Computational Modeling and Design of Capacitive Stretch Sensors

ARVI GJOKA, New York University / Courant, USA
YONGKANG SUN, New York University / Courant and Stony Brook University, USA
ROI PORANNE, University of Haifa, Israel
DANIELE PANOZZO, New York University / Courant, USA

Fig. 1. Given a set of target poses, we find an optimal capacitive sensor array that maximally distinguishes between them. The sensor is fabricated using a silicone mixture and wrapped around the deformable object, providing major improvements when compared with a baseline, unoptimized sensor.

A stretch sensor is a device that attaches to objects and measures the amount by which they deform. These sensors have shown great promise as an alternative to vision-based motion-capture systems, and for robotic sensing. Currently, they are generally limited to linear designs, and require a somewhat challenging calibration process. Our goal is to enable *inverse* design of such sensors, and to largely eliminate the calibration process.

To this end, we introduce an accurate, *differentiable* simulator for *capacitive* stretch sensors, that treats both the elasto- and *electro*-static parts of the system. Differentiability allows optimizing the geometry of the sensor in order to improve its design for specific applications. We demonstrate the accuracy of our simulator and the effectiveness of our sensor optimization process for various use cases, such as human interfaces and robotics.

CCS Concepts: • **Computing methodologies** → **Modeling and simulation**; **Shape modeling**; **Simulation evaluation**.

Additional Key Words and Phrases: Deformable Sensing, Differentiable Simulation, Soft Robotics, Shape Optimization, Finite Element Method

ACM Reference Format:
Arvi Gjoka, Yongkang Sun, Roi Poranne, and Daniele Panozzo. 2025. Computational Modeling and Design of Capacitive Stretch Sensors. *ACM Trans. Graph.* 44, 6, Article 239 (December 2025), 15 pages. https://doi.org/10.1145/3763274

Authors' addresses: Arvi Gjoka, New York University / Courant, New York, USA, arvi.gjoka@nyu.edu; Yongkang Sun, New York University / Courant and Stony Brook University, New York and Stony Brook, USA, yongkang.sun.1@stonybrook.edu; Roi Poranne, University of Haifa, Haifa, Israel, roi.poranne@gmail.com; Daniele Panozzo, New York University / Courant, New York, USA, panozzo@nyu.edu.

ACM 0730-0301/2025/12-ART239
https://doi.org/10.1145/3763274

1 INTRODUCTION

Sensors are devices that measure physical quantities by converting them into interpretable signals. The old *mercury thermometer* is a sensor that allows measuring temperature by converting material expansion into length, which is simple to read. Sensor design required understanding the physical principles underlying the quantity to be measured, and creating a system that can reliably *transduce* that quantity into a usable output. The thermometer highlights the challenges of designing an effective sensor: A thermometer must be calibrated, it has a limited range and resolution, as defined by the number of markings. It can be hard to read, and requires time to reach equilibrium. These are just a subset of properties that can be taken into account and optimized for when designing a sensor. In this paper we discuss the geometric aspect of sensor design, and apply it to deformation measuring devices known as *stretch sensors*.

Stretch sensors are a promising alternative to vision-based motion capture systems. They can be fastened to the surface of a soft object, and allow estimating its deformations as it stretches. This has the advantage in settings with frequent occlusions, such as object manipulation by a human hand. A popular type of stretch sensor is composed of an elastic ribbon (textile or silicone) with an embedded electronic component whose resistance or capacitance changes when the ribbon is stretched. In particular, *capacitive* sensors are based on the principle that the capacitance of a capacitor is determined by its geometry. A *silicone* capacitive sensor is a composite silicone sheet made of a stack of dielectric and conductive layers which form an array of stretchable capacitors: stretching it changes the capacitance, which can be measured directly. While

these sensors hold immense potential, they are currently limited to simple, single-dimensional readings. Our goal is to develop the methodology for a multidimensional, free-form deformation sensing device, based on the simple concepts described above.

The challenge is to recover the geometry of the sensor by reading the capacitance alone. The case of flat capacitor, scaled in one direction, is simple; capacitance is proportional to the plates' area and inversely proportional to the plates' distance. However, treating a generally deformed stretch sensor is not as simple. First, estimating the capacitance is difficult, as there are no simple formulas. Second, the map between shape and capacitance is not unique in this case; there can be multiple deformations leading to the same capacitance values. This problem has been tackled before with a *data-driven* approach, using a calibrated vision-based system to acquire a set of deformations and map them to capacitance values. This process is expensive and lengthy, and limits the ability to iteratively optimize the sensor layout (i.e. how are the capacitors embedded in the silicone sheet). We propose to replace the external system with a *simulation*. Our approach enables calibration and design of stretch sensor layouts without requiring calibration data.

To this end, we develop a hybrid elasto- and electro-static differentiable simulator that accurately estimates the capacitance of a deformed sensor. Boundary conditions can be scripted in a simulated environment. The simulator then computes the deformed sensor pose and the corresponding capacitance using a volumetric electrostatic simulation. This then allows optimizing the sensor layout using shape optimization. Our main contributions include:

(1) A *validated*, differentiable elastic and electrostatic simulator that enables fully virtual calibration of silicon capacitive sensors, eliminating the need for motion capture systems.
(2) A sensor design optimization process for capturing a prescribed set of poses.
(3) Application to real and simulated soft robots equipped with optimized capacitive sensors that enable state estimation.

We believe our contribution will advance applications in graphics and robotics, and drive progress in differentiable simulation.

2 RELATED WORK

Sensing Principles. Various stretch sensing technologies have emerged in recent decades. Resistive sensors utilize an elastic, conductive material that exhibits changes in electrical resistance when stretched. By measuring these resistance changes, the degree of deformation can be determined. Capacitive stretch sensors function as deformable capacitors, where mechanical strain induces measurable changes in capacitance. Alternative sensing approaches include optical, piezoelectric, and triboelectric mechanisms. Sensors differ in properties such as stretchability, durability, sensitivity, measurement accuracy, linearity, response time, and hysteresis effects. Additionally, fabrication costs, duration, and customizability can all be factored in. See [Souri et al. 2020] for a recent review and comparative analysis.

We focus our computational design approach on capacitive sensors due to their large design space (geometry of the conductive layers) coupled with the possibility to fabricate them with off-the-shelf hardware available in most fablabs. Additionally, the sensors have low hysteresis, are cheap, and deformable.

Sensor Calibration. Recovering strains from capacitance measurements requires a calibration process, which presents a significant challenge. Simple, cord- or ribbon-like capacitive sensors tend to respond linearly to stretching. For these sensors, calibration involves stretching them by a predefined, measurable amount, recording the corresponding capacitance, and applying a linear regression model. However, more complex cases will require more sophisticated models. As an example, due to the high degree of hysteresis *resistive* sensors exhibit, [Miodownik et al. 2019] used an LSTM to calibrate such a sensor, essentially training a time-dependent model. Complex geometries require multiple, spatially distributed sensors, or sensor *arrays*, in order to estimate the deformation across the entire surface. In these cases, an external tracking system is used to capture the geometry and fit it to the measurements. For instance, [Chen et al. 2022] used a kinect to fit sensor measurements to a skeleton. Similarly, [Glauser et al. 2019a] used an OptiTrack system to track specific surface locations, while a follow-up work used a hand-tracking system specifically for a glove with embedded stretch sensors [Glauser et al. 2019b].

To eliminate the need for calibration measurements, we propose to *simulate* the deformation and capacitance of the sensor. This approach allows us to generate accurate data without depending on external sources. Similar methods have been previously employed, such as in [Tapia et al. 2020] and [Thuruthel et al. 2020], for a cord-like sensor. However, to our knowledge, no previous work has attempted to fully simulate a stretch sensor in terms of both its elastic properties and electrostatics.

Sensor Design Optimization. While the above-cited works primarily focus on recovering geometric properties from sensor values, we ask: *can we optimize the sensor itself to improve precision and performance?* The canonical problem related to sensor design is the well-known *optimal sensor placement* problem, originating with the seminal work by Kammer [Kammer 1991], and going back to earlier work on information theory and optimal experimental design. This problem addresses how to position sensors to maximize information gain about the system or optimally distinguish between different states. Optimal sensor design and placement have since been investigated across numerous domains. In structural engineering, researchers optimize sensor placement to maximize information gain for structural health monitoring [Papadimitriou et al. 2000]. For antenna design, various optimization techniques have been developed to enhance signal reception and transmission patterns [El Misilmani et al. 2020; Rais et al. 2009]. Similar methods have been employed for MRI coil design [Takahashi 2024; Turner 1993].

There have been several attempts to optimize stretch sensors for robotics and human interfaces in particular, which we briefly overview. For touch sensing applications, [Wu et al. 2020] experimentally optimized different electrode array configurations for a capacitive sensor. Somewhat similarly, [Presti et al. 2024] experimentally optimized 3D-printed wearable strain sensors through trial and error. Some researchers suggested approaching sensor optimization via information theory, using joint entropy-based optimization.

