3D Force and Contact Estimation for a Soft-Bubble Visuotactile Sensor Using FEM

Jing-Chen Peng[1], *IEEE Student Member*, Shaoxiong Yao[1], *IEEE Student Member*, and Kris Hauser[1], *IEEE Senior Member*

***Abstract*— Soft-bubble tactile sensors have the potential to capture dense contact and force information across a large contact surface. However, it is difficult to extract contact forces directly from observing the bubble surface because local contacts change the global surface shape significantly due to membrane mechanics and air pressure. This paper presents a model-based method of reconstructing dense contact forces from the bubble sensor's internal RGBD camera and air pressure sensor. We present a finite element model of the force response of the bubble sensor that uses a linear plane stress approximation that only requires calibrating 3 variables. Our method is shown to reconstruct normal and shear forces significantly more accurately than the state-of-the-art, with comparable accuracy for detecting the contact patch, and with very little calibration data.**

I. Introduction

Tactile information is needed for robots to perform safe and precise manipulation in emerging domains like healthcare [7] and industrial assembly [19]. Recent years have seen a rapid growth in interest in vision-based tactile sensors like Gelsight [23] and Punyo [1], which can provide high-resolution contact region observations by sensing the deformation internally via embedded cameras. Vision-based tactile sensors can be used to perceive objects' material properties [24] and perform efficient manipulation [18]. However, inferring accurate contact information remains a significant challenge. Model-based methods have been applied to contact force estimation for elastomer sensors [23] and normal force estimation for soft-bubble sensors [12]. Learning-based methods have also been used to address estimation problems for such sensors [9, 14, 24], but typically require large training sets and fail to generalize out of distribution.

In this work, we present a finite-element (FE) based model of a *soft-bubble* vision-based tactile sensor [1] to produce dense contact force and patch estimation using the sensor's internal camera and pressure sensor. In such sensors, a depth camera provides 3D deformation information about an elastic membrane, pressurized with air. Simple image differences are quite poor for computing the contact patch [14] because the bubble's deformation is not restricted to the contact area. Our work is an evolution of Kuppuswamy et al. [12], who proposed an FE model based on the curvature of the bubble. However, they assume a frictionless bubble and their model can only estimate normal contact forces. In contrast, our method can estimate general 3D contact forces (i.e., shear). We use depth and optical flow to approximate the deformed mesh shape and a full 3D FE model of the pressure distribution and membrane stresses (Fig. 1). We formulate contact force estimation as a convex optimization problem maximizing the mesh deformation likelihood with L_1 regularization on contact forces. The force estimator has 3 parameters and can be calibrated well using as few as 6 touch trajectories. Experiments show that our method leads to 36% reduced force estimate error compared to the prior state-of-the-art [12] and comparable accuracy in estimating the contact patch. Code for the project is available online at `https://github.com/uiuc-iml/punyo_force_estimation`.

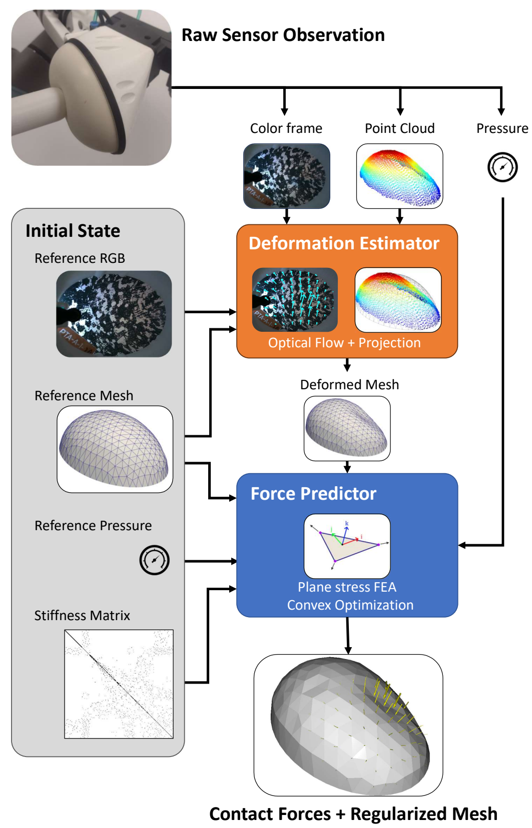

Fig. 1: A summary of the force estimation pipeline. An RGB image and point cloud from the bubble sensor are used to estimate the 3D mesh deformation relative to a reference configuration. We construct a finite element model of the bubble to predict forces from observed deformations and pressure changes. A convex optimization problem combines the physics model and camera observations, giving contact force estimates and regularizing the raw observations.

J.-C. Peng, S. Yao, and K. Hauser are with the Department of Computer Science at the University of Illinois at Urbana-Champaign, Urbana, Illinois, USA. `(jcpeng2,syao16,kkhauser)illinois.edu`

This work is supported by NIFA/USDA Award # 2020-67021-32799.

