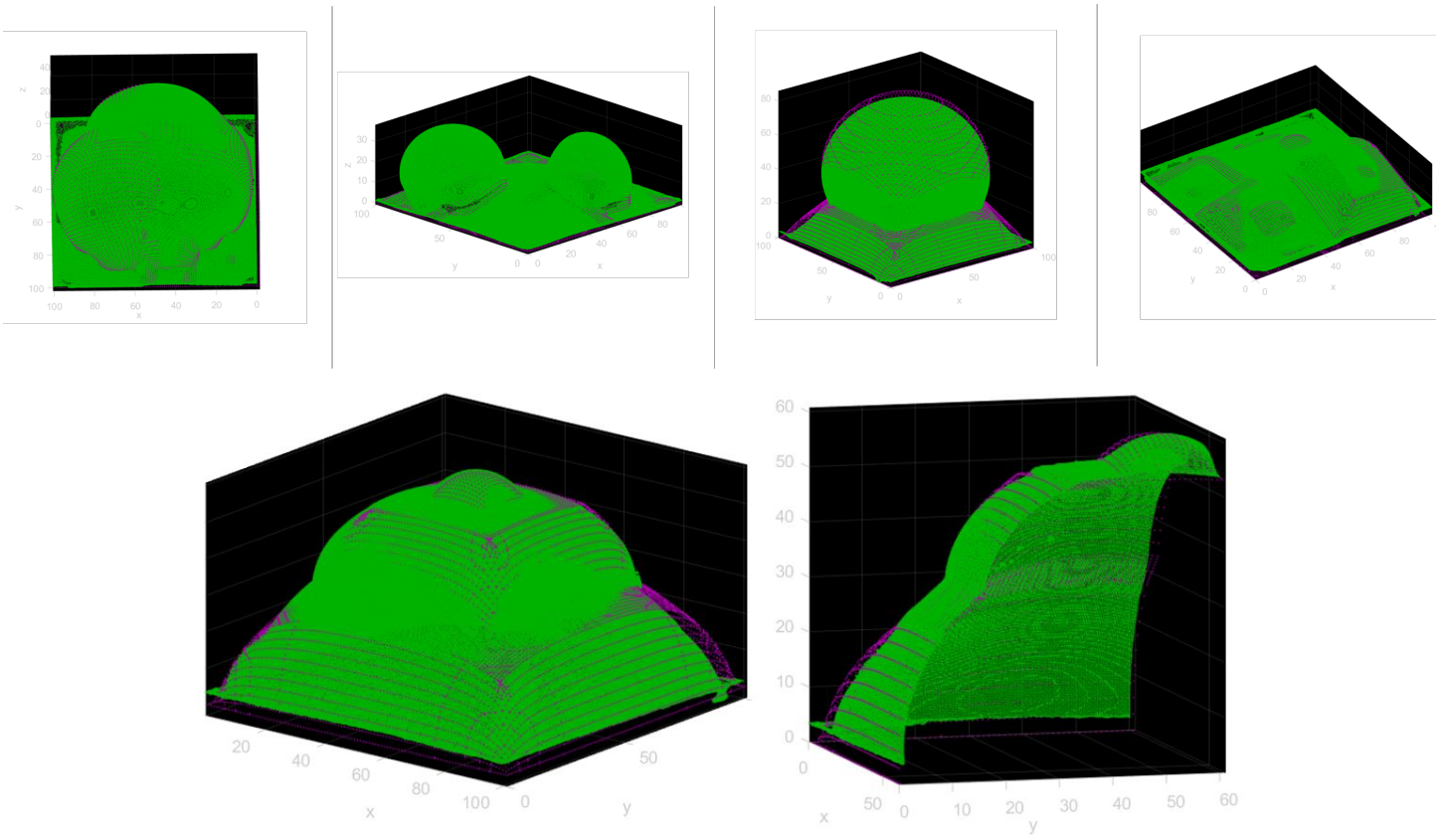

Figure 13: Point cloud comparison between the 3D model of inflated membrane (in green) and the Finite Element simulation of the binary designs (in purple).

