2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Estimating Tactile Models of Heterogeneous Deformable Objects in Real Time

Shaoxiong Yao[1] and Kris Hauser[1]

***Abstract*— This paper introduces a method for learning the force response of heterogeneous, deformable objects directly from robot sensor data without prior knowledge. The method estimates an object's force response given robot force or torque measurements using a novel volumetric stiffness field representation and point-based contact simulator. The stiffness of each point colliding with the robot is estimated independently and is updated upon each observed measurement using a projected diagonal Kalman filter. Experiments show that this method can update a stiffness field over 10^5 points at 23 Hz or higher, and is more accurate than learning-based methods in predicting torque response while touching artificial plants. The method can also be augmented with visual information to help extrapolate stiffness fields to distant parts of the touched object using only a small number of touches.**

I. Introduction

Robots deployed in domestic [1] and agricultural [2] environments have to frequently interact with deformable objects like clothes, human anatomy, and plants. Tactile models that predict force responses given deformation are useful for applications like surgical palpation [3] and food preparation [4]. Although it is known how to plan paths and manipulation trajectories given a known deformable object model [5], [6], it is far less clear how to estimate a model of the deformable object in the first place. Continuum mechanics models such as finite element methods (FEM) are able to simulate realistic deformation behavior, but calibrating such models to match real data is challenging because they require solving a complex inverse problem, and also require substantial amounts of data to accurately model heterogeneity.

In this paper, we present a novel method for tactile model estimation of heterogeneous deformable objects using a simple but dense probabilistic representation that can be updated in real-time. The *volumetric stiffness field* (VSF) model represents force response as a robot passes through an elastic, dense particle cloud defined over a voxel grid. Each particle is attached to its reference cell using a Hookean spring with unknown stiffness, and the probability distribution over spring stiffness is updated as the robot observes force / torque measurements. As the robot arm sweeps across voxels, a fast point-based simulator estimates how the points are displaced by the arm. Compared to, say, FEM models, our method sacrifices fidelity to the underlying continuum mechanics of the object, but it allows us to estimate densely sampled stiffness fields for heterogeneous objects in real time. A decoupled Kalman filter algorithm updates the stiffness distribution of each spring independently as a function of robot force or torque measurements. The decoupled assumption makes each update fast, and we introduce a memory replay technique that empirically improves accuracy to be comparable to full dense Kalman filter updates while still retaining real-time performance.

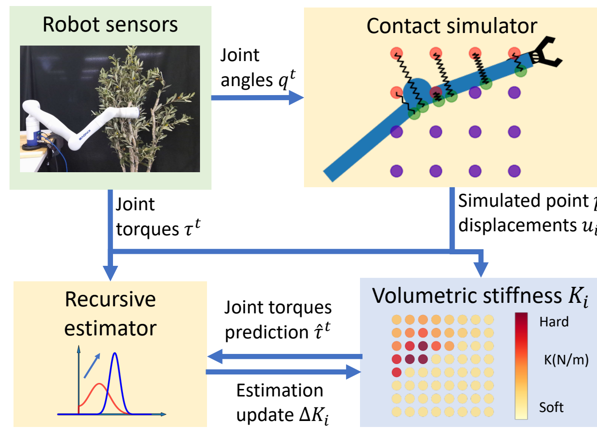


Fig. 1. Our volumetric stiffness field (VSF) estimation system consists of: (1) a contact simulator that simulates the displacement of points pushed by the robot arm, (2) a VSF containing estimated stiffness distribution, (3) a recursive state estimator that updates the VSF belief.

We test our method in a set of real-world scenarios where a robot arm touches artificial plants. Our method updates the belief of a volumetric stiffness field represented by 10^5 points at 23 Hz or higher. The estimated stiffness field is used to predict joint torques when similar locations are touched multiple times, and our method is approximately twice as accurate as learning-based approaches. We also implement a vision-based extrapolation algorithm that predicts the stiffness of untouched regions from touched regions data. The extrapolated results outperform baseline methods in the accuracy of robot joint torque predictions and qualitatively can identify plant parts that have different stiffness, such as leaves, small branches, and large branches.