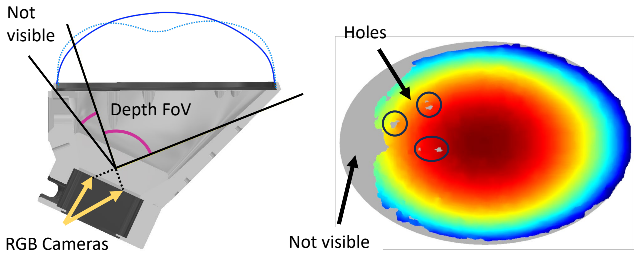

Fig. 2: Left: A schematic of the Punyo soft-bubble sensor used for our experiments, showing the camera layout and visible regions. Right: A typical point cloud reading, colored by Z coordinate. [Best viewed in color]

II. Related Work

Recently, there has been an increasing interest in designing novel tactile sensors [3]. Vision-based tactile sensors such as the Gelsight [23], TacTip [22] and DIGIT [13] can provide high-resolution contact observations and have been proven useful in manipulation tasks [17]. The Punyo *soft-bubble* sensor [1] is an air-inflated bubble with an internally mounted RGBD camera, which has a large sensing area and is resilient to large contact forces. A schematic of the sensor is shown in Figure 2. Due to the richness of sensor data coming from internal cameras, these devices have been used for identifying objects through touch [24] and localization of grasped objects [13, 18]. A more fundamental task is to estimate external contact force and locations across the sensor surface from the observed deformations. By bridging raw observations to physical quantities, such methods enable physics-based reasoning that can be employed in models for higher-level tasks like object pose detection and recognition. Moreover, force estimators could likely be calibrated for a particular sensor with less data than learning estimators for each desired higher-order task.

Existing methods for force estimation can be divided into model-based and data-driven methods. Early work by Ito et al. [11] used the kinematics of dots on the sensor surface to categorize dots as slipping, sticking, or in free space. The initial version of the GelSight [23] has a similar dot pattern that provides high-resolution deformation tracking. To estimate force information, Ma et al. apply inverse FEM to estimate contact forces on the GelSlim 2.0 sensor [15]. They demonstrated that linear FEM can sufficiently capture the relation between deformation and contact forces in a gel-type visuotactile sensor. However, their method models the deforming material using solid elements, which are not suited for modeling a membrane-like sensor architecture. More recently, data-driven methods have been developed to learn a mapping from visual to contact information. Wang et al. [21] use a small neural network to learn a mapping from RGB pixels to surface normals and reconstruct the depth map by solving Poisson's equation. DenseTact 2.0 [5] uses an encoder-decoder network to directly map visual observation to contact deformation and forces. Oller et al. used a PointNet style approach to estimate deformation and external force [16]. Funk et al. [9] used a U-Net structure to directly estimate contact forces from a raw tactile image. While learning-based approaches have seen success, they are very data hungry and time-consuming: For example, Funk et al. [9] train on 277,325 samples across 12 indenting shapes, and DenseTact [5] took more than 10 GPU hours in training. In contrast, we are interested in developing data-efficient methods for force estimation.

For the soft-bubble sensor we studied in this paper, estimating contact information from vision is challenging because contact induces deformation across the entire membrane. To address this challenge, Kuppuswamy et al. use a FE membrane model to simulate the behavior of a soft-bubble sensor and solve the inverse problem to compute contact patches and contact forces [12]. However, their model assumes a frictionless bubble surface, ignoring shear forces. Our model extends their method to 3D mesh deformations and is able to capture general 3D contact forces, improving the contact force estimation by 36% on average while retaining comparable contact patch estimation.

III. Model Based Force Estimation

Our model is based on a triangular mesh discretization of the bubble's surface and estimates contact forces at each node on the mesh. We will refer to the entire mesh as M, an individual triangle as Δ, and the total number of nodes as $|M|$. ∂M will refer to the set of nodes that are on the boundary of M. To estimate contact forces, our model performs three steps: First, the 3D deformation of the bubble's surface is estimated from RGBD camera readings and interpolated to the mesh. Second, a stiffness model is constructed, relating the mesh points' displacements to forces at each mesh node. Finally, a convex optimization problem is solved to estimate contact forces while regularizing for noise introduced from the camera observation and interpolation process. An overview of the process is given in Figure 1.

A. 3D Deformation estimation from RGBD images

To estimate the 3D motion of points on the bubble surface, we combine the depth camera's point cloud with optical flow from the RGB image. We calibrate the camera and indicate 3D to image space projection as $\mathcal{P}(\cdot) : \mathbb{R}^3 \to \mathbb{R}^2$. Given a depth image D, we can also project each pixel to 3D using the interpolated depth map $\mathcal{S}(\cdot) : \mathbb{R}^2 \to \mathbb{R}^3$.

Our model computes deformations relative to a reference configuration, taken when the sensor is inflated and has no external contacts. This consists of a reference mesh M_{ref} with nodes $\mathbf{x}_{ref}$ representing the bubble's reference pose, a reference RGB image for computing optical flow, and a reference pressure for computing pressure differences.