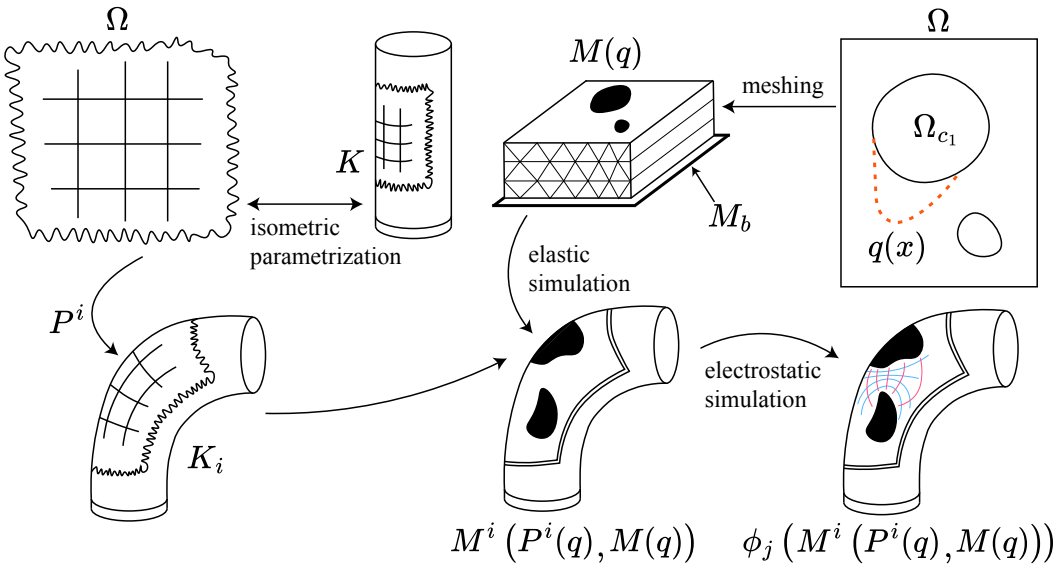


Fig. 2. Visual outline of the sensor simulation algorithm and notation used.

The idea is that maximizing the entropy of the distribution of measurements would be a better utilization of measurement space. For example, [Thuruthel et al. 2020] maximized the entropy of measurements obtained from multiple sensors embedded in a deformable object, assuming a simplistic physical model. [Spielberg et al. 2021] proposed a learning-based approach to optimize sensor placement for specific robotic tasks, though limited to simulation. Finally, more closely related to our goals, [Tapia et al. 2020] introduced a sparsification approach for sensor design for soft robots. However, their approach only treats cord-like sensors and does not consider the full electromagnetic simulation of capacitive sensors.

To summarize, most existing approaches either rely on experimental data collection or simplified physical models for optimization. Our work differs fundamentally by leveraging a differentiable physics simulator that can model both the mechanical deformation *and* capacitive response of the sensor. This enables continuous, principled design optimization without requiring expensive physical prototyping or external tracking systems that were required in, e.g., [Glauser et al. 2019a,b]. Our approach allows us to explore a much larger design space than previous methods while maintaining physical accuracy.

3 OVERVIEW

In the following, we briefly describe the principles of capacitive stretch sensors and their fabrication. We largely follow the low-cost fabrication procedure introduced in [Glauser et al. 2019a], with minor modifications (Appendix B).

Fabrication and Hardware. The common model for a (rigid) capacitor is two parallel, conductive plates, separated by a dielectric material. The capacitance is a function of the area and distance between the plates. Hence, if the plates could stretch and change their area, it would be possible to tell by how much simply by measuring the capacitance. Stretchable capacitors can be made out of silicone: they are a composite material of interleaved conductive and dielectric layers of silicone. Silicone is a dielectric material, but can be made conductive by mixing it with carbon black particles.

The process described in [Glauser et al. 2019a] allows for control of the layout of the conductive layers, effectively forming an elastic Printed Circuit Board (PCB). Their main innovation is the use a grid-like layout that effectively creates many capacitors within a single silicone sheet, enabling localized stretch sensing. Different layouts lead to different capacitances and how they change when the sensor is deformed.

Simulation/Calibration. In contrast to [Glauser et al. 2019a], which requires fabricating first and relying on external sources for calibration, our first contribution is a physical simulation of the sensor (Sec. 4). This enables testing layouts in simulation, avoiding fabrication and data collection.

Our key observation is that to model deformable sensing systems, a one way coupling between the elastodynamics and the sensing modality (i.e. electrostatics) is typically all that is required. This is because to sense deformation, it is undesirable for the sensing modality to affect the deformation itself, such as the electrostatics exerting forces on the sensor. The latter behavior is more common with actuation systems, which may require two way coupling, than sensing systems.

We begin with a 2D layout design of the conductive layer, and use it to build a volumetric mesh representing the layers of silicone. This mesh can be virtually attached to other simulated objects, and repositioning these objects induces a deformation of the sensor, which is computed using the Finite Element Method (FEM). An electrostatic simulation is then used to compute the capacitances in the deformed pose.

Design Optimization. Our second contribution (Section 5) builds upon the first, using the simulator to optimally design a sensor to capture a prescribed set of poses. Since the two simulation steps are differentiable, it is possible to define a *bi-level* optimization and solve using the adjoint method. We support arbitrary high-level objectives: since our goal is to use the measured capacitances to distinguish between different poses, we opted to make the capacitances for different poses as different as possible from each other by maximizing their pairwise distances.

4 SIMULATION

In this section, we formalize the simulation procedure. We first discuss how to generate an appropriate volumetric FEM model for a given sensor layout in Sec. 4.1. To perform the elastic simulation we use PolyFEM [Schneider et al. 2019], an open source FEM framework, which uses the Incremental Potential Contact (IPC) formulation to handle contact [Ferguson et al. 2020; Li et al. 2020]. We discuss this in Sec. 4.2. We explain how to compute sensor capacitances using FEM in Sec. 4.3. Finally, we evaluate the accuracy of our approach in Sec. 4.4.

4.1 Modeling

Layers Geometry. The sensor contains 5 layers of silicone, where the thickness of the i'th layer is t_i. In our case, t_i is on the order of tens of microns. The top, middle, and bottom layers are non-conductive (dielectric) and are made entirely of cured silicone. The remaining two layers, which contain the *plates* of the capacitors, have both conductive and non-conductive silicone parts. Since the sensor is flat, we can describe it by 2D domains. We also assume that Ω_c is the same for both conductive layers. The assumption is valid since non-overlapping conductive parts do not contribute to capacitance considerably and can be neglected. Ω_c is the union of several connected components Ω_{c_i}, each representing a single capacitor c_i. In the following, we will use c_i to refer both to the capacitor and its capacitance. Before fabricating, we connect the components with non-overlapping "bridges" to expose the connections to the capacitors to one side of the sensor: to minimize the number of external connectors, we follow [Glauser et al. 2019a] and use an array layout.

Meshing. To simulate with high accuracy, we opted to use volumetric FEM, requiring us to generate a volumetric tetrahedral mesh. Since the layers are extremely thin, existing meshing tools are likely to fail. Instead, we first generate a compatible triangulation of Ω and Ω_c, and mark the triangles inside Ω_c. We then *extrude* the triangulation out of plane, to create a triangular prism mesh. Each prism can then be split into three tetrahedra, while ensuring that compatibility across prisms is maintained (see [Porumbescu et al. 2005]). This process generates a tetrahedral mesh for *one* layer. To generate all layers, we repeat the process, adjust the thickness of the mesh accordingly, and merge overlapping vertices between layers. Finally, we mark all the elements corresponding to elements of Ω_c in the two conductive layers.

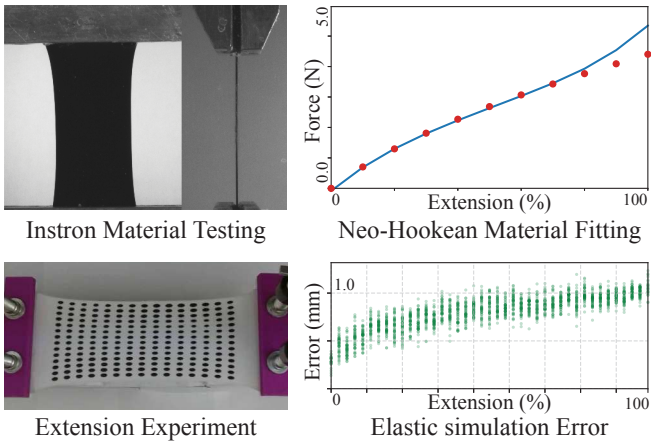


Fig. 3. Experimental determination and validation of material parameters for sensor characterization. Top: Experimental setup and material fitting for the elastic material model (simulation in red, experiment in blue). Bottom: Controlled deformation with the use of an external optical system to track the black ink dots. Each vertical green bar represents one extension experiment and each dot the error of a testing cycle

(the experiment is automated using a motorized linear stage).

4.2 Elastic Simulation

We drive the deformation by defining Dirichlet boundary conditions or by applying external forces through contact. We use the formulation in [Huang et al. 2024], which is differentiable, thus allowing us to optimize sensor designs via continuous optimization in Sec. 5. To achieve an accurate simulation, material parameters must be identified, which we discuss below.

Material Testing/Parameter Identification. We refer to Appendix B for the protocol and materials used to prepare both the clear silicone (non-conductive) and the black silicone (conductive). We attempted to produce standard samples (i.e., cubes of material), but unfortunately, silicone mixed with carbon black does not cure in this form. We instead conduct a uniaxial extension test directly on thin layers using an Instron column testing machine (Fig. 3). We test two thin layer samples made of clear silicone and layered silicone/silicone with carbon black. We fit a neo-Hookean material model to the data, for both the clear silicone and the dark carbon/silicone compound. We found that Poisson's ratio is 0.47 for both, while Young's modulus is $570kPa$ for the clear silicone and $1100kPa$ for the carbon black silicone. These parameters agree with the values found by [Glauser et al. 2019a] in their analysis.

Validation for Elastic Simulation. We further validated the elastic simulation by performing an extension test on a $90mm \times 75mm \times 0.7mm$ thin sheet, for up to 100% extension. We placed markers on a 14×15 grid on the sample and recorded their positions using a calibrated camera for 25 extension cycles. We matched the positions with a simulation and computed the error per extension step (Fig. 3). The maximal error was around $1mm$, or 2%, indicating that the model closely matches the measurements.

