Semi-Empirical Simulation of Learned Force Response Models for Heterogeneous Elastic Objects

Yifan Zhu[1], Kai Lu[2], and Kris Hauser[1]

***Abstract*— This paper presents a semi-empirical method for simulating contact with elastically deformable objects whose force response is learned using entirely data-driven models. A point-based surface representation and an inhomogeneous, nonlinear force response model are learned from a robotic arm acquiring force-displacement curves from a small number of poking interactions. The simulator then estimates displacement and force response when the deformable object is in contact with an arbitrary rigid object. It does so by estimating displacements by solving a Hertzian contact model, and sums the expected forces at individual surface points through querying the learned point stiffness models as a function of their expected displacements. Experiments on a variety of challenging objects show that our approach learns force response with sufficient accuracy to generate plausible contact response for novel rigid objects.**

I. INTRODUCTION

Robotic manipulation of deformable objects has a number of applications, such as in-home environments, cable routing, textile handling, handling packaging in automated warehouses, and soft tissue modeling in biomedical settings, but it remains a topic of active research. Common deformation simulation methods, such as finite element method (FEM) or mass-spring modeling (MSM), require an object's geometry and its material properties to be represented by a volumetric mesh model. Mesh models are challenging to build from noisy sensor data, and calibrating appropriate material constitutive parameters is a tedious, time-consuming, and error-prone process that depends on an appropriate mesh topology and constitutive equations. Data-driven approaches have been studied in robotic palpation of biological tissue, where a robotic probe gathers a dense field of data to estimate tissue stiffness and identify anomalies.

We envision that robots could use *naturalistic* data acquisition to model deformable objects, in which behavior is observed from a small number of interactions and the model is improved continuously during interaction. Humans can very quickly generalize from the data gathered during manipulation to predict plausible outcomes in vastly different scenarios. This paper takes preliminary steps toward providing robots with similar capabilities. We take a *semi-empirical* approach that integrates machine learning with physics. This general idea is appealing because learning from data can enhance physics modeling when the physics of certain phenomenon are not well understood [1]–[3], when the parameters for the governing physics equations are hard to identify [4], [5], or when applying the physics equation is computationally prohibitive [6], [7]. In our approach, the simulator 1) accepts a point-based model of the object surface's force response under displacements, which are learned from data, and then 2) calculates a resultant contact wrench for a novel rigid object using analytical calculations. Our model is quite general, as its deformation response can be heterogeneous and nonlinear. Moreover, the visuo-tactile model constructed by our method can overcome camera calibration error, sensor noise, and "fuzzy" objects with ill-defined boundaries. It is important to note that *we do not expect a robot to need to touch an object hundreds of times during naturalistic interaction*. Our experiments examine performance under a tiny fraction of pokes as training data. Even with only a few touches, reasonably predictive models can be learned, which suggests that our pipeline is promising for on-line use in naturalistic interaction.

We evaluate this approach using an experimental testbed

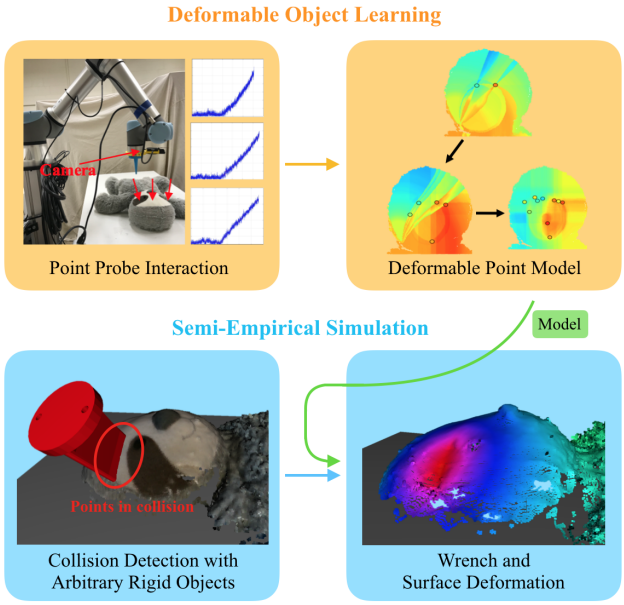


Fig. 1. Flow of our method. A point interaction model of a deformable object is first learned through data acquired with the system shown here after a small number of pokes. The model is then used in a semi-empircal simulator that solves for the deformation and contact wrench with an arbitrary rigid object. The color of the deformed object represents the amount of surface deformation.

*This work was supported in part by NSF grants NRI-#1527826 and NRI-#1830366.