To compute the current position $\mathbf{x}_{cur} \in \mathbb{R}^3$ of a node in the reference mesh $\mathbf{x}_{ref} \in \mathbb{R}^3$, we first project $\mathbf{x}_{ref}$ into image space, computing $\mathbf{r} = P(\mathbf{x}_{ref})$. Then, we use an optical flow map $\mathcal{F}(\cdot) : \mathbb{R}^2 \to \mathbb{R}^2$ to displace $\mathbf{r}$: $\mathbf{r}' = \mathbf{r} + \mathcal{F}(\mathbf{r})$. Finally, we project the displaced pixel into 3D space using $\mathcal{S}$. In summary, the complete mapping from a node's reference position to its deformed position is

$$\mathbf{x}_{cur} = \mathcal{S}(\mathcal{P}(\mathbf{x}_{ref}) + \mathcal{F}(\mathcal{P}(\mathbf{x}_{ref})). \quad (1)$$

$\mathcal{F}$ is computed by interpolating outputs of Farneback's dense flow algorithm [2, 8]. We use optical flow to find pixel correspondences because the *soft-bubble* has a textured pattern on its internal surface [1]. The interpolation of the depth map is performed using SciPy's barycentric interpolation [20], which also helps remove holes in the depth map and fill in occluded regions (See Figure 2).

B. Membrane Model

The goal of the membrane model component is to establish a relationship between deformation of the bubble and their resulting forces. We model the bubble sensor as a homogeneous thin membrane, similar to [12]. Our model considers three types of forces acting on each element of the bubble: Tension forces from neighboring elements, external forces from contact with the environment, and pressure force from the air inside. We consider the bubble deformation to be quasi-static, and solve for static equilibrium:

$$F_{tension} + F_{pressure} + F_{external} = 0. \tag{2}$$

We further linearize Eq. 2 about a fixed reference configuration of the bubble, defined in subsection III-A:

$$\delta F_{tension} + \delta F_{pressure} + F_{external} = 0, \tag{3}$$

where $\delta F_{external} = F_{external}$ since the reference configuration has no external forces acting on it.

We impose these equilibrium conditions at every mesh vertex and solve for values of these quantities at those vertices. For this discussion, $\delta F_i \in \mathbb{R}^{|M|\times 3}$ will denote a matrix of stacked node forces, for each of the three force quantities.

We first lump the change in pressure force to each vertex to compute $\delta F_{pressure}$. We average the areas of triangles incident on a vertex i to approximate the vertex area $a_i = \frac{1}{3}\sum_{\Delta}\{\text{area}(\Delta)\,\forall\,\Delta \in M \,|\, i \in \Delta\}$. Similarly, we compute outward area-weighted vertex normals $\mathbf{n}_i \in \mathbb{R}^3$ for each vertex i. $\delta F_{pressure}$ can be computed as:

$$\delta F_{pressure} = \delta p \begin{bmatrix} a_1\mathbf{n}_1^T \\ a_2\mathbf{n}_2^T \\ \vdots \\ a_{|M|}\mathbf{n}_{|M|}^T \end{bmatrix} \tag{4}$$

where δp is the difference between the observed pressure reading and the reference pressure reading.

$\delta F_{tension}$ is computed by assuming linear elasticity. We characterize the material by its Young's modulus E and Poisson ratio ν. The membrane is thin (0.65mm) compared to its radius of curvature, so we apply the plane stress assumption to simplify the analysis, assuming that the out of plane stress is zero. We also assume that the bubble's bending stiffness is zero. Further, we assume that thickness of the membrane remains constant through the deformation.

For a point on the mesh surface, we define the local coordinate system ijk such that i and j are tangent to the surface and k is perpendicular to the surface. (Refer to Figure 3.) We use $\hat{i}$, $\hat{j}$ and $\hat{k}$ to denote the coordinate directions in global frame. Let σ_{ab} refer to element (a, b) of the stress tensor σ, defined in local coordinates. ε_{xx} refers to the normal strain in direction x, and γ_{xy} refers to the shear strain in plane xy. From the frame of reference of such a point, the linear elasticity equations reduce to:

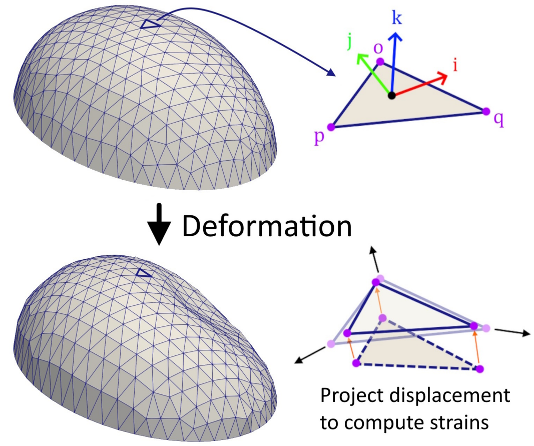

Fig. 3: An illustration of the FEM setup. A local coordinate frame is computed for each triangle, and displacements are projected back to the local ij plane to compute strains. [Best viewed in color]

$$\varepsilon_{ii} = \frac{\sigma_{ii}}{E} - \nu\frac{\sigma_j}{E}, \qquad \varepsilon_{jj} = \frac{\sigma_{jj}}{E} - \nu\frac{\sigma_i}{E}, \\ \gamma_{ij} = \frac{2(1+\nu)}{E}\sigma_{ij}, \tag{5}$$

where the assumptions of no thickness change and no bending stiffness give additional constraints

$$\varepsilon_{kk} = \sigma_{kk} = \sigma_{jk} = 0. \tag{6}$$

These equations are applied on a triangular mesh, treating each triangle individually as a flat surface, following the procedure in [10] (Section 6.2.13).