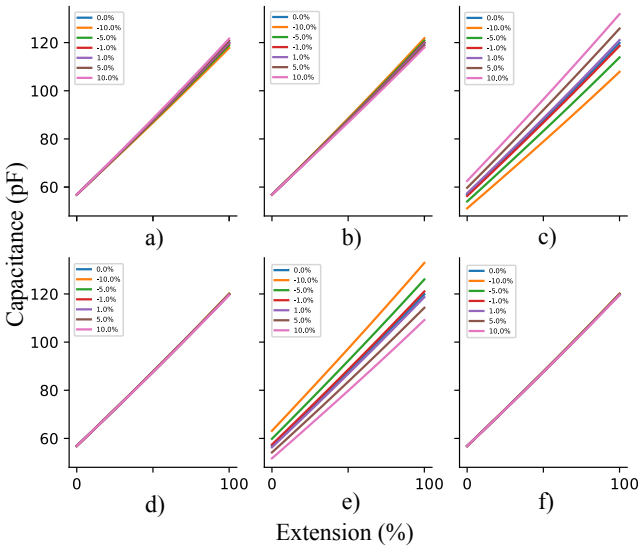


Fig. 4. Effect of varying modeling/simulation parameters (±10%, ±5%, ±1%) on computed capacitances. In a) and b) we vary the Young's modulus of the dielectric and conductive silicone, in c) the dielectric constant, and in d), e), f) the relative widths of the middle three layers.

4.3 Electrostatic Simulation

The next challenge is to calculate capacitance using FEM. For this section, we abstract away the electronic circuitry and assume ideal capacitors.

Electrostratic Primer. Electrostatics is the study of how charged particles or distributions interact. Two like charges q_1 and q_2 exert a repulsive force on one another, which follows an inverse square law, $F \propto E \propto \frac{1}{d^2}$ where F is the force, E is the electric field and d is the distance between the charges. This phenomenon is modeled by Poisson's equation:

$$\nabla^2 \phi = -\frac{\rho}{\epsilon_0} \tag{1}$$

where ϕ is the electric potential ($E = -\nabla\phi$), ρ is the charge density, and ϵ_0 is the vacuum permittivity. In order to increase the capacitance, capacitors are usually embedded in a dielectric medium, which effectively changes the permittivity to a different value ϵ that depends on the specific dielectric.

A conductor is defined by its ability to allow charges to move freely within it. In an electric field, charge will redistribute until reaching equilibrium, where there is no net electric field inside the conductor, i.e., the magnitude of the electric field must be zero. This means that the electric potential must be constant inside a conductor, and in particular on its boundary surface. Therefore, in a charge-free space, outside of a conductor, Poisson's equation turns into a Laplace equation with boundary conditions:

$$\begin{aligned} \epsilon \nabla^2 \phi &= 0 \\ \phi_{\partial\Omega_c} &= u, \end{aligned} \tag{2}$$

where we used Ω_c again to denote the volume of the conductor, and u is the surface potential. Setting u can be easily done in practice by connecting the conductor to a voltage source.

Capacitance Computation. Capacitance is the ability of a system to store electric charge. The mutual capacitance of two conductors is defined as the charge stored divided by the difference in electric potential, $C = q/\Delta V$. This measure of capacitance depends purely on the geometry of the conductors and the dielectric, since q and ΔV are related linearly. Capacitors store energy, given by

$$U = \frac{1}{2} C (\Delta V)^2. \tag{3}$$

The energy stored in an electric field is also the Dirichlet energy electric potential:

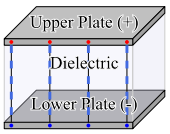
$$U = \frac{1}{2} \int_\Omega \epsilon ||\nabla \phi||^2 dx. \tag{4}$$

Thus, using Eq. (3), two conductors at potentials $\phi_{\partial\Omega_1} = 0V$ and $\phi_{\partial\Omega_2} = 1V$, have the capacitance

$$C = \frac{2U}{(\Delta V)^2} = \int_\Omega \epsilon ||\nabla \phi||^2 dx. \tag{5}$$

Finite Element Solution. Finally, to compute the capacitance, we first solve Poisson's problem in Eq. (2) using PolyFEM to obtain ϕ, and then compute its Dirichlet energy as required in Eq. (5). However, at this point, it is natural to ask if the volumetric simulation is even necessary to compute capacitance. Could we not have used the simple formula for parallel plates as a simple approximation? As we show in Fig. 5, the answer is no, as it is too inaccurate. The reason is that the thickness of the dielectric layer changes as the sensor deforms, which happens in a complicated and non-uniform way that must be modeled through analysis of its elastic behavior.

Measuring Permittivity. To compute the capacitance in simulation, we need to measure the permittivity of the silicone. This can be done by fabricating a simple parallel plate capacitor (see inset). The capacitance in this case is known to be



$$C = \epsilon \frac{A}{d}, \tag{6}$$

where A is the area of the plates and d is the distance between them. To find the permittivity, we fabricate a rectangular silicone capacitor with known dimensions and measure the capacitance. We then take cross sections of the capacitor and measure the average distance between the plates on a calibrated microscope (see Fig. 5). The value we experimentally determined is $\epsilon = 2.84\epsilon_0$.

Grid Capacitance Measurement. In simulation, the capacitance for each element is determined by setting all conductors to $0V$, except for one conductor of the target capacitor, which is set to $1V$. The electrostatic solution is then obtained through Eq. (2), and the capacitance is computed using Eq. (5). This process is repeated for each capacitor element.

The fabricated sensor presents constraints in terms of available wire routing area on the silicone and microcontroller pin count. To address these limitations, we employ the multiplexing technique described in [Glauser et al. 2019a]: for an $n \times n$ capacitor grid, the

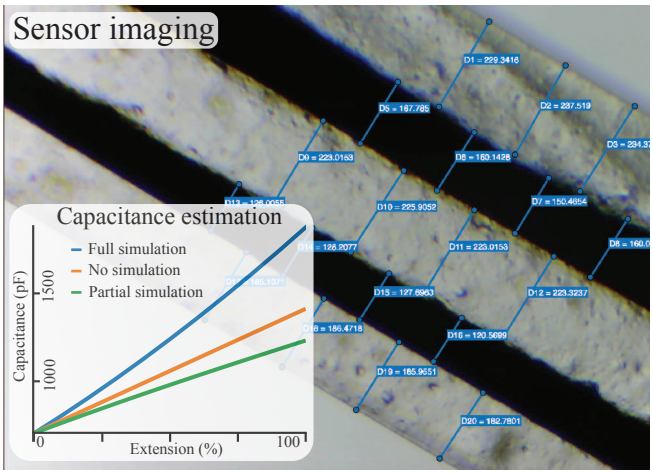


Fig. 5. Cross section of sensor taken using a calibrated microscope. The plot shows a comparison of computing capacitance for a parallel plate capacitor ($8cm \times 8cm$) under 100% extension. The full simulation (solid blue) uses the methodology outlined in the text. The non-physical simulation (dotted orange) scales up the capacitance by the extension, thus not needing an elastic simulation. The physical simulation (dashed green) corrects the area based on the elastic simulation, but does not take into account the change is separation distance.

top conductors are connected column-wise while the bottom conductors are connected row-wise, reducing the required wire count from $2n^2$ to $2n$. Capacitor formation is achieved by partitioning the wires into two groups with shared connections within each group. While we cannot address a single capacitor individually, different capacitor combinations can be activated, and the individual capacitance values can be obtained by solving a linear system, assuming negligible interference between capacitors. Our implementation differs from [Glauser et al. 2019a] in the switching mechanism: we utilize Single Pole Double Throw (SPDT) array ICs (Sec. 5), which introduce parasitic capacitance depending on their state. This requires augmenting the linear system to account for the parasitic contributions of switches in both ON and OFF states, assuming uniform parasitic capacitance within each state. We found that an ON switch contributed $35pF$ of capacitance while an OFF switch contributed $12pF$. For more details, see the discussion in Sec. C.

Parameter Sensitivity Study. We conduct a sensitivity study on the modeling and simulation parameters, specifically on the Young's modulus of the dielectric and conductive silicone, dielectric permittivity, and the thickness of the middle three layers (conductive, dielectric, conductive). We take a simple experiment, a uniaxial extension of a sheet with a single capacitor, and computationally model the capacitance over its extension range for variations of ±10%, ±5%, ±1%, 0% in the parameters. As expected, the capacitance is mostly sensitive to the permittivity of the dielectric and the thickness of the middle layer, although this could vary for different simulation conditions.

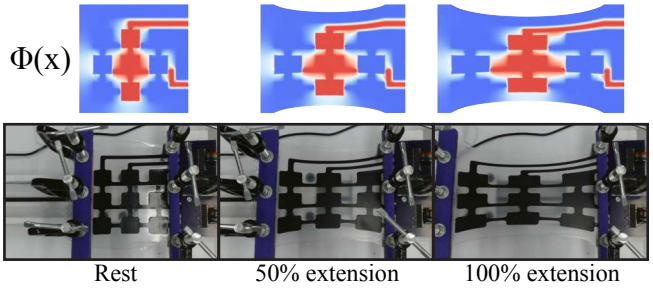


Fig. 6. Extension test and corresponding electrostatic potential visualization for one sensor reading.

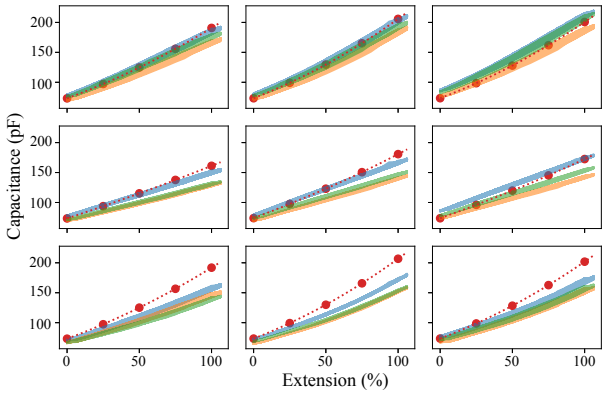


Fig. 7. Quantitative comparison of capacitance values for a 3×3 sensor to over 100% extension: each box represents the capacitance of one of the 9 capacitors. The red line represents the simulation results, the other lines are experimental runs of three different sensors.