4.2 2D pixelated binary designs

Since diverse (non-redundant) datasets can improve the performances of neural networks, we adopt three different strategies to generate 2D pixelated binary designs to be simulated using FE. Each design has in total $N = 10 \times 10$ pixels with N_E pixels of Ecoflex material and N_P pixels of PDMS material.

1. Random algorithm. To generate the random pixelated binary, we first assume all the pixels are made of PDMS material (i.e. $N = N_P$). Then, by seeding N_E pixels to the design randomly, we achieve a randomized binary array as the input of the simulations. In order to obtain most of the non-trival deformation of the membrane in a limited database, we choose the ratio between the Ecoflex pixels $N_E/N \in [0.2, 0.6]$.

2. Islands algorithm. For the islands algorithm, inspired by a recent paper [4], we assume all the pixels are initially made of PDMS material (i.e. $N = N_P$) and randomly seed $N_I \in [1, 10]$ pixels of Ecoflex material. The pixels made of Ecoflex represent the islands of the binary array. Then, we identify the PDMS pixels that directly connected to these islands, and randomly choose one pixel from these selected PDMS pixels and change its from PDMS to Ecoflex. Finally, we repeat this procedure until N_E reaches to a certain value. In order to obtain most of the non-trival deformation of the membrane in a limited database, we choose the ratio between the Ecoflex pixels N_E to the total amount of the pixels N in the range of $[0.2, 0.6]$.

3. Fibers algorithm. For the fibers algorithm, we start by approaching the problem as a permutations with repetition problem. A permutation problem identifies the different ways in which we can order a subset of objects, taken from the larger set of objects. If some of the objects are the same, the problem becomes a permutation problem with repetitions. A classic example would be 'how many different words can you create shuffling the letters of the word Mississippi?'. Since there are 11 letters in the word 'Mississippi' ($n = 11$, i.e. the total number of objects) and we want to create new words made out of 11 letters ($r = 11$, i.e. number of object selected), the number of all possible variants is given by

$$ {}_nP_r = \frac{n!}{(n-r)!}. \tag{S2} $$

In the presence of repetitions, the number of *unique* words can be obtained by dividing it by the number of ways we can arrange each set i of repeating letters. For example, in the case of 'Mississippi', we could identify a first set corresponds to the letter 's' which repeats $x_1 = 4$ times. This can be arranged in 4! different ways. Therefore

$$ N_{unique} = \frac{{}_nP_r}{\prod_{i=1}^{k} x_i!}, \tag{S3} $$

where k is the number of different sets. For our 'Mississippi' example ${}_nP_r =_n P_n = n! = 11!$, therefore $N_{unique} = \frac{11!}{1!4!4!2!} = 34650$.
We apply this method to produce designs. We start by creating a vector that has 10 vacant positions. We can fill these position with binary values (0 and 1) corresponding to soft and stiff pixels respectively. It is known that features aggregating more consecutive pixels of the same type will have a stronger effect on the deformation of the membrane [5]. For this reason we firstly aim at producing designs with fibers that have a minimum size of 2 pixels. We divide the the vector in 5 spatial domains, each containing a pair of equal pixels. We can treat each of these domains as a super-pixel. This is the equivalent of finding all the possible acronyms of a 5-letters word composed by only 2 letter types (i.e. $n = 5$, $r = 5$, $k = 2$).
When computing the permutation with repetitions, the result could be one of the following cases:

- all stiff pixels $\rightarrow N_{1unique} = 1$
- 1 soft pixel and 4 stiff $\rightarrow N_{2unique} = 10$
- 2 soft pixels and 3 stiff pixels $\rightarrow N_{3unique} = 5$