As part of this simplification, when considering displacements of points in 3D, we project them onto the triangle plane to calculate strain within each individual triangle. Let B be the partial derivative matrix for a linear triangle element in 2D, such that

$$\begin{bmatrix} \varepsilon_{ii} \\ \varepsilon_{jj} \\ \gamma_{ij} \end{bmatrix} = B\begin{bmatrix} \mathbf{u}_{o,2D}^T & \mathbf{u}_{p,2D}^T & \mathbf{u}_{q,2D}^T \end{bmatrix}^T, \tag{7}$$

where $\mathbf{u}_{v,2D}$ refers to the 2D displacement of vertex v. For our simplification, we set

$$\mathbf{u}_{v,2D} = \text{project}(\mathbf{u}_v) = \begin{bmatrix} \hat{i} & \hat{j} \end{bmatrix}^T \mathbf{u}_v, \tag{8}$$

where $\mathbf{u}_v$ refers to the observed 3D displacement of vertex v (Figure 3):

$$\mathbf{u}_v = \mathbf{x}_{v,cur} - \mathbf{x}_{v,ref} \tag{9}$$

The 2D computed node forces are translated back to 3D in an analogous fashion:

$$\mathbf{f}_v = \begin{bmatrix} \hat{i} & \hat{j} \end{bmatrix} \mathbf{f}_{v,2D}. \tag{10}$$

Following the standard FEM assembly procedure, we get a linear map K between node displacements and node tension force changes:

$$\delta F_{tension} = K\left(\begin{bmatrix}\mathbf{u}_1^T & \mathbf{u}_2^T & \dots & \mathbf{u}_{|M|}^T\end{bmatrix}\right) = K(\mathbf{U}) \quad (11)$$

where $\mathbf{U} \in \mathbb{R}^{3|M|}$ denotes a vector of stacked node displacements.

Substituting Eqs. 4 and 11 into Eq. 3, we obtain an expression for the unknown contact force $F_{external}$ as a function of the observed displacements $\mathbf{U}$ and pressure change δp:

$$F_{external}(\mathbf{U}, \delta p) = -\left(K(\mathbf{U}) + \delta p \begin{bmatrix} a_1 \mathbf{n}_1^T \\ a_2 \mathbf{n}_2^T \\ \vdots \\ a_{|M|} \mathbf{n}_{|M|}^T \end{bmatrix}\right) \quad (12)$$

To compute contact pressures $P \in \mathbb{R}^{|M| \times 3}$ at each node, we divide the computed node force by the area at each vertex:

$$P_{contact} = diag(\mathbf{a})^{-1} F_{external}. \quad (13)$$

We additionally define the continuous pressure distribution $\mathcal{P}_{contact}$ by barycentric interpolation of $P_{contact}$ within each triangular face.

The total contact force $\mathbf{f}_{net}$ is computed by summing the node forces across the mesh surface, excluding the boundary:

$$\mathbf{f}_{net} = \sum_{v \in M \setminus \partial M} (F_{external})_v \quad (14)$$

C. Regularization

Although Eq. 12 allows direct computation of contact forces from the observed node displacements in theory, in practice noise from the camera observation and the non-physical nature of the interpolation scheme in Sec. III-A produce strong artifacts in the force prediction. We introduce a term $\delta\mathbf{U} \in \mathbb{R}^{|M| \times 3}$ to correct for noise in the raw observations, and formulate a group LASSO optimization problem:

$$\begin{aligned} \arg\min_{\delta\mathbf{U}} \quad & \|F_{external}(\mathbf{U} + \delta\mathbf{U})\|_{2,1;W_f} + \|\delta\mathbf{U}\|_{2;W_u}^2 \\ \text{s.t.} \quad & \mathbf{u}_i = 0 \text{ if vertex } i \in \partial M \end{aligned} \quad (15)$$

where $\|\cdot\|_W$ refers to the W-weighted norm of a vector $\mathbf{x} \in \mathbb{R}^n$ or matrix $A \in \mathbb{R}^{r \times c}$:

$$\|A\|_{2,1;W} = \left\| W \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_r \end{bmatrix} \right\|_1 \; ; \quad s_k = \sqrt{\sum_{j=1}^{c} A_{jk}^2} \quad (16)$$
$$\|\mathbf{x}\|_{2,W}^2 = \mathbf{x}^\top W \mathbf{x}.$$

Eq. 15 is designed with a $(2,1)$-norm term on the external forces to encourage sparsity in the contact region detection without introducing artifacts involving the coordinate axes. The second 2-norm term regularizes the predicted deformation towards the observed deformations. We formulate and solve this optimization using CVXPY [4].