4.4 Simulation Example

Uniaxial extension of a 3×3 capacitor array. We validated our entire simulation method (elastic and electrostatic) by performing a uniaxial extension test of a soft capacitor array and comparing it with our simulated result. For the capacitor, we chose a 3 by 3 array of square parallel capacitors of dimension $2cm \times 2cm$ each. From earlier estimations of layer thicknesses, we estimated a capacitance of around $70pF$ per square capacitor. We then performed uniaxial extension beyond 100% of its width for three samples, recording individual capacitances.

We repeated this experiment in simulation and computed individual capacitances at each extension level. We used second-order Lagrangian elements for both the elastic and electrostatic simulations. The results of the simulation and experiment are shown in Fig. 7. In general, we see a good agreement between simulation and experiment; the error in the worst case is around 20%, but this is under extreme extension. We note some differences between samples, suggesting a better fabrication procedure could bring the error even lower. We also see a slight error increase as we moved vertically along the capacitor (each row of the figure) but not horizontally, suggesting there might be other artifacts in the fabrication process.

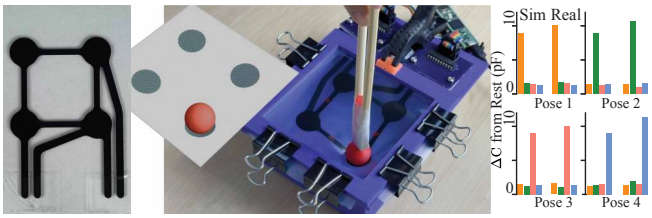


Fig. 8. We fabricate a 2x2 poking sensor to evaluate the accuracy of our simulator. In the histograms, we show the capacitance reading for each of the capacitors in different poses, comparing our simulated results (Sim) with the values measured from our fabricated sensor (Real). Our combined elastic and electrostatic simulation closely matches the measured capacitances.

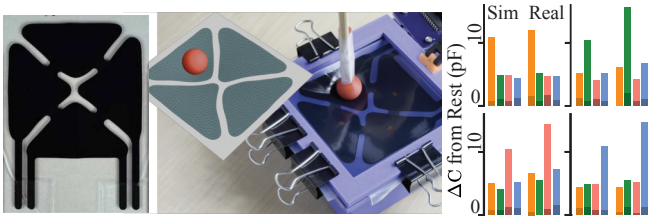


Fig. 9. A poking sensor is optimized to distinguish between 4 poking points arranged in a cross. The starting point of the optimization is the design from Fig. 8. Histograms for the sensor before optimization are shown overlayed, in darker colors. Before optimization, the histograms for all poses are difficult to distinguish. After optimization, the histogram of each pose is unique. The differences between our simulated result (which is used for inverse optimization) and the measured values in our fabricated sample are negligible.

Poking Sensor. We demonstrate how to use our simulator to virtually calibrate a planar poking sensor.

We take a 2×2 capacitor grid on a $10cm \times 10cm$ silicone sheet (see Fig. 8) and fix the ends. Then, if we restrict the deformations to poking on top of the capacitors, we would like to know the capacitance readings of the sensors when each capacitor is pressed. While it may seem obvious that the capacitance of the pressed capacitor will increase, in general, we have no idea by how much. We also do not know whether the other capacitors will change.

We use our method to create a virtual sensor and press it down by $1.3cm$ with a small $2cm$ ball. At rest, the capacitance of each node is computed to be approximately $57pF$. When pressed, the capacitance of that node is predicted to increase by around $9pF$ while the capacitance of the rest of the nodes by $1.3pF$ each. When we do the experiment, we find the capacitor values at rest to be $59.7pF$, $60.4pF$, $53.4pF$, $59.5pF$ (all within 6% of predicted). We poke each of them by hand, trying to match the displacement we see in simulation, and see an average of around $10.5pF$ increase for the pressed sensor and around $1.5pF$ increase in all other sensors. We repeated the procedure 3 times and averaged the values, all of which are very close. Our experimental results are consistent with simulation predictions.

5 SENSOR LAYOUT OPTIMIZATION

Equipped with an accurate differentiable forward simulator, we can consider the problem of optimizing the conductive layouts to increase the effectiveness of the sensor, e.g. its ability to distinguish between different poses.

Problem Formulation. We refer to Fig. 2 for a visual overview of the notation used. We recall (Sec. 4.1) that Ω is the 2D domain representing the shape of the sensor, and $\bigcup_i \Omega_{c_i} = \Omega_c \subset \Omega$ represent an *initial* shape of the capacitors and the conductive layout. Our goal is to deform Ω_c into an optimized shape. We express the deformation using a displacement map $q : \Omega \rightarrow \Omega$, where $q(x) = x + u(x)$. To define $q(x)$ we used a linear blend skinning planar deformer. We provide more detail below, however for the purpose of current discussion we shall treat q as the degrees of freedom of our problem. We let $M = M(q)$ be the mesh generated from $q(\Omega)$, as described in Sec. 4.1. We note that M_c, the mesh for the conductor, has $M_c \subset M$ by construction, that is, it is a sub-mesh of M. Finally, $\bar{M}_c$, the complement of M_c, represents the dielectric medium, i.e. the non-conductive silicone.

Poses. The main application for the sensor is to be glued to a deformable object and detect specific poses. We simulate this application by selecting a triangle mesh representing the object K, and deforming it into various poses K_i, which the sensor will be optimized for. We then "glue" the bottom boundary M_b of M to K and compute a mapping from M_b to K_i. In order to obtain a C_1 mapping between the two triangle meshes, we use the Clough-Tocker interpolant [Renka et al. 1984]. We define the mappings by $P^i(q(x)) \in \mathcal{P}$. These mappings are used to define Dirichlet boundary conditions for the elastic simulator, e.g. $q(x) \rightarrow P^i(q(x))$. We denote the deformation obtained for K_i using $P^i(q(x))$ as boundary conditions

$$M^i = M^i(P^i(q(x)), M(q(x))).$$

Note that M^i depends on $q(x)$ twice: once because affects the deformed pose and again because it affects the rest pose. We remark that since the sensor is thin and flexible, it is generally safe to assume that the object can deform freely, and that the elastic force exerted by the sensor is negligible.

Deformed Sensor Capacitances. For each pose M^i, we can compute the capacitance c^i_j of each capacitor using the electrostatics simulator, as described in Sec. 4.3. To do so, we first compute the corresponding electrostatic potential for M^i and c_j

$$\phi_j \left(M^i(P^i(q), M(q)) \right),$$

where we denote $q = q(x)$ for brevity, using the procedure described in 4.3. Then, the capacitance is found by integrating:

$$c^i_j(q) = c_j \left(M^i(P^i, q) \right) = \int_{\bar{M}^i_c} \epsilon \left\| \nabla \phi_j \left(M^i(P^i, M(q)) \right) \right\|^2 dx.$$

We note again that $c_j(M^i(P^i, M(q)))$ is differentiable w.r.t. q via the chain-rule. This allows us to optimize any smooth objective function of $c^i_j(q)$. Herein, we consider a specific class of objective functions. To define it, we first define $C_i(q) = \left(c^i_j(q) \right)_j$, the vector of

capacitances for pose M^i. This vector can be thought of as a *feature* vector that characterizes M^i.

Objective Function. We define a general objective function by

$$J(q) = d(C_1(q), C_2(q), \cdots) + R(M_c, q),$$

where d is a *dissimilarity* measure, and $R(M_c, q)$ is a regularization term that regularizes the shape of the capacitors and possibly q itself, described below.

Our goal in designing d is to improve the quality of the sensor. Given a pose space with a regular sampling of poses, we'd like the capacitance readings to be sufficiently spaced apart so that from a capacitance vector with possible (systematic and random) errors, we can compute the inverse map with as little error as possible. Consequently, we try to maximize the minimum distance between any two pairs of capacitance vectors from the poses that we are given. To make this a smooth function, we use "LogSumExp" as a smooth maximum function and construct d as follows.

$$d(C_1(q), C_2(q), \cdots) = -\log\left(\sum_{i_1, i_2} e^{-||C_{i_1} - C_{i_2}||_2^2}\right)$$

which has the effect of pushing capacitance vectors apart where they are closest first. We also experimented with simpler objective functions, such as maximizing the squared distance between all capacitance pairs, but found that this failed to remove the capacitance degeneracy of different poses and made it hard to distinguish them in the fabricated experiments.

An important assumption is that we have poses sufficiently well spaced out in pose space. Relaxing this assumption would introduce issues in our objective function, and we leave a more complete optimization over a continuous pose space to future work.

In the following, we include further details about the process.

Taking Derivatives. To optimize J, we need to compute $\nabla_q J$, which requires the gradients of the capacitances of each capacitor in each pose:

$$\nabla_q c_j^i(q) = \frac{\partial c_j^i(M^i)}{\partial M^i}\left(\frac{\partial M^i}{\partial M}\frac{\partial M}{\partial q} + \frac{\partial M^i}{\partial P^i}\frac{\partial P^i}{\partial q}\right).$$

The first term of the chain rule is the change in capacitance w.r.t. change in the *deformed pose* (electrostatic simulation), while the second term is the change in deformed pose w.r.t. changes in q (elastic simulation). The term $\frac{\partial M}{\partial q}$ is trivial since vertices of $M(q)$ are duplicates of q, so this term is effectively the identity. We explain how to compute the rest in the following.

Evaluation of $\frac{\partial P^i}{\partial q}$. The map P^i is defined by establishing a correspondence using barycentric coordinates between two meshes of the same connectivity. These two meshes can be obtained either via a direct simulation or by creating the poses using other deformation techniques such as linear blend skinning. We compute its derivatives with respect to the parameters q using finite differences.