[1]: Y. Zhu and K. Hauser are with the Departments of Computer Science, University of Illinois at Urbana-Champaign, IL, USA. {yifan16, kkhauser}@illinois.edu

[2]: K. Lu is with the Department of Automation, Tsinghua University, Beijing, China. lu-k16@mails.tsinghua.edu.cn

(Fig. 1) consisting of a robot arm equipped with a RGB-D camera, a tapered probe attached at the end-effector, and a force-torque (F/T) sensor. Experiments show that our approach can learn a point deformation model that predicts force response 0.49 N of root mean squared error (RMSE) over 5 test objects with 10 interactions each. When making contact with novel rigid probes, which are a line-shaped paddle and a cylinder, our simulator's force prediction has 1.5 N RMSE. This accuracy is comparable to learned models obtained with the same number of interactions. We also show an application of our method to planning for packing a rigid object on top of a deformable object into a box.

II. Related Work

Deformable object modeling has long been studied in various fields including computer graphics, continuum mechanics, and biomechanics. An overview of the existing methods on deformable object modeling can be found in Ref. [8]. It still remains a challenge to build models that match real-world objects accurately both in deformation and force response.

Several researchers have studied the problem of building elastic models of real-world object using experimental data. Frank et al. attempt to build homogeneous FEM models by first collecting force probing data using a mobile robot arm, a thin probe, an F/T sensor and a RGBD camera [9]. An error minimization approach is used to estimate the FEM model parameters such that the simulations match recorded data. Boonvisut and Cavusoglu follow a similar scheme to build FEM models for elastic objects under stretching [10]. Both methods require tens of minutes to perform the error minimization, making it less likely to be used online. Bickle et al apply error minimization to inhomogeneous, nonlinear FEM models [11]. Radial basis functions (RBF) are used to interpolate the strain-stress relationship in each finite element, and error minimization optimizes the weights used for RBF interpolation to match observations. Although being able to handle more complex materials, the error minimization step is prone to being stuck in local minima and the data acquisition process requires complex apparatus setup, including multiple cameras, carefully positioned and tuned light sources, and drawing markers on the testing objects.

Empirical data have also been used to tune MSM and meshless methods. Deussen et al use simulated annealing to estimate the elastic properties of an MSM object model, given data from simulation experiments [12]. Burion et al solve a similar problem using a particle filter [13]. Pauwels et al. build meshless models of homogeneous 2D foams whose shape upon contact with a rigid object depends on a deformability constant. This constant is estimated by error minimization of real-world data. Because the method only simulates deformable shape, it is unable to predicte forces at contacts.

A highly related topic is robotic palpation. Yamamoto et al use a robot equipped with an F/T sensor to poke at discrete locations on flat phantom tissue to estimate the stiffness across the object surface and find anomalies [14]. Salman et al. adopt a similar approach but select the next poking location using Bayesian optimization [15]. Goldman et al. first design a hybrid motion-force controller to follow cycloidal paths on a tissue surface to estimate the shape, then palpate at discretized surface locations with multiple grid resolutions to measure the impedance matrices across the surface [16]. Liu et al. map the stiffness by moving a mechanical roller across the tissue with a robot arm [17]. While these works are able to map the stiffness of heterogeneous tissues, force response is assumed to be linear.

III. Point Deformation Model Learning

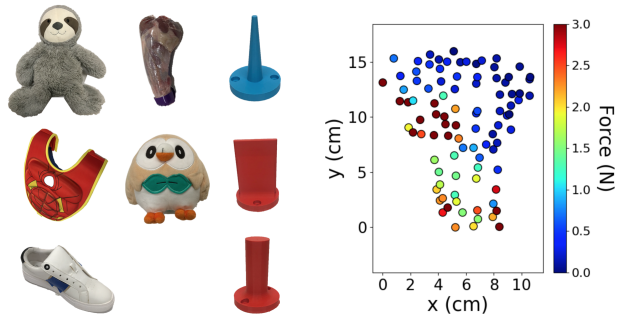
Fig. 1 shows the flow of our overall system. The first stage of our system learns a visuo-tactile model of the deformable object using data. We shall discuss the learning stage in the current section, leaving the discussion of simulation for Section IV. We assume that the object is elastic, quasistatic, and that interaction forces are dominated by normal deformation rather than friction and shear deformation. We hope to relax these assumptions in future work.

The point deformation model consists of an *equilibrium surface model*, and a *force response model*. The equilibrium surface model is a set S of points $x \in \mathbb{R}^9$, with each point $x = [p_x, p_y, p_z, r, g, b, n_x, n_y, n_z]$, representing position, color, and (outward) normal, respectively. The position and normal are represented in the world frame. The force response model is denoted as $f = y(x, d)$, where $f \in \mathbb{R}$ is force magnitude in the normal direction, x is a point in S, and $d \in \mathbb{R}$ is displacement. Letting $\mathbf{n} = [n_x, n_y, n_z] \in \mathbb{R}^3$ denote the normal in x and $\mathbf{p} = [p_x, p_y, p_z] \in \mathbb{R}^3$ the position, this map gives the force a point probe would feel in the direction of $\mathbf{n}$ when the point is moved to position $\mathbf{p} - d \cdot \mathbf{n}$.