D. Contact patch estimation

Following [12], we use a fraction of the average contact pressure to estimate contact patches. Define the inward normal contact pressure vector $\mathbf{p} \in \mathbb{R}^{|M|}$ as

$$p_i = -(\mathbf{n}_i)^T \cdot (P_{contact})_i \quad (17)$$

Vertex v on the deformed bubble is considered in contact if

$$p_v > \text{thresh}(\mathbf{p}) = \max\left(k_{const}, k_{linear} \frac{\mathbf{a}^T \mathbf{p}}{\|\mathbf{a}\|_1}\right), \quad (18)$$

with $k_{const} = 2000$ Pa and $k_{linear} = 2$, where $\mathbf{a}^T \mathbf{p} / \|\mathbf{a}\|_1$ is the average contact pressure across the entire bubble. k_{const} is tuned to compensate for noise that is present in the model output due to random fluctuations from the sensor readings.

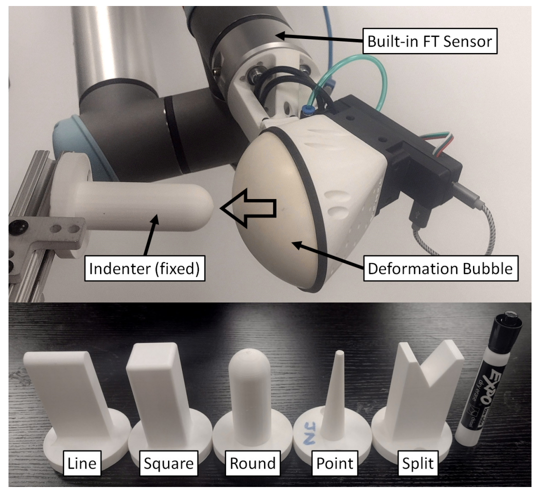

Fig. 4: Photo of the experiment setup. A Punyo sensor is mounted to a UR5 robot, with a built-in force-torque sensor on its end effector. Indenters are mounted to a calibrated, fixed location. The 5 indenters are shown with an EXPO marker for scale. The model was calibrated on 5 trajectories against the Round indenter and one trajectory against the Point indenter, and tested against all five indenters with varying trajectories.

IV. Model Calibration

Calibration of our model involves calibrating the material parameters E, ν and the optimization weight matrices W_f, W_u. To reduce model complexity we set W_f and W_u as diagonal matrices. We specify k_f and k_x as the penalties on the force and displacement terms respectively, and define

$$(W_f)_{ii} = \begin{cases} k_f \frac{|\partial M|}{|M|} & \text{node } i \in \partial M \\ k_f & \text{otherwise,} \end{cases} \quad (19)$$

$$(W_u)_{ii} = \frac{k_x}{|M|}. \quad (20)$$

This parametrization ensures that the magnitude of the force and displacement penalty terms are invariant to mesh element size: To account for the faster growth of surface mesh points compared to boundary points as the mesh becomes more refined, the boundary force penalty must decrease to ensure the total force penalty stays constant.

We fix the Poisson ratio ν at 0.5, modeling rubber as incompressible [6]. This leaves the model with three parameters to calibrate: k_f, k_u, and the Young's modulus E.

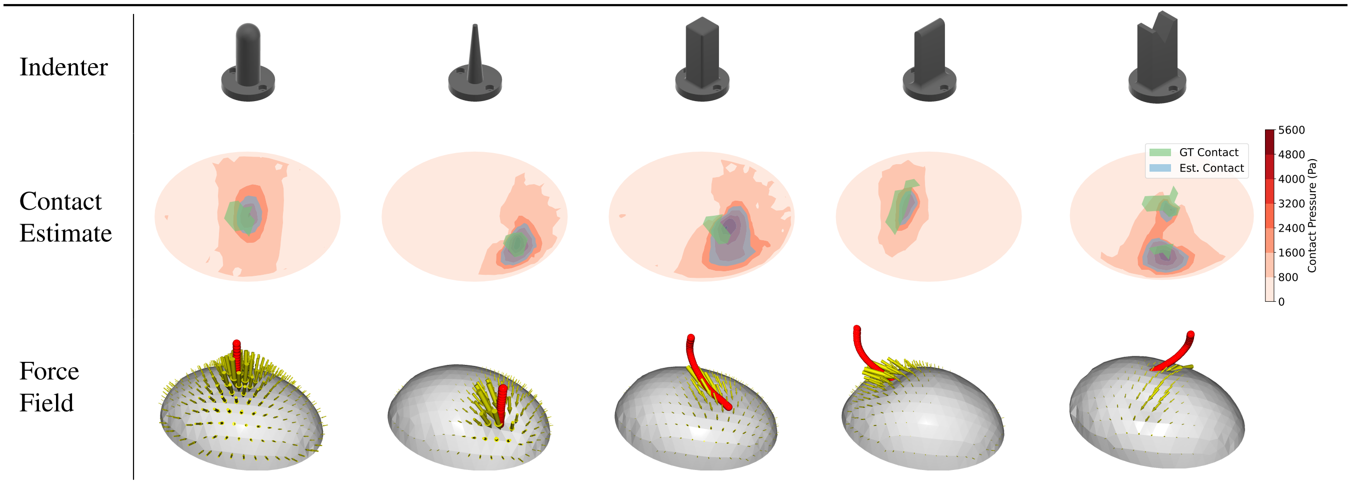

Fig. 5: Qualitative results of model predictions across different indenters and movement trajectories. In contact patch plots, the green region represents the ground truth contact, while the blue region represents the estimated contact patch. The contact trajectory is shown in red. [Best viewed in color]