II. Related Work

Deformable object modeling. Various deformable object models have been used in robotics applications [7]. Material parameter identification methods are available to match observed deformation or force/torque data to FEM [6] or mass-spring models [8]. However, they are computationally expensive and assume a given mesh topology. Recent

[1]: S. Yao and K. Hauser are with the Department of Computer Science, University of Illinois at Urbana-Champaign, IL, USA. {syao16, kkhauser}@illinois.edu. This project was supported by USDA/NIFA Grant #2020-67021-32799.

learning-based deformable object models, like graph neural networks [9] or implicit neural functions [10], can relax these assumptions, but usually require a large amount of training data and compute resources. Tactile models based on machine learning methods predict the force response at varying touch locations and avoid strong assumptions about the mesh structure [11]. However, these prior works assume a roughly planar object, discrete touches, and coherence in the object's force response across the object's surface. Recent work took a similar point-based discretization with Neo-Hookean stiffness model [12], capturing complex interactions between particles in the object but cannot be estimated in real time.

Our proposed Hookean spring representation is similar to existing models of contact force response. The method of dimensionality reduction also uses a set of springs without interconnection to simulate contact effects [13]. Similar representations have been used as potential field methods for real-time haptic rendering [14]. However, in prior work such models have been used for simulation rather than estimation.

Tactile model estimation. Existing tactile estimation methods deal with mostly planar objects and point touches. Frequently, the touch sensors used restrict contact to the robot's end-effector [4], [6], [15]. In an open-world setup, where the robot works in an unstructured environment, contact may not happen at the exact location. Full-arm tactile skins can deal with unstructured contacts but are still far from common deployment [16].

Joint torques sensors equipped on most industrial robot arms are a promising alternative [17]. However, using joint torque sensors to perceive objects' stiffness has several unique challenges. Pang and Tedrake [18] pointed out that there is ambiguity to localize contacts from joint torque measurements. Existing works use probabilistic filters to estimate contact locations [19], [20], but do not estimate the objects' model. Probabilistic filters were used to map the shape of rigid objects from contact [21], [22], but not the material properties of deformable objects. In this work, we take a probabilistic filtering approach to estimate deformable objects' stiffness in real time.

III. Method

We present a tactile model estimation system that recursively updates a probabilistic volumetric stiffness model of elastic deformable objects in a computationally efficient manner. Our system takes joint angles and joint torques measurements as input and uses the difference between expected and actual torques measurements to correct the belief of stiffness distribution over a particle field. Fig. 2 illustrates the stiffness estimation process in an 1D toy example. The estimated volumetric stiffness field can be used to predict torque measurements given novel contact configurations or locations. This method is certainly imperfect at representing deformable solids in that it fails to capture the coupled relationship between displacements at one point and another, and also assumes linear elasticity. But since the representation is dense, redundant, and can be updated online to match incoming observations with high accuracy, it can approximate many force response phenomena well enough to predict the results of hypothetical movements and to identify stiff vs soft components of heterogeneous objects.

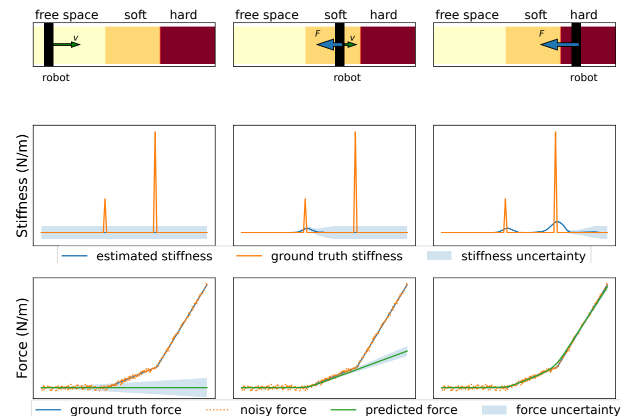


Fig. 2. Stiffness estimation process in 1D environment. Top: a robot (black bar) pushes through three regions with different stiffness. Middle: the VSF stiffness belief updates a probability distribution over each point in the domain from the robot's noisy force measurements. Bottom: predicted force response. Shaded regions indicate one standard deviation of the VSF belief / prediction.