A. Data Collection

Although our methods are applicable to naturalistic interactions, for the comparison purposes in this paper we employ a separate data acquisition stage similar to Frank et al [9] to build a ground truth dataset. The deformable object is laid on a flat surface, and it is assumed to return to its original position when pressed in the normal direction. If it is found to move while pushed, then a central point on the bottom of the object is affixed to the table using Velcro tape. In future work it may be possible to simply track the movement of the object during acquisition. We include 5 different testing objects in our experiments, Sloth, Vest, Lamb, Shoe, and Bird, shown in Fig. 2a. Sloth and Bird are stuffed animals, Vest and Shoe are clothing items, and Lamb is a piece of lamb leg consisting of muscle, fat, and bone. Each object has varying stiffness across the object surface. The regions of interest of the objects are the surfaces facing up when the objects are laid flat on a table, with the exception that we use only the face region of the sloth due to poor visual perception on the furry torso.

To build a 3D point cloud of the equilibrium surface, a wrist mounted RGB-D camera is used. Some objects need a few scans from different angles to generate a complete point



(a) Sloth, Vest, Lamb, Shoe, and Bird objects, along with rigid probes used in testing.

(b) Ground truth data on Lamb, with color showing force felt at d = 0.5 mm.

Fig. 2. The objects and probes used in the experiments. Data is gathered using the point probe, and the simulator generalizes force predictions to the line and cylinder probes.

cloud. Open3D [18] is applied to perform outlier removal and normal estimation.

To capture ground truth force response data, we randomly select a set of locations on the object to be poked with a point probe, shown in Fig. 1. During experiments, we collect data on a uniform grid, and randomly select among these locations to be training data, and the rest to be testing data. To complete one probing action at a point x, we align the point probe's axis with the point's surface normal. Starting at a user-specified distance from the surface, the probe pushes in the negative normal direction at a constant speed of 2mm/s until reaching a force threshold, which is set to 3-5 N in our experiments depending on the overall stiffness of the testing objects. The amount of displacement and force in the normal direction, measured by the wrist mounted F/T sensor, is recorded at 250 Hz. As a result, our force measurements are quite sparse in the spatial features, while being quite dense in force / displacement. One example of the collected data on the Lamb is shown in Fig. 2b.

If both the visual and tactile data have no error, then we would expect the force felt by the point probe tip away from the object surface to be zero and positive otherwise. However, this is often not the case. As shown in Fig. 4a, we often observe nonzero forces beginning anywhere from -5 to 5 mm of displacement. This is largely due to error in the calibration and bias in the camera's depth estimates, and further motivates the use of learning to correlate visual and tactile data.

For validation purposes, we collect second and third sets of poking data with a line probe and a cylindrical probe (capped at 3-14 N, depending on the objects), shown in Fig.2a. These are not used for training, but only for evaluation of how well our simulation model can generalize from point probe data. We probe once at each location using the cylinder probe, and three times with the line probe at each location with random orientation changes (about the local probe x-axis). We also record the torque about the EE frame center. The parameters of all three datasets are summarized in Table 1.

TABLE I

STATISTICS OF COLLECTED DATA ON THE TESTING OBJECTS

	Point Cloud	Point Probe	Line Probe	Cylinder Probe
Sloth	67,923	121	363	121
Vest	62,318	116	348	116
Lamb	18,059	94	276	94
Shoe	32,408	96	192	96
Bird	41,515	116	315	116

B. Force Response Model Learning

The next step in our pipeline learns to predict force responses at different locations on the object surface. Recent advancements in automated machine learning (AutoML) have greatly reduced the amount of manual effort needed to select a regression model and tune hyperparameters. In this paper we use the AutoML [19] package from scikit-learn [20] to perform the regression tasks.