To calibrate the model, we collected a set of six trajectories of the robot pushing the sensor against an indenter (See Figure 4). Five trajectories were collected with the `round` indenter, and one with the `point` indenter. These indenters were chosen both to calibrate larger touch regions as well as sharp edges. Each trajectory consists of 30-50 data points, for a total of 205 data points. Each data point consists of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection between the indentor geometry, whose pose is estimated by the robot's forward kinematics, and a slightly expanded deformed bubble mesh. A force/torque sensor mounted on the UR5's wrist measures the ground truth contact force. Note that in real-world applications, a robot is equipped with just the Punyo sensor, without a separate force/torque sensor. The model's hyperparameters were optimized by blackbox function optimization [20].

The optimization loss function is a function of the observation $O =$ (RGB image, point cloud, δp), and the ground truth force and contact measurements $\mathbf{f}_{gt}$ and $\mathbf{c}_{gt}$, where $c_{i,gt} = 1$ if node i is in contact:

$$L(O, \mathbf{f}_{gt}, \mathbf{c}_{gt}) = \lambda_1 \left\|\mathbf{f}_{net}(O) - \mathbf{f}_{gt}\right\|_2 + \lambda_2 \frac{1}{|M|} \sum_{i=1}^{|M|} \left\|C_i(O) - c_{i,gt}\right\|_1 \tag{21}$$

$$C_i(O) = \mathrm{sigmoid}\left(k_{opt} \frac{p_i(O) - \mathrm{thresh}(\mathbf{p}(O))}{\mathrm{thresh}(\mathbf{p}(O))}\right)$$

where $\mathrm{sigmoid}(x) = 1/(1+e^{-x})$. λ_1, λ_2, and k_{opt} control the optimizer's penalty on force predictions, contact predictions, and contact prediction aggressiveness respectively. We picked $\lambda_1 = 1\ \mathrm{N}^{-1}, \lambda_2 = 10$, and $k_{opt} = 2$ without tuning and they happened to work well. We optimize the sum of the loss across all observations simultaneously. The resulting optimized values for our set of trajectories are: $k_f = 0.3322\ \mathrm{N}^{-1}, k_u = 537592\ \mathrm{m}^{-2}, E = 1.317$ MPa.

V. Experiments

We conduct experiments to evaluate the force and contact estimation accuracy of our proposed method. We use the set of 5 indentors in Figure 4 to generate a poking dataset, collecting ground truth contact regions and force measurements following the method in Sec. IV. We compare our method against Kuppuswamy et al. [12] and demonstrate more accurate force prediction with comparable contact patch estimation.

A. Dataset Preparation

We sample poking trajectories with diverse locations and contact directions using the following procedure:

1) Waypoint sampling: We sequentially sample three points. One in free space, one near the indenter, and one causing contact between the indenter and the sensor.
2) Sensor direction sampling: We sample a 3D orientation of the sensor which keeps it roughly pointed towards the indentor. We use the same orientation for the entire trajectory.
3) Trajectory validation: Given sampled waypoints and the sensor's target orientation, we use inverse kinematics to check the feasibility of the trajectory.

The center of inflated membrane is used as the reference point to move along each sampled trajectory.

We executed 34 sampled trajectories against each of five different indenter shapes, for a total of 170 testing trajectories. Each trajectory consists of 100 data points, as described in Sec. IV. A summary of the collected trajectories is given in Figure 6, showing that they are fairly diverse and cover a large range of displacements and contact forces in both normal and shear dimensions.

B. Accuracy Evaluation

Results are shown in Table I and Figure 5, comparing our model to the baseline frictionless FE model of Kuppuswamy et al. [12]. Our model is able to produce accurate total contact force predictions both qualitatively and quantitatively. Figure 7 show contact force prediction results aligned with the ground truth contact forces. We significantly reduce the force prediction error by 36% on average because our model can capture shear forces well. Our method is able to predict force

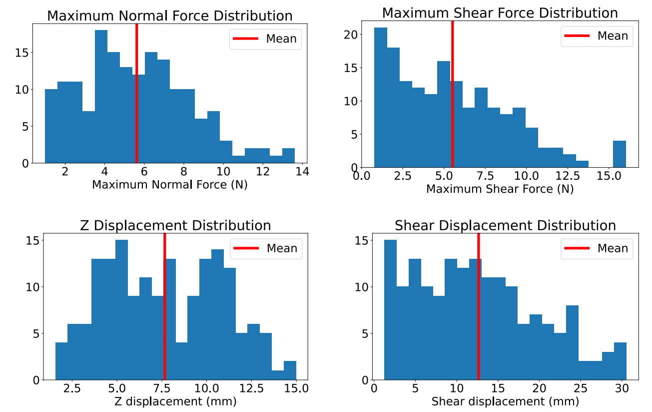

Fig. 6: Summary of collected trajectory data. Collected trajectories cover light and heavy touches, with varying shear and normal components.