Also taking into account the complementary cases, we have a total number of unique designs $N_{unique} = 32$. After removing mirrored versions of these vectors (flipped in the horizontal direction, e.g. 00011 and 11000) and the two all-soft and all-stiff vectors, we obtain the designs in Figure 14a. To augment the number of unique 1D vector designs, we return to the pixel level resolution, which means that each 1D vector returns to 10 pixels. We then take each vector and shift it to the left by 1 pixel. If the shifted version is a new original 1D design, and no mirrored version is present in the current pool, the algorithm keeps the new design, or discards it otherwise. After this step we obtain the pool of designs in Figure 14b. Then we flatten the 1D designs into a single 1D row vector, p, and obtain its column version by transposing it, q. We perform Boolean operations between the row and column vectors to create 2D square designs, comprising 10x10 pixels (Figure 14c). All possible designs are obtained by using the Boolean operations:
$p \wedge q, \neg(p \wedge q), \neg p \wedge q, \neg(\neg p \wedge q), p \wedge \neg q, \neg(p \wedge \neg q), \neg p \wedge \neg q, \neg(\neg p \wedge \neg q), (p \veebar q), (\neg p \veebar q)$,
where $\wedge$, $\veebar$ and $\neg$ indicates the Boolean operators AND, XOR and NEGATE respectively.
Once the designs have been created, the algorithm checks that each 2D design is unique, and that there are no rotated and symmetric versions in the final pool. The algorithm produces a final pool of about 2000 different designs with $n = 5$, $r = 5$, $k = 2$. Since the random and the island datasets comprise 2500 different designs, we also created a bigger pool of fibers designs without resorting to the super-pixel approach, which means we used $n = 10$, $r = 10$, $k = 2$. Note that the shifting step is not used in this case. We then randomly select about 500 designs from this new pool, to obtain a final fibers dataset of 2500 2D designs.