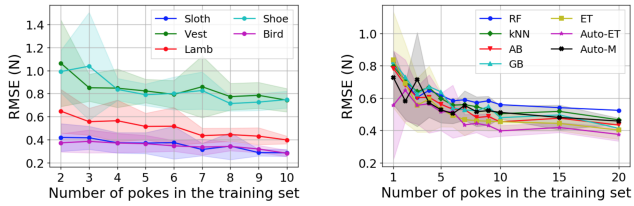
We compare performance of different types of regressors. The AutoML package searches among ensembles of different types of regressors and preprocessors and runs until a user-specified time limit is reached. We can also specify the type of regressors during AutoML training to reduce the number of hyperparameters that need to be optimized, thus improving training speed. Another option is to select an appropriate regressor directly, without relying on AutoML to select the hyperparamters. In particular, we test the following methods: (1) AutoML (Auto-M) trained for 360 s (2) AutoML with Extra Trees as the regressor type (Auto-ET) trained for 30 s; (3) Adaboost (AB), Gradient Boosting (GB), Random Forest (RF), and Extra Trees (ET), all trained using scikit-learn with default parameters. Fig. 3b show the results on the Lamb model. Although single regressors can train a model in less 1 s, Auto-ET generally achieves the best results in 30 s. If Auto-M were to be given more training time, it could potentially learn a better model, but doing so would require hours or even days of training. We settle on Auto-ET for the rest of the paper given its balance between speed and accuracy on our dataset. The resulting force predictions that evolve after 2, 5, 10 locations are used as training data for a 0.5 mm displacement on Lamb are shown in Fig. 4.

Results with Auto-ET on all 5 testing objects are shown in Fig. 3a. The Shoe and the Vest are the hardest to learn because they are not completely elastic and are made of composite materials, resulting in significantly varying stiffness across their surface.

Prediction speed per point varies with the query size, because of query overhead. With 1 point in each query, it takes ≈17 ms per point, while with 10,000 points per query, it takes ≈0.015 ms per point. Also, prediction speed is not directly related to the number of training points.

IV. SEMI-EMPIRICAL SIMULATOR

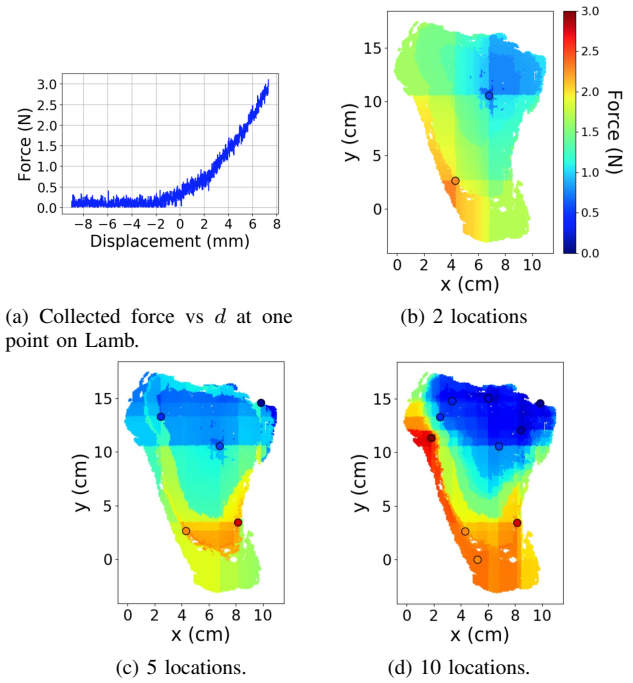
The second phase of our pipeline estimates reaction forces between the deformable object and a contacting rigid object. The rigid object is represented as a 3D surface mesh M defined in the global frame. The contact detector determines if it is in contact with the deformable object's equilibrium



(a) Testing error on each object using Auto-ET, with varying training set size.

(b) Comparing testing error between regressors, over all objects.

Fig. 3. Breakdown of learning curves for learning force response models. Regressors are trained over 10 sampled subsets of the ground truth dataset, and plots show RMSE mean and variance across samples.



(a) Collected force vs d at one point on Lamb.

(b) 2 locations

(c) 5 locations.

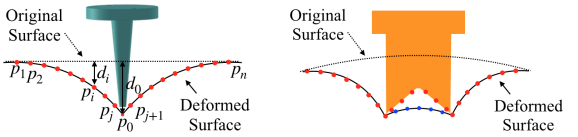
(d) 10 locations.

Fig. 4. Collected force-displacement curve at one point and force predictions on the Lamb object, learned with different numbers of poking locations. Poking locations are indicated as circles, colored by the true force. d is set to be 0.5 mm. Because of camera bias, there is already significant amount of force at this depth. [Best viewed in color]

surface model S, and if so, which points must be displaced to resolve the contact. The contact force solver then computes the 3D reaction force and torque and the deformed shape of the object.

A. Contact Force Solver

Classic Herztian contact theory predicts that displacement at a surface point on a deformable object will cause neighboring surface points to move in a manner described by a function of distance and material properties (Fig. 5a). Hence, we cannot simply sum the forces caused by each independent point displacement, but instead have to account for correlations in displacement. We assume that normal forces and displacements are the dominant effect, and experiments on our object set suggest this assumption is reasonable for normally-displaced objects.