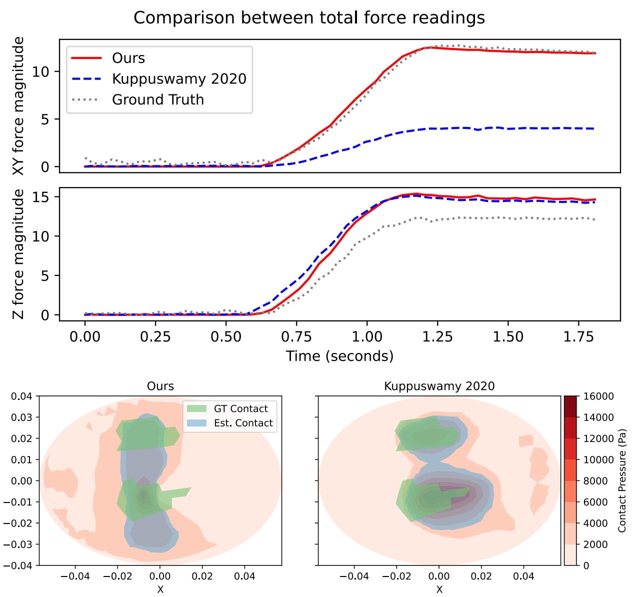

Fig. 7: Qualitative comparison between our model and [12] using the fifth indentor illustrated in Fig. 5. Our model predicts more accurate XY forces by accounting for shear effects. However, our model is worse at distinguishing separate contact regions. [Best viewed in color]

distributions that reflect the general nature of the contact, as seen in Figure 5. The choice of L_1 regularization lets the optimization more easily reject noise introduced by the sensor, but also tends to reject weaker touches: Our model tends to underestimate or fail to detect lighter contacts (net contact force ~ 1 N). The linear plane stress approximation behaves very poorly in the presence of large curvature changes. These effects result in slightly worse contact patch prediction accuracy compared to the baseline, resulting in a 10% reduction in mIoU (mean Intersection over Union) (visible in Figure 7 and the `square` column of Figure 5).

C. Runtime and Scaling

Our definition of k_f and k_w allows the same set of tuned constants to be used with different mesh resolutions. We tested three different mesh resolutions, coarse ($|M| = 227$), medium ($|M| = 390$), and fine ($|M| = 749$), shown in Figure 8. Our pipeline runs in near-realtime with the coarse

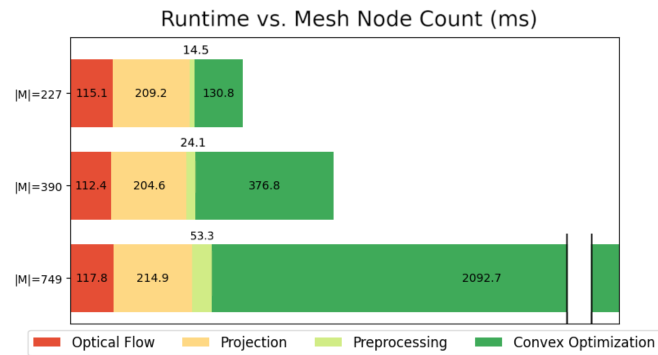

Fig. 8: Average running time breakdown of our model at different mesh sizes. At coarser discretizations the model runs in near real time (~ 2 Hz). The time taken to solve the convex optimization problem scales quickly with the number of mesh nodes. [Best viewed in color]

TABLE I: Comparison of force and contact predictions. Our method performs worse than the baseline at contact detection but gives significantly more accurate force estimation.

	Ours		Kuppuswamy et al. [12]	
Geometry	Contact mIoU ↑	Avg. Force error (N) ↓	Contact mIoU ↑	Avg. Force error (N) ↓
Round	0.528	**1.04**	**0.597**	1.34
Point	0.530	**1.16**	**0.555**	1.77
Line	0.542	**1.18**	**0.574**	1.81
Square	0.485	**1.33**	**0.540**	2.25
Split	0.478	**1.52**	**0.537**	2.56
Average	0.513	**1.24**	**0.560**	1.95

or medium mesh ($1-2$ Hz), while the fine mesh requires a $\sim 4\times$ increase in computation time.

We observe the accuracy of our model does not change significantly as mesh resolution increases, as the total predicted force stays consistent. The predicted contact area tends to behave similarly, with only small gains in resolution. Hence, the data and figures in this paper reflect the outputs from the medium mesh ($|M| = 390$).

VI. Conclusion

We presented a finite element force estimation method for soft-bubble grippers with only three parameters that can be calibrated with small amounts of data. Our model can run in near real-time and produce force predictions with accuracy beyond the current state of the art, especially for shear forces. In future work, we hope to develop a more accurate physical model for the bubble's deformation: Our current bubble model uses a simplified model of membrane deformations, ignoring any effect from changes in the bubble's curvature or from large displacements changing the orientation of individual mesh elements. Higher order elements including curvature effects should also improve accuracy. We also hope to achieve speed improvements by implementation in a compiled language.