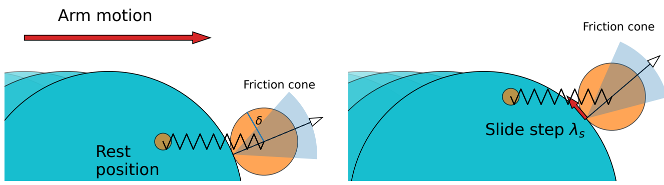
Our method has two variants: a *blind* touch-only version that only takes robot joint angles and joint torque measurements as input, and a *touch+vision* version that uses color information of the observed object point cloud to extrapolate stiffness from touched regions to untouched regions.

A. Volumetric Stiffness Field

The VSF models the object as a set of points $\{p_i\}_{i=1..N}$ from a region V expressed in a world frame. When the robot arm sweeps through V, a subset of points will be pushed and displaced by the arm. We denote the indices of contact points at time t as $S^t \subset \{1, .., N\}$ and the displacement of point p_i as $u_i^t \in \mathbb{R}^3$. The model makes three key assumptions about the displacements and force responses:

(i) Hookean force response. Each point has a virtual spring with time-invariant stiffness $K_i \in \mathbb{R}_+$ connecting it to its rest position p_i. The force response F_i^t at time t is:
$$F_i^t = -K_i \cdot u_i^t \tag{1}$$
(ii) Quasi-static, stick-slip motion. We assume the robot arm moves slowly enough such that points are in quasi-static equilibrium with the arm. Points in contact can either stick to the arm or slip along the surface of the arm, as shown in Fig. 3;
(iii) Independent motion. We assume there is no interaction between points, and the belief over of the stiffness coefficient of each point is independent ($P(K_i, K_j) = P(K_i)P(K_j)$ for $i \neq j$);

This last assumption ignores the complex interaction between points compared to mass-spring and FEM models, but as we shall see it allows us to perform an estimation of the stiffness field extremely quickly and is a reasonable approximation for some objects. Also note that we make no distinction between empty and occupied space; with a properly estimated VSF, a point p_i in empty space should have stiffness $K_i = 0$.



(a) Stick motion. (b) Slip motion

Fig. 3. Contact simulator illustration. A distance query determines whether a point in VSF is in contact with the robot arm with threshold δ. For contacting points, we simulate stick and slip motion on the robot's surface using the point's displacement from the rest position as the force direction.

Our notation groups the stiffness coefficients into the vector $K = [K_1, \cdots, K_N]^\top$ and displacements at time t into the matrix $u^t = [u_1^t, \cdots, u_N^t]^\top$.

B. Estimating VSF in real-time

A key innovation that makes our system work in real time is to *decouple* the stiffness estimator from the contact simulator. At each step, the contact simulator will update the contact set S^t and displacements u^t given the new arm configuration $q^t \in \mathbb{R}^d$, independent of the VSF. Here, d indicates the number of DOFs of the robot arm. Then, using u^t and the belief over K, the VSF will predict force response F_i^t for $i \in S^t$. The joint torques $\tau^t \in \mathbb{R}^d$ will be a low-dimensional linear observation of K. Finally, the recursive estimator will update the belief over K.