(a) A point indentation moves nearby points on the deformable object.

(b) Displaced points calculated by the collision detector (red) vs actual displacements (blue).

Fig. 5. Definitions used in the surface deformation model.

Let d_i denote the *nominal displacement* of a surface point at a contact patch, we assume that it consists of both displacement caused by neighboring point displacement and that actually contributes to the total force, which we call *actual displacement* and denote as $\tilde{d}_i$. We solve for non-negative actual displacements that are no greater than the nominal displacements. A smaller nominal displacement would indicate penetration, while a negative actual displacement would cause the deformable to apply a negative (pulling) force on the rigid object at that point (Fig. 5b).

1) Nominal Displacement Calculation: To calculate nominal displacements of the deformable object's surface points we use collision detection and estimate penetration depths.

To do so, we first discretize the surface of the rigid object by placing equally spaced points on edges and 2D grids on facets based on a user-selected discretization. Via experimentation, we found that simulation results are relatively insensitive to the selection of this discretization, and we use a 3 mm resolution for all experiments.

Next, we find the pairs of rigid surface points and the deformable surface points that will be touching after deformation and the nominal displacement of the deformable surface points. Point correspondence is calculated by projecting all the points to a 2D plane and performing nearest-neighbor searches. We let $\mathbf{a}$ be the approach direction of the rigid object and the normal of the projection plane, shown in Fig. 6. To better approximate the shape of the underlying continuous surface, we average the nominal displacements of the the 3 nearest deformable surface points for each rigid surface point. Each nominal displacement is calculated according to

$$d_i = -(\mathbf{p}_{rigid_j} - \mathbf{p}_{deformable_i})^T \cdot \mathbf{n}_{deformable_i}, \qquad (1)$$

where $\mathbf{p}_{deformable_i}, \mathbf{n}_{deformable_i} \in \mathbb{R}^3$ are the equilibrium surface position of point i and its associated normal. We average the equilibrium positions and nominal displacements of the 3 points to obtain the *estimated equilibrium surface point* and displacement, which we refer to as $\mathbf{q}$ and d. By keeping only the points with $d > 0$, we have N pairs of rigid surface points and its corresponding estimated equilibrium surface points along with nominal displacements.

2) Force Solving: The force solver uses the nominal displacements calculated above combined with an indentation basis function to compute the actual displacements.

The shape function is defined as follows. For each pair of points on the deformable object, we define a shape function $g(r) : \mathbb{R} \to \mathbb{R}$ to describe the correlation of a point

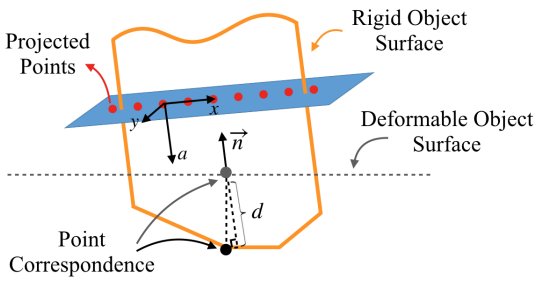


Fig. 6. An illustration of how the nominal displacements are calculated for a polyhedral rigid object.

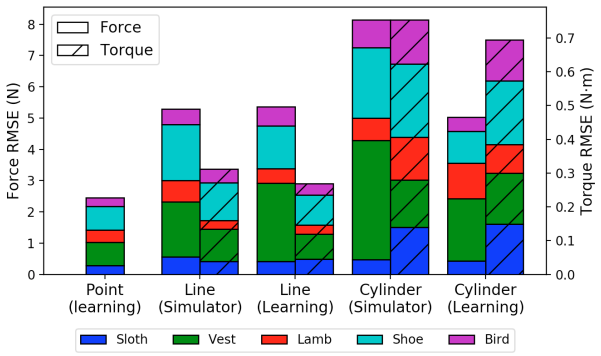


Fig. 7. Predictions errors of our semi-empirical simulator on novel rigid objects and that of pure data-driven models. Errors on different objects are shown in different colors. We also include the errors of the point models used in predictions as a reference.

indentation on its neighborhood. Here r is the Euclidean distance between 2 points on the equilibrium surface, and we use the inverse function:

$$g(r) = c/(c+r) \tag{2}$$

$f(r)$ describes an inverse relationship between points' relative displacement and equilibrium surface distance, which matches the Hertzian theory. The constant $c > 0$ is picked by observation. However, for future work, we could determine this constant using a camera tracking the shape of deformation during poking. While this constant may actually vary across the object surface, we keep it constant across the object in this paper.

