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

Jing-Chen Peng¹, IEEE Student Member, Shaoxiong Yao¹, IEEE Student Member, and Kris Hauser¹, 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.
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 shown 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 $\Delta$, and the total number of nodes as $|M|$. $\partial M$ 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 $P(\cdot):\mathbb{R}^3 \rightarrow \mathbb{R}^2$. Given a depth image $D$, we can also project each pixel to 3D using the interpolated depth map $S(\cdot):\mathbb{R}^2 \rightarrow \mathbb{R}^3$.

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

$$x_{cur} = S(P(x_{ref}) + F(P(x_{ref}))).$$ (1)
\( F \) 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:

\[
\delta F_{\text{tension}} + \delta F_{\text{pressure}} + \delta F_{\text{external}} = 0. \tag{2}
\]

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

\[
\delta F_{\text{tension}} + \delta F_{\text{pressure}} + \delta F_{\text{external}} = 0, \tag{3}
\]

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

\[
\delta F_{\text{pressure}} = \delta p \begin{bmatrix}
    a_1 \hat{n}_1^T \\
    a_2 \hat{n}_2^T \\
    \vdots \\
    a_M \hat{n}_M^T
\end{bmatrix} \tag{4}
\]

where \(\delta p\) is the difference between the observed pressure reading and the reference pressure reading.

\(\delta F_{\text{tension}}\) is computed by assuming linear elasticity. We characterize the material by its Young’s modulus \(E\) and Poisson ratio \(\nu\). 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 \(i\), \(j\) and \(k\) to denote the coordinate directions in global frame. Let \(\sigma_{ij}\) refer to element \((a,b)\) of the stress tensor \(\sigma\), defined in local coordinates. \(\varepsilon_{xx}\) refers to the normal strain in direction \(x\), and \(\gamma_{xy}\) refers to the shear strain in plane \(xy\). From the frame of reference of such a point, the linear elasticity equations reduce to:

\[
\varepsilon_{ii} = \frac{\sigma_{ii}}{E} - \nu \frac{\sigma_{jj}}{E}, \quad \varepsilon_{jj} = \frac{\sigma_{jj}}{E} - \nu \frac{\sigma_{ii}}{E}, \quad \gamma_{ij} = \frac{2(1 + \nu)}{E} \sigma_{ij}, \tag{5}
\]

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 plane to calculate strain within each individual triangle. Let \(B\) be the partial derivative matrix for a linear triangle element in 2D, such that

\[
\begin{bmatrix}
    \varepsilon_{ii} \\
    \varepsilon_{jj} \\
    \gamma_{ij}
\end{bmatrix} = B \begin{bmatrix}
    u_{v,2D}^T \\
    u_{p,2D}^T \\
    u_{q,2D}^T
\end{bmatrix}^T, \tag{7}
\]

where \(u_{v,2D}\) refers to the 2D displacement of vertex \(v\). For our simplification, we set

\[
u_{v,2D} = \text{project}(u_v) = \begin{bmatrix} i & j \end{bmatrix}^T u_v, \tag{8}
\]

where \(u_v\) refers to the observed 3D displacement of vertex \(v\) (Figure 3):

\[
u_v = x_{v,cur} - x_{v,ref} \tag{9}
\]

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

\[
\nu = \begin{bmatrix} i & j \end{bmatrix} f_{v,2D}. \tag{10}
\]
Following the standard FEM assembly procedure, we get a linear map $K$ between node displacements and node tension force changes:

$$\delta F_{\text{tension}} = K\left(\begin{bmatrix} u_1^T & u_2^T & \ldots & u_{|M|}^T \end{bmatrix}\right) = K(U)$$  \hspace{1cm} (11)

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

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

$$F_{\text{external}}(U, \delta p) = -K(U)\frac{1}{\delta p}\begin{bmatrix} a_{11}u_1^T \\ a_{21}u_2^T \\ \vdots \\ a_{|M|}u_{|M|}^T \end{bmatrix}$$ \hspace{1cm} (12)

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

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

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

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

$$f_{\text{net}} = \sum_{v \in M\setminus\partial M} (F_{\text{external}})_v$$ \hspace{1cm} (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 $\delta U \in \mathbb{R}^{3|M|\times 3}$ to correct for noise in the raw observations, and formulate a group LASSO optimization problem:

$$\arg\min_\delta U \left\| F_{\text{external}}(U + \delta U) \right\|_2, W_f + \|\delta U\|_2, W_u$$

s.t. $u_i = 0$ if vertex $i \in \partial M$

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

$$\|A\|_{2,1;W} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_k \end{bmatrix}, \quad s_k = \sum_{j=1}^c A_{jk}^2$$ \hspace{1cm} (16)

$$\|x\|_{2,2;W}^2 = x^T W x.$$  

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 $p \in \mathbb{R}^{3|M|}$ as

$$p_i = -(n_i^T \cdot (P_{\text{contact}}))_i$$ \hspace{1cm} (17)

Vertex $v$ on the deformed bubble is considered in contact if

$$p_v > \text{thresh}(p) = \max(k_{\text{const}}, k_{\text{linear}} \frac{a^T \theta}{a_1})$$ \hspace{1cm} (18)

with $k_{\text{const}} = 2000$ Pa and $k_{\text{linear}} = 2$, where $a^T \theta/\|a\|_1$ is the average contact pressure across the entire bubble. $k_{\text{const}}$ 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 $E, \nu$ 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

$$(W_f)_{ii} = \begin{cases} k_f & \text{node } i \in \partial M \\ k_f & \text{otherwise} \end{cases}$$ \hspace{1cm} (19)

$$(W_u)_{ii} = \frac{k_x}{|M|}.$$ \hspace{1cm} (20)

This parameterization 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 $\nu$ 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 $E$. 

---

[4]: CVXPY library
[6]: Incompressible rubber model
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 indenter shapes 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 of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection of an RGBD image, pressure reading, and ground truth contact. The contact patch is computed as the intersection of an RGBD image, pressure reading, and ground truth contact.

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 shows 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 $\sim 1$ 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 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