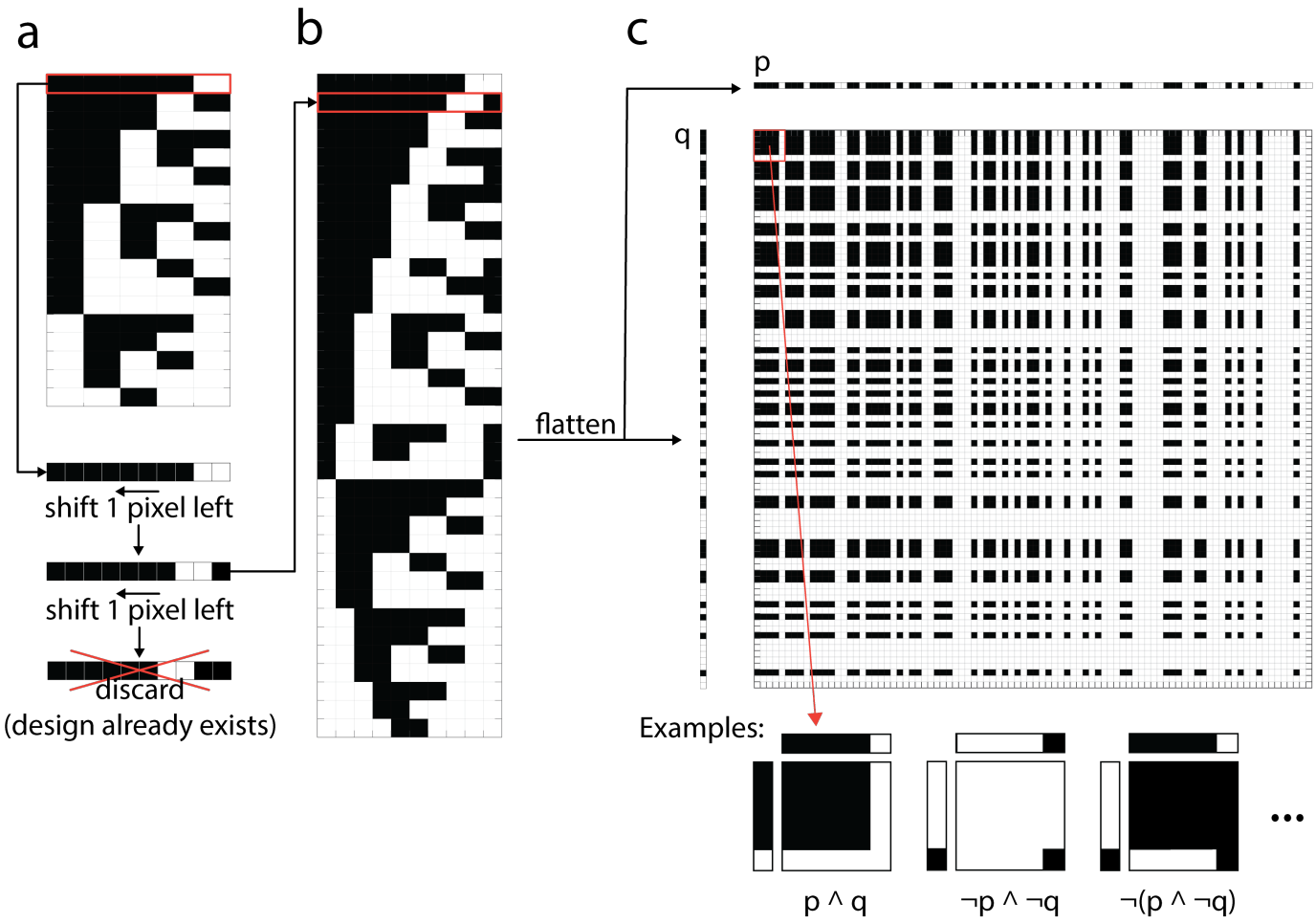

Figure 14: (a) 1D binary vector designs after computing permutations with repetitions with $n = 5, r = 5, k = 2$. (b) Design augmentation using shifting. (c) Matrix of designs obtained by flattening all the 1D vectors into a single 1D array and using different Boolean operations to combine them.

5 Machine Learning

In this section we provide details on the machine learning model, hyperparameters search, and the investigation of neural networks trained on different combinations of training sets.

5.1 Machine learning model

We built our neural networks (NNs) on TensorFlow version r1.12 [6]. The model was run on NVIDIA Tesla P100 GPU card. The Adam optimizer was used to minimize the mean squared error with a learning rate of 0.0001. We used a batch size of 100 and number of epoch of 50 for all training [7, 8].

5.2 Neural networks for inverse design

Our goal is to do inverse design of inflatable membranes using neural networks. Specifically, we want to predict the pressure level and binary design for a given inflated target shape. We denote the pressure as p, the binary design as $\mathbf{X}$ and the inflated shape (voxels) as $\mathbf{Y}$.
We use two fully connected layers (FCL) in which the computation in each layer is given by

$$a^{(l+1)} = g(W^{(l)}a^{(l)} + b^{(l)}), \tag{S4}$$

where g is a non-linear or linear function, $a^{(l)}$, $b^{(l)}$, $W^{(l)}$ are the activation (input) biases and the weights in a layer l, respectively. Rectified linear unit functions (ReLU) are applied to all layers. The first layer takes an inflated shape (voxels) $\mathbf{Y}$ as an input and the last layer outputs a pressure level p and a binary design $\mathbf{X}$. To obtain binary values (0 or 1) for $\mathbf{X}$, we apply sigmoid functions only to the last 100 activations of the last layer as the pressure (first activation) is a continuous variable.
We train the NNs to map Y to $\overline{p} \oplus \overline{\mathbf{X}}$ by minimizing the mean squared loss function [8]

$$\mathcal{L}_{\mathrm{NN}} = \frac{1}{N_{\mathrm{train}}} \sum_{i=1}^{N_{\mathrm{train}}} |\mathbf{X}_i - \overline{\mathbf{X}}_i|^2 + \lambda |p_i - \overline{p}_i|^2, \tag{S5}$$