By calculating the relative distance $r_{i,j}$ between all pairs of deformable points $\mathbf{q_i}$ and $\mathbf{q_j}$ in collision with the rigid object, we obtain a linear complementary problem (LCP) relating nominal and actual displacements:

$$\mathbf{w} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \end{bmatrix} - \begin{bmatrix} g(r_{1,1}) & g(r_{1,2}) & \dots & g(r_{1,n}) \\ g(r_{2,1}) & g(r_{2,2}) & \dots & g(r_{2,n}) \\ \vdots & \vdots & \ddots & \vdots \\ g(r_{n,1}) & g(r_{n,2}) & \dots & g(r_{n,n}) \end{bmatrix} \begin{bmatrix} \tilde{d}_1 \\ \tilde{d}_2 \\ \vdots \\ \tilde{d}_n \end{bmatrix} \tag{3}$$

$$0 \leq \mathbf{w} \perp [\tilde{d}_1, \tilde{d}_2, \dots, \tilde{d}_n]^T \geq 0,$$

where the subscripts denote the indices of the deformable surface points in collision. When there are not any duplicated points, i.e. $g(r_{i,j}) \neq 1$ except for $i = j$, this symmetric matrix is invertible. We solve the LCP by iteratively assuming equalities, solving the system of linear equations by matrix inversion, and removing the points with negative actual displacements.

From the actual displacements we sum the forces predicted by the point deformation model to obtain the contact wrench. Given the N_{final} points with positive actual displacements, the total response wrench $(F_{total}, \tau_{total})$ on the object are calculated as follows:

$$\begin{aligned} f_i &= y([q_i, d_i]) - y([q_i, d_i - \tilde{d}_i]) \\ F_{total} &= \Sigma_{i=1}^{N_{final}} f_i \cdot \mathbf{n}_i \\ \tau_{total} &= \Sigma_{i=1}^{N_{final}} (\tilde{\mathbf{r}}_j) \times (f_i \cdot \mathbf{n}_i), \end{aligned} \tag{4}$$

where $\tilde{r}_j$ is the vector from the torque center to the corresponding rigid body point $\mathbf{p}_{rigid_j}$.

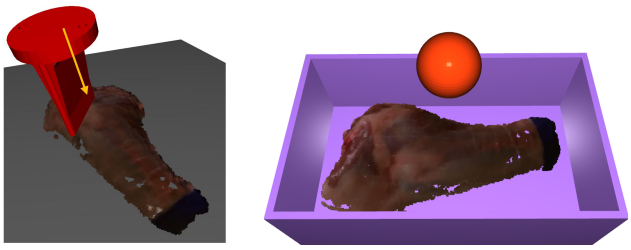
To predict the displacement of the entire object, not only the points in contact, we sum up contributions from the shape function over every positive actual displacement.

B. Experiments

Throughout this section, we use the point deformation models trained with Auto-ET and data from 10 poking locations. All implementation is performed in Python code with the Numpy package for linear algebra, and all timing is conducted on a desktop PC with an Intel i7 3.8GHz processor.

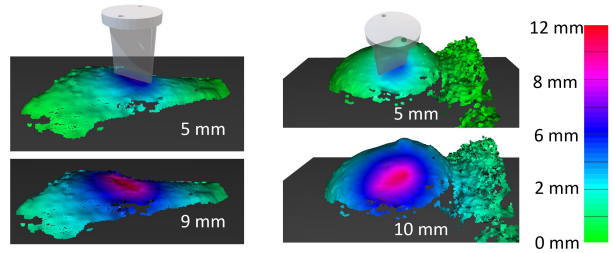
1) Simulation accuracy for Line and Cylinder contact: Fig. 7 shows evaluation of our simulator with line and cylinder probe contact in predicting 3D force and torque vectors. The simulator prediction errors are compared to pure learning using the correct geometry, trained directly on 10 pokes from the ground truth line and cylinder probing data, where Auto-ET is also used as the regressor. We notice that the actual displacements calculated could exceed the range of displacements in the point training data. We simply extrapolate the training data following the estimated force-displacement slope to incorporate all possible queries from the simulator. Overall, our simulator performs slightly worse than pure learning, better on Lamb and Bird, and worse on Vest and Shoe. We believe this is mainly due to the fact that Vest and Shoe are made of composite materials with a mixture of elastic and inelastic components. As a result, shape functions approximate the deformation of these objects sub-optimally and lead to errors in the simulator. The key advantage of our method is that we do not need to re-acquire a new training set, and we can also predict force response for different rolls and pitches of the rigid objects, which are not included as features in the learning-only training set.