Evaluation of Elastic Shape Derivatives. $\frac{\partial M^i}{\partial M}$ and $\frac{\partial M^i}{\partial P^i}$ are elastic shape and Dirichlet derivatives, respectively, which we compute using the adjoint formulation introduced in [Huang et al. 2024]. This step requires solving an adjoint problem for each pose, which can then be reused to compute the shape derivatives for each capacitor.

Evaluation of Electrostatic Shape Derivatives. The electrostatic system reduces to a Poisson equation, in which case the shape derivatives of the Dirichlet energy, $\frac{\partial C_k}{\partial M^i}$, can be computed according to the derivations set out in [Huang et al. 2024].

Regularization Term. $R(M_c, q)$ is a regularization term made up of (1) a 2D incremental potential contact [Ferguson et al. 2020] that diverges when the capacitors overlap and (2) a Laplacian smoothing term defined on the boundary $\partial\Omega_c$. The first term ensures that two capacitors do not overlap, while the second encourages the boundary of the capacitors not to have spikes, which are problematic for fabrication.

Hierarchical Linear Blend Skinning. To parametrize q we use the hierarchical linear blend skinning approach introduced in [Gjoka et al. 2024]: a set of points are uniformly distributed over the boundary, linear blend skinning weights are automatically computed using [Jacobson et al. 2011], and we then use a translation attached to each point as our solution space. During the optimization, we increase the sample density to provide additional degrees of freedom after the optimization is close to the optimum.

5.1 Optimization Algorithm

We minimize $J(q)$ using an L-BFGS algorithm where we interleave optimization with a remeshing algorithm to control mesh distortion under large deformations.

(1) Initialization: An initial mesh for the sensor is generated using our meshing algorithm 4.1, and we compute an LBS basis for each of our initial control points. The translations parameters of these control points form the q function.
(2) L-BFGS is used on $J(q)$, providing an explicit evaluation of $d_q J$ to the solver. Each evaluation of the shape derivatives requires solving i adjoint solves for elasticity and $i * k$ adjoints for electrostatic, where i is the number of poses, and k the number of capacitors.
(3) After every iteration, we check the quality of the mesh used for Ω^0, and remesh it if any internal angles are less than 10°.
(4) After the optimization converges, we double the number of LBS samples and continue the optimization with a larger space of parameters q. We stop this process when the optimization runs out of cascade steps.

Multi-start Optimization. We experimented with a multi-start optimization strategy, in order to explore more optimal designs. To obtain different initial configurations, we randomly and systematically pick the number of capacitors in the array, and randomly sample their initial location. More precisely,

(1) For $k \in [2, m]$ number of capacitors, we initialize n initial configurations either randomly (80%) or uniformly (20%, using Lloyd's algorithm). These are individually optimized. For our experiments, we used $m = 6$ and $n = 25$.
(2) Using the minimum L^1 norm between all poses as a metric, we pick the optimal final configuration for each k. Then, to choose among different k, we pick the smallest number of capacitors

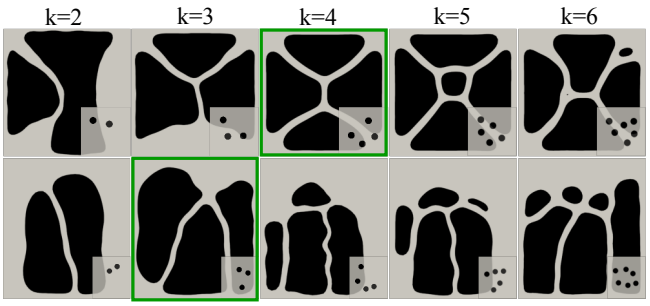


Fig. 10. Result of multi-start optimization on the biaxial stretch experiment (first row) and joystick experiment (second row). The optimal configurations are highlighted, and the initial configuration is in the bottom-right. The number of capacitors, k, is indicated on top of the column.

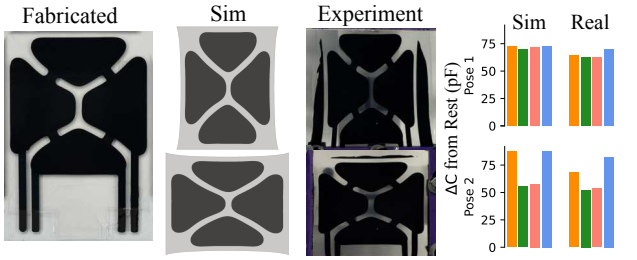


Fig. 11. A sensor is optimized to distinguish between horizontal and vertical uniaxial stretch. We show the optimized fabricated sensor, the simulation of stretching, the corresponding experiment, which yields a good visual match, as well as the capacitance predictions and measurements for each stretching direction.

beyond which the L^1 norm changes very little. Since we have a limited capacitor area and capacitors need to be separated by some distance, we observe that the L^1 norm generally decreases as more capacitors are added.

Some results are depicted in Fig. 10.

6 RESULTS

We use our modeling and optimization methodology to explore a range of sensor designs. First, we optimize sensing of different modalities, such as poking, stretching, and inflation. Then, we demonstrate the performance of our optimized sensors in pose reconstruction for a fabricated joystick, as well as a sensorized wrist and glove in simulation.

Poking Sensor. We again consider the sensor we used to register poking from Section 4.4. This is an effective sensor at detecting poking on top of the capacitors, but it is ineffective at detecting poking in other locations: if we poke in a cross pattern, the readings are not discriminative, as the deformation does not propagate to the capacitors. We optimize this design using our algorithm, obtaining the pattern in Fig. 9. The new pattern leads to distinct measurements for all poses. We note that our simulation (which was used for the optimization) is very accurate: the predicted capacitances at rest

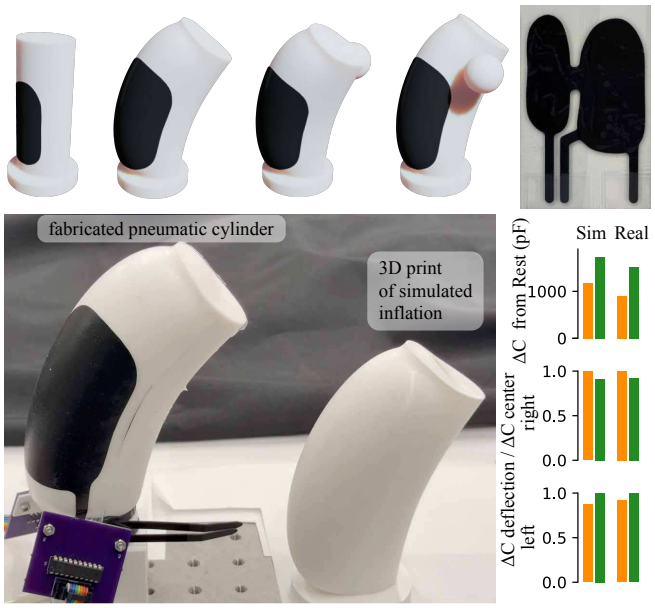


Fig. 12. We optimize a sensor to allow a pneumatically actuated soft robot to determine if there is an obstacle in its surroundings: the deformation on the robot surface changes during inflation depending on the presence of the obstacle (simulation on top row). Our optimization algorithms automatically takes advantage of this difference to find an optimized sensor layout able to sense this difference.

are 294.9pF, 287.6pF, 286.5pF, 301.5pF, while the measured capacitances from the fabricated sensor are 296.0pF, 303.9pF, 267.7pF, 320.8pF, respectively (all within 6.5%).

Biaxial Stretch Sensor. Next, we consider designing a sensor that can distinguish in-plane stretch in the vertical and horizontal directions. At first, it is not clear whether such a design exists, much less what it would look like. We use our multi-start optimization method to explore the design space, ranging from 2 to 6 capacitors (Fig. 10) and pick $k = 4$ as optimal. We validated this in fabrication and ovserve a close match with simulation, indicating that the optimal design indeed does allow biaxial stretch to be distinguished and measured from rest.

Pneumatic Sensor. In the following result, we focus on sensing inflation by augmenting a pneumatic soft actuator with sensing capabilities. The actuator is a silicone cylinder with an offset cylindrical cavity inside. As the inside cavity is pressurized, differences in wall thickness lead to the actuator bending to one side as it expands from the internal pressure. While simulation can predict the inflation behavior of the cylinder, sensing capabilities are needed to determine if there are other objects in the scene that the actuator collides with. We aim to optimize a sensor wrapped outside the cylindrical actuator to determine if the cylinder has collided with a fixed obstacle in the scene. The ball takes two positions, pushing the cylinder to the left or right. We want to design a sensor that determines if the cylinder inflated normally or was pushed to either side. The results of the optimization are given in Fig. 12. From the

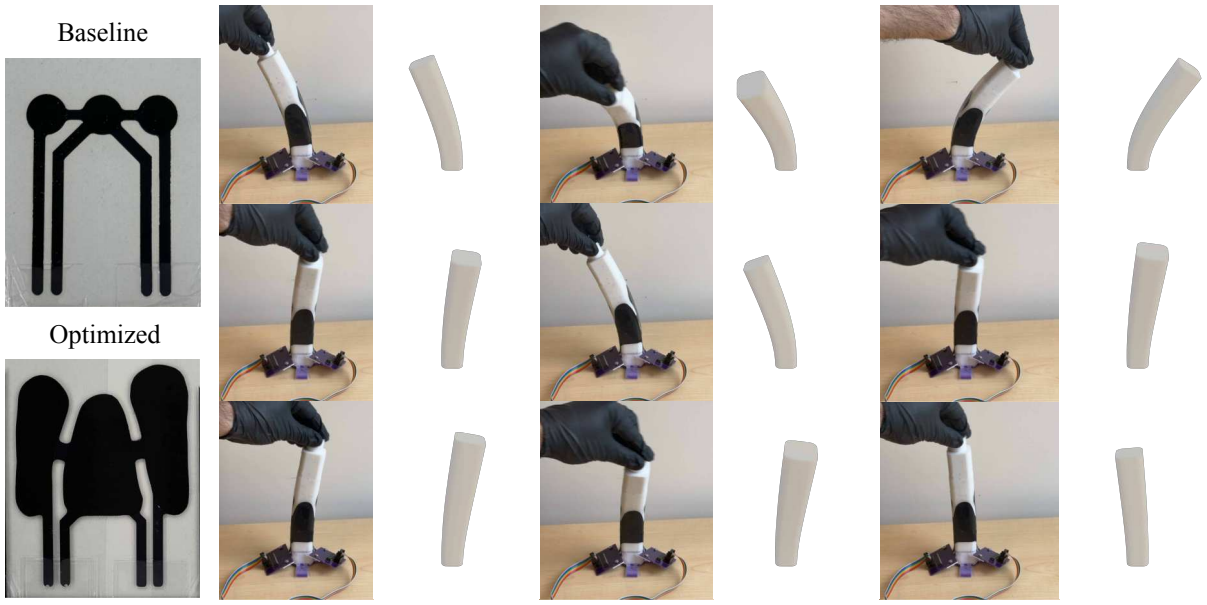


Fig. 13. We add sensing capabilities to a silicone rubber joystick by wrapping a sensor around it. The baseline and optimized sensor patterns are shown on the left. The reconstruction on the right is generated entirely from simulation data, using the optimized design. Comparison with the baseline is in the attached video.