a) Contact simulator: The contact simulator receives the robot arm's current configuration q^t at every time step. It also maintains an internal state of contact points S^{t-1} and displacements u^{t-1}. The simulator simulates the stick-slip behavior of particles but updates the displacement of each point in parallel following the independence assumption (iii). The simulator simulates three effects: (1) contact detection by distance threshold δ, and the distance is determined by a pre-computed signed distance field of the robot geometry using the Klampt library [23]; (2) stick motion with friction coefficient ν_f, we compute surface normal unit vector $\hat{n}_i^t$ using the gradient of signed distance field, and the point sticks to the surface when ${u_i^t}^\top \hat{n}_i \geq \|u_i^t\|_2/\sqrt{1+\nu_f^2}$; (3) slip motion in the tangential direction with fixed step size λ_s in each iteration, when slip happens, the slip direction is determined by $r_s = u_i^t - \hat{n}_i {u_i^t}^\top \hat{n}_i$, $u_i^{t+1} = u_i^t - \lambda_s r_s/\|r_s\|_2$. A point is detached from the surface when ${u_i^t}^\top \hat{n}_i < 0$ and the point is placed to the rest position and $u_i^{t+1} = 0$ after detached. We choose $\delta = 2\,\mathrm{mm}$, $\nu_f = 0.15$ and $\lambda_s = 0.1\,\mathrm{mm}$ and the estimation performance is insensitive to the choice of these parameters. The running time per point is $O(1)$ and hence the time complexity of each timestep is $O(|S^t|)$. Fig. 3 illustrates the three effects simulated by the contact simulator.

b) Observation model: We assume the robot arm moves quasi-statically (ii), so the external torques τ are obtained by subtracting out the gravitational torques. The relation between the external joint torques and force responses from VSF is given by:

$$\tau^t = \sum_{i \in S^t} J(q^t, p_i + u_i^t)^\top F_i^t + \epsilon_\tau, \tag{2}$$

where $J(q^t, p_i + u_i^t)$ is the contact Jacobian matrix that maps joint velocity to the translational velocity at $p_i + u_i^t$, and ϵ_τ is a noise term. Combining Eq. 1 and Eq. 2, we have

$$\tau^t = -\sum_{i \in S^t} J(q^t, p_i + u_i^t)^\top u_i^t K_i + \epsilon_\tau. \tag{3}$$

The observation error at each step is additive independent Gaussian noise with magnitude σ_τ, i.e., $\epsilon_\tau \sim \mathcal{N}(0, \sigma_\tau^2 I)$.

c) Recursive estimator: The update step updates the belief over the stiffness coefficients K given the joint torque observations. Here we show that the observation is *linear* in stiffness once the displacements are known, making the update equivalent to a projected Kalman filter step.

Denote the joint torques contributed by each point per unit stiffness as $w_i^t \in \mathbb{R}^d$,

$$w_i^t = \begin{cases} -J(q^t, p_i + u_i^t)^\top u_i^t, & i \in S^t, \\ \mathbf{0}_d, & \text{otherwise.} \end{cases}$$

Here $\mathbf{0}_d \in \mathbb{R}^d$ is the zero vector. Now we define the observation matrix $W^t = [w_1^t, \cdots, w_N^t]$ and write the observation equation 3 in a matrix form

$$\tau^t = W^t K + \epsilon_\tau. \tag{4}$$

This equation means joint torques are *linear* observations of K and a Kalman filter is suitable to update the belief. However, because the stiffness should be non-negative, we follow a state projection approach discussed in [24].

A standard projected Kalman filter maintains a multivariate Gaussian belief over $K \sim \mathcal{N}(\mu^t, \Sigma^t)$ with mean $\mu = [\mu_1, \cdots, \mu_N]^\top$. It performs the following update rule in the information filter form:

$$\begin{aligned} (\Sigma^{t+1})^{-1} &= (\Sigma^t)^{-1} + \frac{1}{\sigma_\tau^2} {W^{t+1}}^\top W^{t+1} \\ \tilde{\mu}^{t+1} &= \mu^t + \frac{1}{\sigma_\tau^2} \Sigma^{t+1} {W^{t+1}}^\top (\tau^{t+1} - W^{t+1}\mu^t) \\ \mu^{t+1} &= \max(\tilde{\mu}^{t+1}, 0) \end{aligned} \tag{5}$$

Here the last equation executes the state projection as a special case of equation (47) in [24]. This naïve Kalman filter requires maintaining and inverting a matrix of size $|S^t| \times |S^t|$. This operation fails to run in real time when the number of contact points increases into the thousands (Fig. 4).