where N_{train} is the number of training samples, overbar denotes ML predictions, and λ is an adjustable hyperparameter that controls the relative weight between the two losses.
To evaluate the performance of our NNs we introduce two accuracy scores: an accuracy on the predicted pressure level (R^2_{pressure}) and an accuracy on the predicted binary design (A_{binary}). Since the pressure level is a continuous variable, we use the R^2 metric to evaluate the pressure accuracy

$$R^2_{\mathrm{pressure}} = 1 - \frac{\sum_{i=1}^{N_{\mathrm{test}}} |p_i - \overline{p}_i|^2}{\sum_{i=1}^{N_{\mathrm{test}}} |p_i - \frac{1}{N_{\mathrm{test}}} \sum_{i=1}^{N_{\mathrm{test}}} p_i|^2}, \tag{S6}$$

where p is the true pressure, $\overline{p}$ is the predicted pressure and N_{test} is the number of datapoints in the test/validat set. To evaluate the binary prediction accuracy, A_{binary}, we use the average of the ratio of correctly identified pixels (N^i_{correct}) in the i-th data point to the total number of pixels ($N_{\mathrm{pixels}} = 100$)

$$A_{\mathrm{binary}} = \frac{1}{N_{\mathrm{test}}} \sum_{i=1}^{N_{\mathrm{test}}} \frac{N^i_{\mathrm{correct}}}{N_{\mathrm{pixels}}}. \tag{S7}$$

5.3 Varying hyperparameters λ and number of neurons

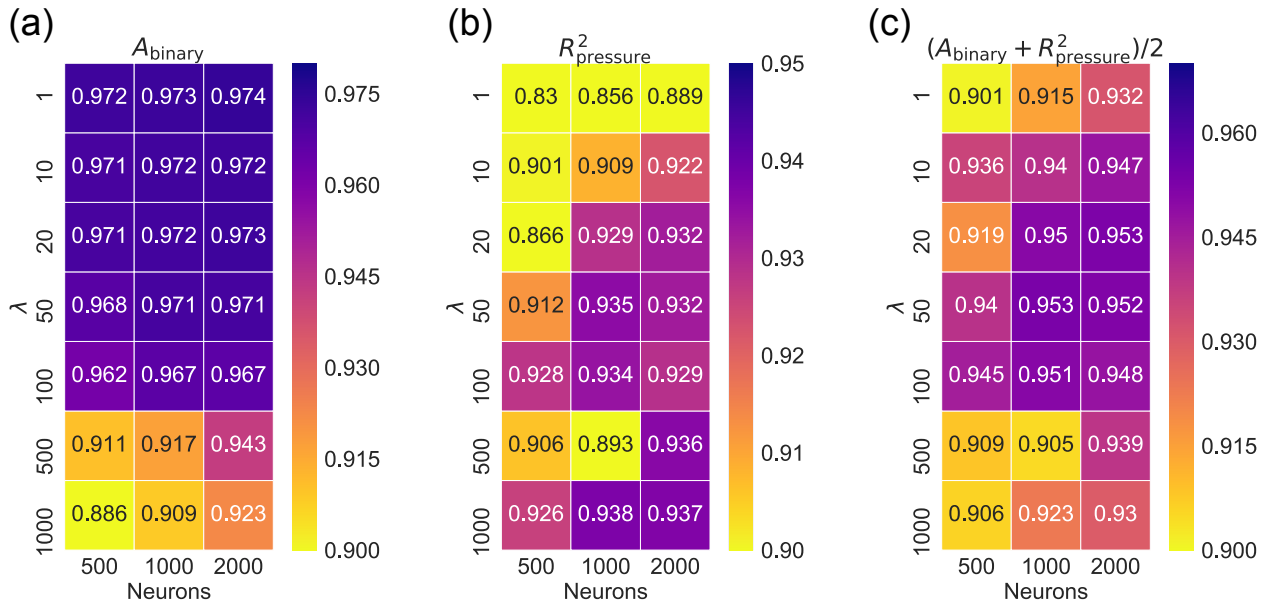

Figure 15: Effect of the hyperparameter λ and the number of neurons N_{neu} on (a) binary accuracy A_{binary}, (b) pressure accuracy $R^2_{pressure}$, and (c) average accuracy score $(R^2_{pressure} + A_{binary}))/2$.