capacitance changes shown in the bar graph, it is clear that inflation from rest increases capacitance significantly. If the actuator collides with an obstacle that causes it to bend laterally, however, significant differences are observed in the left and right side capacitors. This can then be used to sense if the actuator was disturbed from its inflation path and predict the magnitude of this deviation.

Joystick Sensor. Inspired by Tapia et al. [2020], we design a joystick-like object that can sense the direction in which the end effector is moved. In their work, resistors were routed through the cantilever and the change in resistance during deformation was used to estimate the state. In contrast, we aim to tackle this problem using sensors that are placed entirely outside the cantilever, which can be beneficial in settings where we do not have full control of the fabrication process for the actuator. We optimize a sensor to discriminate between 9 principal poses, representing the main directions of movement of the capacitor (center, up, down, left, right, and four diagonals). The results of the optimization are shown in Fig. 13. To evaluate such a joystick, a typical procedure would be to fabricate the sensor, set up a motion capture system, and collect data for hundreds of poses to train a reduced model and predict the poses from capacitance readings. Instead, we create 180 poses in simulation and compute their capacitances. Then, we use a Radial Basis Function (RBF) interpolator from SciPy to fit the simulation capacitance readings to the poses. To test, we measure the capacitances of a fabricated sensorized joystick and use the RBF interpolator to predict the pose data. We repeat this procedure for the optimization initial guess (baseline) as well as the optimized sensor, and we find that the optimized sensor is much more accurate and stable from frame to frame. We show some reconstructed poses in Fig. 13 and the full sequence with baseline comparison in the video.

Fig. 14. We wrap a sensor around a wrist and optimize for its ability to distinguish between 9 poses, some of which are visualized at the bottom along with the optimized sensor. With no intuition on where to initialize the sensor placement, we use our multi-start optimization algorithm to generate optimal designs, shown on top (along with the corresponding initial designs), and pick 2 capacitors as the optimal configuration.

Wrist Sensor. We optimize a sensor placed over a human wrist, targeting an example from [Glauser et al. 2019a], which was used to reconstruct the pose of the wrist. To model this, we use a preexisting rigged human body model, which we manipulate using Blender to generate different wrist poses for the optimization and evaluation.

Due to the complex deformations involved, a good starting configuration for the optimization is not obvious. We use our global optimization procedure to find an optimized layout, considering placing between 2 and 6 capacitors. Our global optimization criteria reveal that the optimal sensor pattern consists of two large capacitors, shown in Fig. 14. To evaluate, we pick 5 random starting

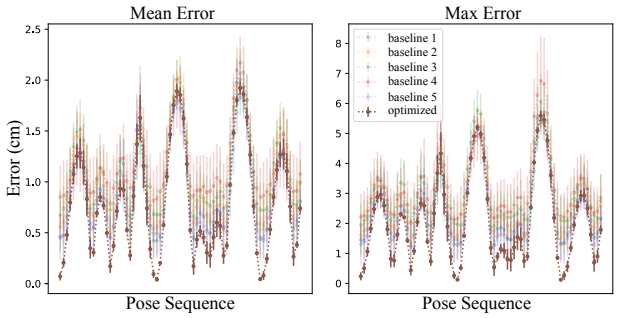


Fig. 15. Mean and max per vertex error of pose reconstruction on the wrist sensor. 100 runs were attempted for each of the baselines and the optimized sensor and we plot here the mean and standard deviation. The optimized sensor performs better than the rest over all of the poses in the sequence, even in out of distribution poses (where errors increase for all sensors).

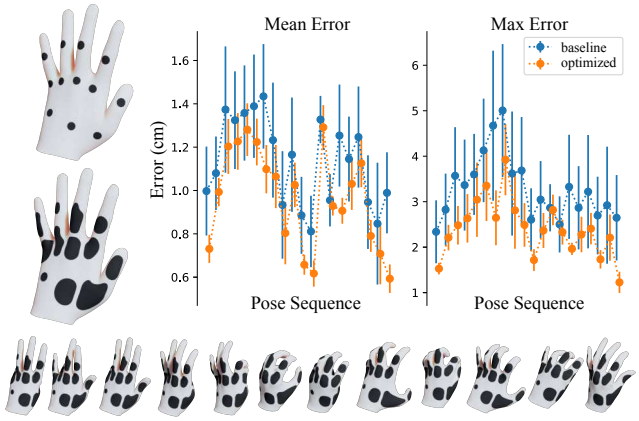


Fig. 16. A sensor is glued to the back of a hand model deformed with linear blend skinning. Our algorithm optimizes for the sensors ability to distinguish between 23 poses (shown at the bottom). The initial and optimized sensors are shown on the left. We test both sensor designs on an evaluation sequence and find that the optimized result performs better in max and mean per vertex error over 100 runs (mean and standard deviation plotted).

configurations as baselines in addition to our optimized example and generate wrist pose data to fit an RBF regressor, as above. Then, we take a new sequence of pose data and evaluate the performance of these regressors fitted on the capacitance data from the baselines and the optimized sensor. We add a physically realistic amount of random noise (see Appendix A) to the evaluation pose capacitances and measure the max and mean per vertex error as compared to the ground truth (repeated for 100 runs). In Fig. 15, we can see that while some baselines perform better than others, the optimized sensor outperforms all of them, with a lower error and smaller error standard deviation. This can also be observed in the full pose reconstruction sequences, shown in the video.

Glove Sensor. We test our optimization methodology on a geometrically more complex example, sensorizing a glove for purposes of hand pose estimation. This was successfully demonstrated by [Glauser et al. 2019b] using these same capacitive stretch sensors, yet an optimal design for the sensor is unknown. Another major limitation is the need to collect data with an expensive custom mocap system the need to repeat this for every new sensor design. We aim to tackle the first problem, optimal design, and remove the need to collect data through simulation. To produce an optimal design, we first take a dataset of scanned, meshed, high quality hand poses from [Romero et al. 2017]. We place the sensor over hand in the hand rest pose and compute an isometric parametrization to the plane. Then, we choose a starting capacitor design, shown in Fig. 16. Since we have some intuition on how a hand deforms, we have a pretty good starting point for the optimization by placing the capacitors in high-stretch areas around the joints. The optimization reveals an interesting final design. To analyze the effectiveness of the optimization, we capture a long sequence of hand pose data using a single camera reconstruction model and a sequence that we use for evaluation. For model simplicity, we use an RBF interpolator as before and compare the performance of the interpolation using the starting and optimized capacitor patterns. We add a physically plausible amount of noise (see Appendix A) and observe that the optimized result outperforms the baseline (Fig. 16) in both the mean and maximum vertex reconstruction error (tested over 100 runs). Full reconstruction comparisons between the baseline and optimized designs is in the video.

7 CONCLUSION

Our simulation and optimization framework opens the door to multiple interesting future directions, including:

(1) *Multiple Layers:* We are currently optimizing for 2 conductive layers, and we assume they have the same geometry. These two restrictions could be lifted to increase the design space, at the cost of a more involved and time-consuming fabrication procedure,
(2) *Pose Space:* Automatic sampling and optimization of the sensor to capture a given parametrized pose space, instead of a set of discrete poses,
(3) *Soft/Rigid Sensors:* Embedding stiffer materials (for example, stiff fibers) in the sensor would provide further control over its mechanical properties, potentially making it more effective at measuring a specific set of loads applied to it.

This framework could be applied to a wide range of sensors. Different materials and more layers require slight changes to the mesh and elastic simulation parameters. While we only show capacitive stretch sensors, arbitrary capacitive sensors could be modeled in the same way to sense traction or compression (though how to fabricate them is an open question). Furthermore, other sensing modalities (resistive, magnetic, electromagnetic, etc.) could be incorporated in a similar way to electrostatics in our framework.

Our main limitation is that the optimization is computationally expensive, requiring minutes for the forward simulation, and hours for the inverse optimization. A possible way to address this would be to use a shell model with varying thickness in lieu of our volumetric approach for the elastic part, which is an interesting avenue for future work. A second limitation is that our optimization requires that a set of distinct poses be produced that are sufficiently spaced

in pose space, however, the optimal design for a set of prescribed poses may not be optimal for the pose space itself. To overcome this, an optimization methodology could be developed that dynamically explores the pose and capacitance spaces. A third limitation is that we are currently doing the routing for the wires connecting the capacitors manually after the design is optimized. This could be automated and accounted for directly by the optimization.

ACKNOWLEDGMENTS

This work was supported in part through the NYU IT High Performance Computing resources, services, and staff expertise. This work was partially supported by the NSF grant OAC-2411349.