2) Simulation Visualization: We illustrate our simulator by visualizing the simulation of a line or cylinder probe poking straight at different objects, resembling the data collection process in Section III.A. An example is shown in Fig. 8a. The simulation frames are displayed in Fig. 9. As the probes penetrate deeper, the calculated actual displacements also increase, thus deforming the entire surface more

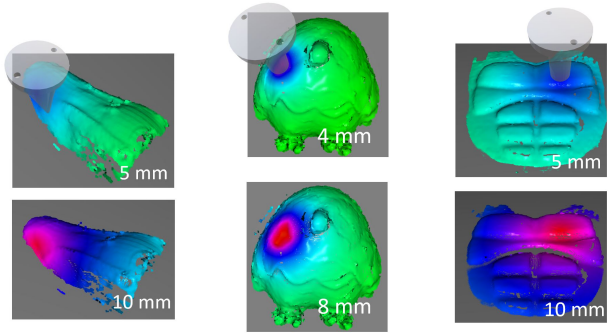


(a) Contact with a line probe. (b) Robot packing setup.

Fig. 8. Simulation experiments, Lamb object.



(a) Lamb probed by a line probe. (b) Sloth probed by a line probe.

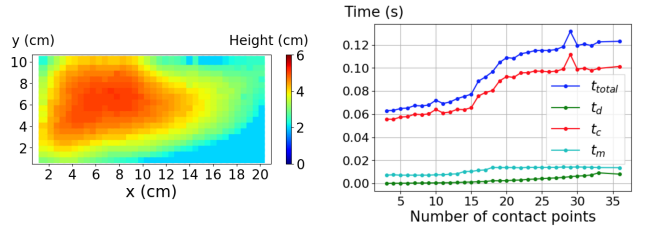


(c) Shoe probed by a line probe. (d) Bird probed by a cylinder probe. (e) Vest probed by a cylinder probe.

Fig. 9. Simulation of the line and cylinder probes poking straightly at all 5 experiment objects, with the numbers indicating the amount of probe displacements. The objects are colored with the displacements of the surface points, caused by the probes. [Best viewed in color]

significantly, calculated by the shape function. We can see grooves on the Lamb, the Bird, and the Sloth that fit the probe geometry at deep penetration. The Shoe and the Vest are stiffer objects, with c in the shape function much greater than the other 3 objects. As a result, the grooves are harder to observe.

3) Application to a Packing Problem: We consider a simple packing problem as an illustration of our approach, shown in Fig. 8b. Given a rigid box with a deformable object (the Lamb) already packed, we would like to know if the robot can pack another rigid item (a sphere, with diameter = 40 mm and discretized into 150 surface points) into the box. In particular, we would like to know at what positions in the box the sphere can be placed such that its interaction force with the deformable object does not exceed 2 N. We perform a grid search over the x-y-z position of the rigid sphere centroid inside the box, where z starts with the maximum allowed height in the box and finishes the search



(a) The minimum height at which the rigid sphere can be packed under force threshold 2 N. (b) Computation time vs number of contacts, showing total time (t_{total}), displacement calculation (t_d), collision detection (t_c), and point model queries (t_m).

Fig. 10. Results from the planning problem.

in the negative z direction once the interaction force limit is exceeded, shown in Fig. 8b. A plot of minimum possible height of the sphere on the x-y grid is shown in Fig. 10a.

On average in this experiment, a contact reaction force query takes 0.086 s. Each query consists of 0.0738 s of collision detection, 0.010 s of querying the point model, and 0.0017 s of LCP solving by our simulator. The average computation time as a function of the number of detected contact points m is plotted in Fig. 10b. The time for LCP increases cubically with m, because it mainly involves inverting a dense matrix. The time for querying the point model stays somewhat constant. There is a lot of room for improvement in speed, including implementation in a compiled language, better collision detection algorithms [21], and parallelization.

V. Conclusion

The paper proposed a 2-stage data-driven framework to learn and simulate the force responses of heterogeneous elastic objects. In the first stage, a point deformation model of a testing object is learned via a robot arm poking at the object a few times. In the second stage, a semi-empirical simulator predicts the contact wrench between the deformable object and a novel rigid object by integrating analytic calculation and the learned point model to obtain the expected restoring forces at a series of contact points and summing them.

For future work, we would like to relax several assumptions made in this paper. First, learning and generalizing shear and friction forces with small numbers of touches is an open problem. Perhaps the deformation and force response of a novel deformable object can be bootstrapped by transfer learning from an object that has already been acquired. We would also like learn the deformation and force response of an object while it is undergoing gross movements, separating the rigid body and elastic components of the movement. This could also be extended to handle plastic deformation. Finally, we would like to enhance our simulator to handle combined rigid and elastic deformation, as well as contact between deformable objects.

Acknowledgment

We thank Joao M. C. Marques for proofreading the paper.