VII. Acknowledgements

We would like to thank Patrick Naughton, James Nam, Yifan Zhu, and Mengchao Zhang for help in proofreading and editing this paper. We thank Toyota Research Institute for loaning us the Punyo soft-bubble sensor.

REFERENCES

[1] A. Alspach, K. Hashimoto, N. S. Kuppuswamy, and R. Tedrake, "Soft-bubble: A highly compliant dense geometry tactile sensor for robot manipulation," *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, pp. 597–604, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:102336499.

[2] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[3] C. Chi, X. Sun, N. Xue, T. Li, and C. Liu, "Recent progress in technologies for tactile sensors," *Sensors*, vol. 18, no. 4, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/4/948.

[4] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[5] W. K. Do, B. Jurewicz, and M. Kennedy, "Densetact 2.0: Optical tactile sensor for shape and force reconstruction," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 549–12 555.

[6] Engineering Toolbox, *Poisson's ratio*, 2008. [Online]. Available: https://www.engineeringtoolbox.com/poissons-ratio-d_1224.html.

[7] Z. Erickson, H. M. Clever, G. Turk, C. K. Liu, and C. C. Kemp, "Deep haptic model predictive control for robot-assisted dressing," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 4437–4444.

[8] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Image Analysis*, J. Bigun and T. Gustavsson, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 363–370.

[9] N. Funk, P. O. Müller, B. Belousov, A. Savchenko, R. Findeisen, and J. Peters, "High-resolution pixelwise contact area and normal force estimation for the gelsight mini visuotactile sensor using neural networks," in *Embracing Contacts - Workshop at ICRA 2023*, 2023. [Online]. Available: https://openreview.net/forum?id=dUO0QQw4FW.

[10] T. J. R. Hughes, *The finite element method : linear static and dynamic finite element analysis*, eng. Mineola, NY: Dover Publications, 2000 - 1987.

[11] Y. Ito, Y. Kim, and G. Obinata, "Contact region estimation based on a vision-based tactile sensor using a deformable touchpad," *Sensors*, vol. 14, no. 4, pp. 5805–5822, 2014. [Online]. Available: https://www.mdpi.com/1424-8220/14/4/5805.

[12] N. Kuppuswamy, A. Castro, C. Phillips-Grafflin, A. Alspach, and R. Tedrake, "Fast model-based contact patch and pose estimation for highly deformable dense-geometry tactile sensors," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1811–1818, 2020.

[13] M. Lambeta, P.-W. Chou, S. Tian, B. Yang, B. Maloon, V. R. Most, D. Stroud, R. Santos, A. Byagowi, G. Kammerer, *et al.*, "Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3838–3845, 2020.

[14] M. Lambeta, H. Xu, J. Xu, P.-W. Chou, S. Wang, T. Darrell, and R. Calandra, "Pytouch: A machine learning library for touch processing," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 13 208–13 214.

[15] D. Ma, E. Donlon, S. Dong, and A. Rodriguez, "Dense tactile force estimation using gelslim and inverse fem," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 5418–5424.

[16] M. Oller, M. P. i. Lisbona, D. Berenson, and N. Fazeli, "Manipulation via membranes: High-resolution and highly deformable tactile sensing and control," in *Proceedings of The 6th Conference on Robot Learning*, K. Liu, D. Kulic, and J. Ichnowski, Eds., ser. Proceedings of Machine Learning Research, vol. 205, PMLR, 2023, pp. 1850–1859. [Online]. Available: https://proceedings.mlr.press/v205/oller23a.html.

[17] U. H. Shah, R. Muthusamy, D. Gan, Y. Zweiri, and L. Seneviratne, "On the design and development of vision-based tactile sensors," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 82, 2021. [Online]. Available: https://doi.org/10.1007/s10846-021-01431-0.

[18] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson, "Cable manipulation with a tactile-reactive gripper," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1385–1401, 2021. [Online]. Available: https://doi.org/10.1177/02783649211027233.

[19] B. Tang, M. A. Lin, I. A. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, and Y. S. Narang, "IndustReal: Transferring Contact-Rich Assembly Tasks from Simulation to Reality," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 2023.

[20] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[21] S. Wang, Y. She, B. Romero, and E. H. Adelson, "Gelsight wedge: Measuring high-resolution 3d contact geometry with a compact robot finger," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6468–6475, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235446468.

[22] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. Giannaccini, J. Rossiter, and N. Lepora, "The tactip family: Soft optical tactile sensors with 3d-printed biomimetic morphologies," *Soft Robotics*, vol. 5, Jan. 2018.

[23] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, vol. 17, no. 12, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/12/2762.

[24] W. Yuan, C. Zhu, A. Owens, M. A. Srinivasan, and E. H. Adelson, "Shape-independent hardness estimation using deep learning and a gelsight tactile sensor," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 951–958.