REFERENCES

Zhiyong Chen, Qingsuo Wang, Yunce Bi, Juncong Lin, Wei Yang, Chaoyang Deng, Shihui Guo, and Minghong Liao. 2022. Analyzing Human Muscle State with Flexible Sensors. *Journal of Sensors* 2022 (2022). https://doi.org/10.1155/2022/5227955

Hilal M. El Misilmani, Tarek Naous, and Salwa K. Al Khatib. 2020. A review on the design and optimization of antennas using machine learning algorithms and techniques. *International Journal of RF and Microwave Computer-Aided Engineering* 30, 10 (2020), e22356. https://doi.org/10.1002/mmce.22356 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/mmce.22356

Factor2. 2024. Elkem RTV 4420. https://www.factor2.com/product_p/a-rtv-4420.htm

Zachary Ferguson et al. 2020. *IPC Toolkit.* https://ipc-sim.github.io/ipc-toolkit/

Arvi Gjoka, Espen Knoop, Moritz Bächer, Denis Zorin, and Daniele Panozzo. 2024. Soft Pneumatic Actuator Design using Differentiable Simulation. In *ACM SIGGRAPH 2024 Conference Papers* (Denver, CO, USA) *(SIGGRAPH '24)*. Association for Computing Machinery, New York, NY, USA, Article 106, 11 pages. https://doi.org/10.1145/3641519.3657467

Oliver Glauser, Daniele Panozzo, Otmar Hilliges, and Olga Sorkine-Hornung. 2019a. Deformation Capture via Soft and Stretchable Sensor Arrays. *ACM Trans. Graph.* 38, 2, Article 16 (March 2019), 16 pages. https://doi.org/10.1145/3311972

Oliver Glauser, Shihao Wu, Daniele Panozzo, Otmar Hilliges, and Olga Sorkine-Hornung. 2019b. Interactive hand pose estimation using a stretch-sensing soft glove. *ACM Trans. Graph.* 38, 4, Article 41 (July 2019), 15 pages. https://doi.org/10.1145/3306346.3322957

Zizhou Huang, Davi Colli Tozoni, Arvi Gjoka, Zachary Ferguson, Teseo Schneider, Daniele Panozzo, and Denis Zorin. 2024. Differentiable solver for time-dependent deformation problems with contact. *ACM Trans. Graph.* 43, 3, Article 31 (May 2024), 30 pages. https://doi.org/10.1145/3657648

Imerys. 2024. ENSACO® for rubber - High purity carbon blacks for conductivity. https://www.imerys.com/product-ranges/ensaco-rubber

Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. 2011. Bounded Biharmonic Weights for Real-Time Deformation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)* 30, 4 (2011), 78:1–78:8.

Daniel C. Kammer. 1991. Sensor placement for on-orbit modal identification and correlation of large space structures. *Journal of Guidance, Control, and Dynamics* 14, 2 (1991), 251–259. https://doi.org/10.2514/3.20635 arXiv:https://doi.org/10.2514/3.20635

Minchen Li, Zachary Ferguson, Teseo Schneider, Timothy Langlois, Denis Zorin, Daniele Panozzo, Chenfanfu Jiang, and Danny M. Kaufman. 2020. Incremental Potential Contact: Intersection- and Inversion-free Large Deformation Dynamics. *ACM Trans. Graph. (SIGGRAPH)* 39, 4, Article 50 (2020).

Mark Miodownik, Ben Oldfrey, Richard Jackson, and Peter Smitham. 2019. A deep learning approach to non-linearity in wearable stretch sensors. *Frontiers Robotics AI* 6 (2019). Issue APR. https://doi.org/10.3389/frobt.2019.00027

Costas Papadimitriou, James L. Beck, and Siu-Kui Au. 2000. Entropy-Based Optimal Sensor Location for Structural Model Updating. *Journal of Vibration and Control* 6, 5 (2000), 781–800. https://doi.org/10.1177/107754630000600508 arXiv:https://doi.org/10.1177/107754630000600508

Serban D. Porumbescu, Brian Budge, Louis Feng, and Kenneth I. Joy. 2005. Shell maps. *ACM Trans. Graph.* 24, 3 (July 2005), 626–633. https://doi.org/10.1145/1073204.1073239

Daniela Lo Presti, Daniele Bianchi, Carlo Massaroni, Chiara Coricciati, Alberto Rainer, Sergio Silvestri, Alessio Gizzi, and Emiliano Schena. 2024. Optimization and characterization of a 3D-printed wearable strain sensor for respiration and heartbeat measurements. *Measurement: Journal of the International Measurement Confederation* 228 (3 2024). https://doi.org/10.1016/j.measurement.2024.114377

N. H. M. Rais, P. J. Soh, F. Malek, S. Ahmad, N.B.M. Hashim, and P.S Hall. 2009. A review of wearable antenna. In *2009 Loughborough Antennas and Propagation Conference*. 225–228. https://doi.org/10.1109/LAPC.2009.5352373

R. J. Renka, R. L. Renka, and A. K. Cline. 1984. A Triangle-Based C^1 Interpolation Method. *The Rocky Mountain Journal of Mathematics* 14, 1 (1984), 223–237. http://www.jstor.org/stable/44236796

Javier Romero, Dimitrios Tzionas, and Michael J. Black. 2017. Embodied Hands: Modeling and Capturing Hands and Bodies Together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)* 36, 6 (Nov. 2017).

Teseo Schneider, Jérémie Dumas, Xifeng Gao, Denis Zorin, and Daniele Panozzo. 2019. PolyFEM. https://polyfem.github.io/.

Siser. 2024. Juliet - High-Definition Cutter. https://www.siserna.com/juliet/

Hamid Souri, Hritwick Banerjee, Ardian Jusufi, Norbert Radacsi, Adam A Stokes, Inkyu Park, Metin Sitti, and Morteza Amjadi. 2020. Wearable and stretchable strain sensors: materials, sensing mechanisms, and applications. *Advanced Intelligent Systems* 2, 8 (2020), 2000039.

Andrew Spielberg, Alexander Amini, Lillian Chin, Wojciech Matusik, and Daniela Rus. 2021. Co-learning of task and sensor placement for soft robotics. *IEEE Robotics and Automation Letters* 6 (4 2021), 1208–1215. Issue 2. https://doi.org/10.1109/LRA.2021.3056369

MSE Supplies. 2024. Adjustable Film Applicator. https://www.msesupplies.com/products/mse-pro-micrometer-adjustable-film-applicator-for-battery-slurry-

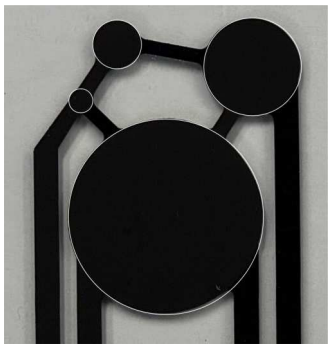


Fig. 17. Noise measurement experiment with doubling capacitor radii. The individual capacitors are circled in white.

Table 1. Runtime info for optimization examples. Simulation time is given in minutes, whereas optimization time is given in hours. Multi-start optimizations were executed on an HPC cluster, whereas everything else was on Mac Mini M4 Pro. Our implementation ran the simulations for each optimization iteration in serial, but they are trivially parallelizable.

Sensor Type	Pose #	Sim Time	Opt Time	Opt Iters
Poking	5	3.5	4	68
Biaxial Stretch	3	1.2	8	136
Pneumatic Inflation	4	2.3	2.5	66
Joystick	9	2.25	24	71
Wrist	9	1.4	13.5	63
Glove	23	1.4	22	42

coating-tape-casting-width-options-50-100-150-200-mm

Toru Takahashi. 2024. Designing gradient coils with the shape derivative and the closed B-spline curves. *Magnetic Resonance Imaging* 110 (July 2024), 112–127. https://doi.org/10.1016/j.mri.2024.03.042

Javier Tapia, Espen Knoop, Mojmir Mutný, Miguel A. Otaduy, and Moritz Bächer. 2020. MakeSense: Automated Sensor Design for Proprioceptive Soft Robots. *Soft Robotics* 7, 3 (2020).

Thomas George Thuruthel, Kieran Gilday, and Fumiya Iida. 2020. Drift-Free Latent Space Representation for Soft Strain Sensors. In *2020 3rd IEEE International Conference on Soft Robotics, RoboSoft 2020*. Institute of Electrical and Electronics Engineers Inc., 138–143. https://doi.org/10.1109/RoboSoft48309.2020.9116021

Robert Turner. 1993. Gradient coil design: A review of methods. *Magnetic Resonance Imaging* 11, 7 (Jan. 1993), 903–920. https://doi.org/10.1016/0730-725x(93)90209-v

Te Yen Wu, Lu Tan, Yuji Zhang, Teddy Seyed, and Xing Dong Yang. 2020. Capacitivo: Contact-based object recognition on interactive fabrics using capacitive sensing. In *UIST 2020 - Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. Association for Computing Machinery, Inc, 649–661. https://doi.org/10.1145/3379337.3415829

A MODELING NOISE

To model the noise for the wrist and glove simulation validations, we shift the capacitance readings in a random range of ±15% per run and add a small amount of random noise per pose, $\pm 2pF$. These magnitudes were determined from a combination of observations from the quantitative 3×3 capacitor experiment and an experiment with increasing capacitor sizes, where we observed noise from the measurement apparatus and the proximity of human skin (at $0.4mm$) independent of the capacitor size (Fig. 17). As a reference for these magnitudes, the parasitic capacitance of the reading circuit was around $30pF$, while for each switch, $C_{ON} = 35pF$ and $C_{OFF} = 12pF$. These assumptions on error magnitude are also supported by the jitter observed in the baseline joystick reconstruction, which appears similar qualitatively to the jitter observed in the baseline wrist and glove reconstructions (sequences in the video) for similar-sized capacitors.

B FABRICATION PROTOCOL

To fabricate the sensors, we largely follow the procedure from [Glauser et al. 2019a], however, we make a few modifications to speed up fabrication. For completeness, we outline the full procedure below.