REFERENCES

[1] J. Zhou, R. Paolini, J. A. Bagnell, and M. T. Mason, "A Convex Polynomial Force-Motion Model for Planar Sliding : Identification and Application," *IEEE Int. Conf. Rob. Aut.*, no. 3, pp. 372–377, 2016.

[2] A. Kloss, S. Schaal, and J. Bohg, "Combining learned and analytical models for predicting action effects," in *arXiv preprint arXiv:1710.04102*, 2017.

[3] J. Wu, J.-x. Wang, and H. Xiao, "Physics-Informed Machine Learning for Predictive Turbulence Modeling: A Priori Assessment of Prediction Confidence," in *arXiv preprint arXiv:1607.04563*, 2016.

[4] M. Raissi and G. E. Karniadakis, "Hidden physics models : Machine learning of nonlinear partial differential equations," *Journal of Computational Physics*, vol. 357, pp. 125–141, 2018. [Online]. Available: https://doi.org/10.1016/j.jcp.2017.11.039

[5] G. Carleo, I. Cirac, L. Cranmer, KyleDaudet, D. Bourse, M. Schuld, L. Vogt-maranto, and L. Zdeborová, "Machine learning and the physical sciences," 2019.

[6] Y. Zhu, L. Abdulmajeid, and K. Hauser, "A Data-driven Approach for Fast Simulation of Robot Locomotion on Granular Media," in *IEEE Int. Conf. Rob. Aut.*, 2019.

[7] A. Oishi and G. Yagawa, "Computational Mechanics Enhanced by Deep Learning," *Comput. Methods Appl. Mech. Engrg.*, vol. 327, pp. 327–351, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.cma.2017.08.040

[8] A. Nealen, M. Matthias, R. Keiser, E. Boxerman, and M. Carlson, "Physically Based Deformable Models in Computer Graphics," *Computer Graphics Forum*, vol. 25, no. 4, pp. 809–836, 2006.

[9] B. Frank, R. Schemedding, C. Stachniss, M. Teschner, and W. Burgard, "Learning the elasticity parameters of deformable objects with a manipulation robot," in *IEEE/RSJ Int. Conf. Intel. Rob. Sys.*, no. October, 2010.

[10] P. Boonvisut and M. C. Cavusoglu, "Estimation of Soft Tissue Mechanical Parameters From Robotic Manipulation Data," *Transactions on Mechatronics*, vol. 18, no. 5, pp. 1602–1611, 2013.

[11] B. Bickel, M. Bächer, M. A. Otaduy, W. Matusik, H. Pfister, and M. Gross, "Capture and modeling of non-linear heterogeneous soft tissue," in *SIGGRAPH*, New York, New York, USA, 2009, p. 1.

[12] O. Deussen, L. Kobbelt, and T. Peter, "Using Simulated Annealing to Obtain Good Nodal Approximations of Deformable Bodies," in *Eurographics Workshop*, no. September, 1995.

[13] S. Burion, F. Conti, A. Petrovskaya, C. Baur, and O. Khatib, "Identifying Physical Properties of Deformable Objects by Using Particle Filters," in *IEEE Int. Conf. Rob. Aut.*, 2008, pp. 1112–1117.

[14] T. Yamamoto, B. Vagvolgyi, K. Balaji, L. L. Whitcomb, and A. M. Okamura, "Tissue Property Estimation and Graphical Display for Teleoperated Robot-Assisted Surgery," *IEEE Int. Conf. Rob. Aut.*, pp. 4239–4245, 2009.

[15] H. Salman, E. Ayvali, R. A. Srivatsan, Y. Ma, N. Zevallos, R. Yasin, L. Wang, N. Siman, and H. Choset, "Trajectory-Optimized Sensing for Active Search of Tissue Abnormalities in Robotic Surgery," in *IEEE Int. Conf. Rob. Aut.*, 2018.

[16] R. E. Goldman, A. Bajo, and N. Simaan, "Algorithms for autonomous exploration and estimation in compliant environments," *Robotica*, vol. 31, no. 1, 2013.

[17] H. Liu, D. P. Noonan, B. J. Challacombe, P. Dasgupta, L. D. Seneviratne, and K. Althoefer, "Rolling Mechanical Imaging for Tissue Abnormality Localization During Minimally Invasive Surgery," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 2, pp. 404–414, 2010.

[18] Q.-y. Zhou, J. Park, and V. Koltun, "Open3D : A Modern Library for 3D Data Processing," *arXiv:1801.0984*, 2018.

[19] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and Robust Automated Machine Learning," in *Advances in Neural Information Processing Systems*, 2015, pp. 2962—-2970.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M.Perrot, and E. Duchesnay, "Scikit-learn : Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[21] R. Weller, "A Brief Overview of Collision Detection," in *New Geometric Data Structures for Collision Detection and Haptics*, 2013, pp. 9–46.