We first use all classes of FE data to find the optimal NNs architecture. We use 80%, 10%, 10% of total simulations data for training, validation, and test, respectively. Note that the total number of training datapoints is 144000. We use a grid search to optimize our NNs. Specifically, we varied λ from 1 to 1000 and number of neurons from 500 to 2000. Note that we set the number of neurons in the first and second layer to be equal.

In Figure 15, we show how A_{binary}, $R^2_{pressure}$ and $(R^2_{pressure} + A_{binary})/2$ change with varying λ and number of neurons N_{neu}. Increasing λ generally improves the accuracy score of pressure as we penalize the model more when the pressure prediction deviates from the true pressure. For very small λ, the NNs predicts binary design better than pressure, whereas for very large λ, NNs predict pressure better. To select the best model, we choose a model with the highest average accuracy score $(A_{binary} + R^2_{pressure})/2$ on the validation set. The model with $N_{neu} = 1000$ and $\lambda = 50$ was found to be the best model with the highest validation average accuracy score of 0.953.

5.4 Performance on different training sets

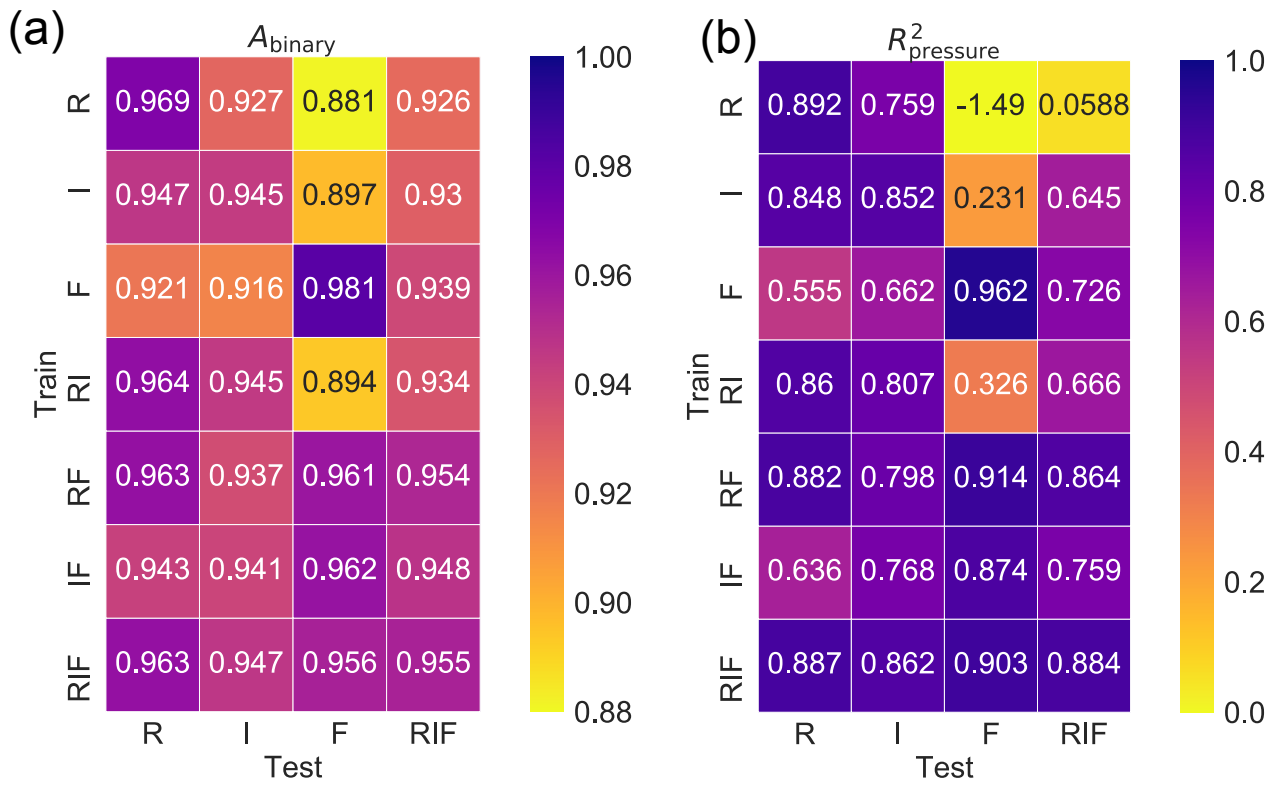

Figure 16: Accuracy scores of NNs with $\lambda = 50$ and $N_{neu} = 1000$ trained and tested on different combinations of classes: (a) A_{binary}; (b) $R^2_{pressure}$. Note that the limits of the color bars in plots (a) and (b) are different.

After optimizing the NNs, we also investigate how training NNs with different combinations of training sets give different performances. As discussed in Section 4B, in this study we consider three classes of designs: (*i*) random designs (R); (*ii*) island designs (I); and (*iii*) fiber designs (F).
We start by training and testing the NNs with single-class datasets (R, I or F). For each class of dataset we employ 80% of the datapoints as training set, 10% as test set and 10% as validation set. This means that the NNs get trained on N_{train} =48000 designs from one class and tested on unseen N_{test}=6000 designs from the same class. The results reported in Fig. 16 show that, when trained on one single class of designs and tested on designs from the same class, the NNs provide similar accuracy for the three considered datasets. Differently, we find that a model trained with one set of designs does not generalize well to other classes, leading to poor predictions on pressure ($R^2_{pressure} < 0.7$) and lower accuracy on binary designs.