The sensors are made up of five layers. The main capacitive part is composed of two patterned layers of conductive silicone separated by a dielectric silicone layer so they are not electrically connected. These are then encased on top and bottom by two dielectric layers that protect against mechanical forces on the delicate conductive layer and prevent electrostatic discharge. In order, then, we have: a dielectric protective layer, a conductive layer, a dielectric layer, a conductive layer, and a dielectric protective layer. The exact mixtures of these layers are given in Appendix B.1.

The dielectric silicone layers are made by placing a certain amount of the viscous mixture on a flat glass plate and using a micrometer adjustable film applicator from [Supplies 2024] to scrape a thin layer. The applicator rails rest outside of the sensor, on the glass, so the micrometer is adjusted to the desired cumulative thickness, including the desired layer. The glass plate is then placed in an oven for the solvent to evaporate and the silicone to cure at $43^{\circ}C$ for $1 - 1.5$ hours.

The conductive silicone layers are patterned, unlike the dielectric layers. To make these layers, we use a stencil cutter ([Siser 2024]) to cut patterns on a 4-mil thick Mylar sheet. We place this stencil on an existing dielectric layer, use a spatula to spread the paste-like conductive mixture, and use the film applicator to leave behind a thin layer (it is important to scrape slowly). The sample is then cured in the oven as above, with the stencil embedded. We remove the stencil post-curing and pat down any conductive part that is disturbed from removing the stencil.

To be able to connect to the sensor, we need to have exposed conductive silicone in the final sensor. This is impossible to be done after the fabrication, so we mask the conductive layer leads with a small square of Mylar sheet before covering with the next dielectric layers. After we have fabricated the sensor, we can use a blade directly on the Mylar mask to cut open and extract it without damaging the sensor.

An important note is that viscous fluid effects and solvent evaporation make it such that the final layer thickness depends on the micrometer setting but is not precisely controlled by it, so layer thicknesses must be measured after fabrication for modeling. This can be done once per set of micrometer settings, if the fabrication method is fairly repeatable, which we found to be the case to within 10%. The cumulative micrometer settings we use are: $0.2mm$, $0.3mm$, $0.5mm$, $0.6mm$, $0.8mm$.

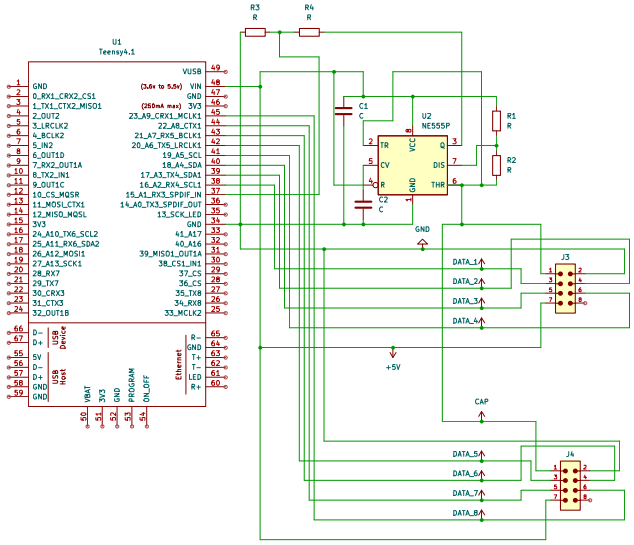


Fig. 18. Capacitance Measurement Circuit

B.1 Silicone Mixtures

Protective layer: Elkem RTV-4420 [Factor2 2024] component A (weight ratio: 1.0) and Ethyl Acetate (1.0) are mixed, then Elkem RTV-4420 component B (1.0) is added.
Conductive layer: First, Imerys Ensaco 250P [Imerys 2024] Conductive Carbon Black (CCB) (0.2) is mixed with Isopropyl Alcohol (IPA) (2.0). The IPA is added in five portions, with the respective ratios of 1.0, 0.4, 0.2, 0.2, and 0.2. After each addition, we stir for about 30 seconds until fully mixed. After all portions have been added, we stir for approximately 1 minute. In a separate glass beaker, we mix the silicone with the same composition and ratio as the protective layer: 1.0 for each of the three components. Finally, we pour the mixed silicone into the container with the CCB and IPA mixture and stir for 3 minutes until fully combined. We wipe off any unmixed or dried CCB from the container's rim and inner walls before use.
Dielectric layer: Same as the protective layer.
Note: When doing this procedure for small batch sizes, care must be taken to minimize residual losses from viscous mixtures sticking to containers.

C CAPACITANCE READING CIRCUIT

C.1 Electronics

We use the same sensing board and principle as [Glauser et al. 2019a], utilizing a TLC555CP in astable mode to create an output square wave whose frequency depends on the capacitance of the sensor we are trying to measure. The microcontroller (MCU) used is a Teensy 4.1, which measures the frequency from the sensing board by measuring the time between rises in the signal, computing the signal period, and then the capacitance. The measurement circuit is given in Fig. 18 and a photo of the final PCB in Fig. 20.

C.2 Measuring Individual Capacitance

As referenced in the main text, we use a multiplexing technique from [Glauser et al. 2019a] to measure individual capacitances in an array. If we consider an $m \times n$ capacitor array, there are mn conductive patches on the top and bottom conductive layers. In the top layer, we connect the conductors in each column together, while in the bottom layer, we connect the conductors in each row together. Each set of connected leads can then be set to either ground (GND) or the MCU logic level (LOGIC, $3.3V$ in this case). We can represent the state of conductors in the top layer by the binary matrix a_{ij} and bottom layer by b_{ij} (either GND or LOGIC). A capacitor is formed when the top and bottom conductors have different voltage levels; formally,

$$c_{ij} = a_{ij} \text{ XOR } b_{ij} \tag{7}$$

where c_{ij} is the state (ON or OFF) of capacitor ij. For example, in the case of a 2×2 capacitor array, setting the first row and first column to LOGIC and the rest to GND, turns on c_{12}, c_{21} and turns off c_{11}, c_{22}. This means that we cannot turn on a single c_{ij}, but must deduce their state from Eq. 7. Given this, we can exploit the fact that the vast majority of the electrostatic potential energy is between individual capacitor plates to approximate individual capacitors as independent with respect to one another. This means that c_{ij} does not affect the state of c_{kl}, and it means that we can setup a linear system $M\gamma = r$, where γ is the vector of individual capacitances, M is a rectangular indicator matrix where every row represents the flattened matrix c_{ij} for each measurement, and r is the vector of capacitance readings for each combination of activated a_{ij} and b_{ij}. To compute γ, then, M must be at least a full rank matrix to be able to solve the linear system. In practice, we setup an overdetermined system as it is more robust to noise.

To do this measurement in a rapid manner, the lead switching between LOGIC and GND needs to happen very quickly. We accomplish this via an ADG333A, a Single Pole Dual Throw (SPDT) switch that is controlled via the MCU. However, it turns out that the SPDT contributes a variable amount of parasitic capacitance to the circuit, depending on if it is ON (C_{ON}) or OFF (C_{OFF}), which violates our assumption that the individual capacitances are independent from one another. From the datasheet, this difference can be on the order of $20 - 30pF$, which overlaps with our measurement range, leading to large errors. To account for this, we add extra entries to γ for C_{ON} and C_{OFF}, as well as entries in each row of M indicating the number of SPDTs that were turned ON and OFF. A diagram of the switching circuit is given in Fig. 19.

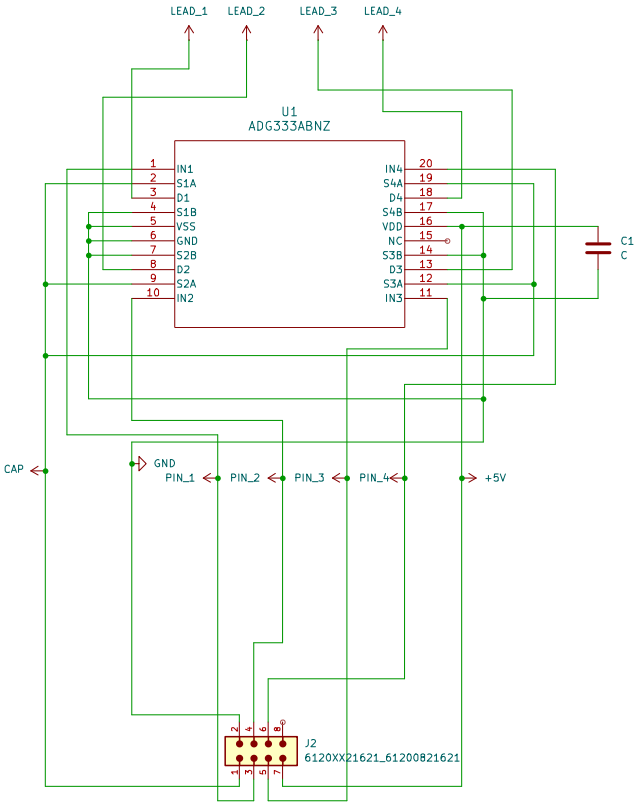


Fig. 19. Switching circuit

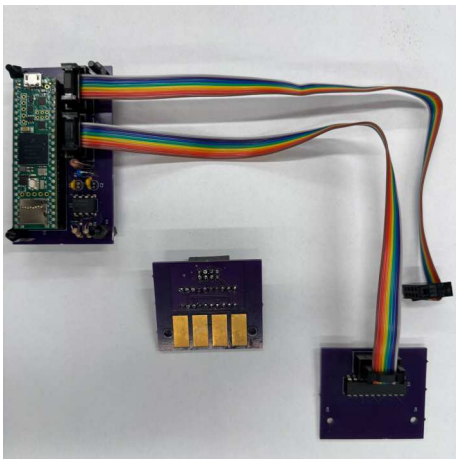


Fig. 20. Fabricated PCBs