However, we find that the results can be improved by training the NNs with combinations of different design classes. Specifically, we consider combinations of random and islands (RI), random and fibers (RF), islands and fibers (IF) and random, islands and fibers (RIF) designs. In order to compare performances, we always train the NNs with 48000 datapoints split equally between classes (e.g., for the combination RF the NNs is trained with 24000 datapoints from the R class and 24000 from the F class). As shown in Fig. 16b, we find that models trained on the four hybrid classes (i.e., RF, IF and RIF) outperform the single-class trained models, with only the RI model having poor performance on the pressure predictions when tested with designs from the F class ($R^2_{pressure} = 0.326$). On average, a model trained with the combination of three different classes of designs (RIF) shows the best performance both in binary design prediction and pressure prediction (average $A_{binary} = 0.955$ and average $R^2_{pressure} = 0.884$). Hence, we train our NNs using all three classes of designs (RIF) to perform the inverse design of membranes morphing into user-defined 3D shapes upon inflation.

Membrane Fabrication Fabrication procedure of the membranes via multi step molding.

References

[1] M. Liu, J. Sun, Q. Chen, *Sensors and Actuators A: Physical* **2009**, *151*, 1 42.

[2] I. D. Johnston, D. K. McCluskey, C. K. L. Tan, M. C. Tracey, *Journal of Micromechanics and Microengineering* **2014**, *24*, 3 035017.

[3] A. Gent, *Rubber chemistry and technology* **1996**, *69*, 1 59.

[4] Y. Mao, Q. He, X. Zhao, *Science Advances* **2020**, *6*, 17 eaaz4169.

[5] J. Pikul, S. Li, H. Bai, R. Hanlon, I. Cohen, R. Shepherd, *Science* **2017**, *358*, 6360 210.

[6] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, **2015**, URL `https://www.tensorflow.org/`, Software available from tensorflow.org.

[7] P. Z. Hanakata, E. D. Cubuk, D. K. Campbell, H. S. Park, *Physical review letters* **2018**, *121*, 25 255304.

[8] P. Z. Hanakata, E. D. Cubuk, D. K. Campbell, H. S. Park, *Physical Review Research* **2020**, *2*, 4 042006.