Our approach exploits the independence assumption (iii) to significantly reduce the computation. The belief of each K_i is represented by an independent Gaussian distribution with no correlation between different points:

$$K_i \sim \mathcal{N}(\mu_i, \sigma_i^2) \tag{6}$$

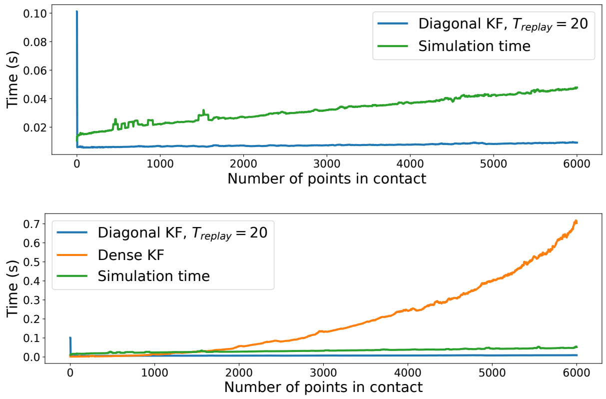


Fig. 4. Update time per iteration comparison between dense Kalman filter and proposed diagonal Kalman filter. Both algorithms are implemented using GPU-accelerated linear algebra operation from PyTorch.

Algorithm 1: Memory replay update

Input: $\{\mu_i^t, \sigma_i^t\}$, $\mathcal{B}^t$, (W^{t+1}, τ^{t+1});
$\mathcal{B}^{t+1} \leftarrow \mathcal{B}^t \cup \{(W^{t+1}, \tau^{t+1})\}$;
$\{\overline{\mu}_i^0, \overline{\sigma}_i^0\} \leftarrow DiagKF(\{\mu_i^t, \sigma_i^t\}, W^{t+1}, \tau^{t+1})$;
for k *in* $1..T_{replay}$ **do**
    $(W^j, \tau^j) \leftarrow \text{Sample}(\mathcal{B}^{t+1})$, $1 \leq j \leq t+1$;
    $\{\overline{\mu}_i^k, \overline{\sigma}_i^k\} \leftarrow DiagKF(\{\overline{\mu}_i^{k-1}, \overline{\sigma}_i^{k-1}\}, W^j, \tau^j)$;
return $\{\overline{\mu}_i^{T_{\text{replay}}}, \overline{\sigma}_i^{T_{\text{replay}}}\}, \mathcal{B}^{t+1}$;

so we only need to maintain a diagonal covariance matrix. The update rule will be a parallel operation over μ_i^t and σ_i^t,

$$
\begin{aligned}
(\sigma_i^{t+1})^{-2} &= (\sigma_i^t)^{-2} + \frac{1}{\sigma_\tau^2} w_i^{t+1\top} w_i^{t+1} \\
\tilde{\mu}_i^{t+1} &= \mu_i^t + \frac{1}{\sigma_\tau^2} (\sigma_i^{t+1})^2 w_i^{t+1\top} (\tau^{t+1} - W^{t+1}\mu^t), \\
\mu_i^{t+1} &= \max\left(\tilde{\mu}_i^{t+1}, 0\right)
\end{aligned}
\tag{7}
$$

Such an independent belief representation reduces the belief update time complexity from $O(|S^t|^3)$ to $O(|S^t|)$. The parallel update in Eq. 7 was accelerated by GPU using standard matrix operations from PyTorch [25]. The current time bottleneck is in simulator stepping implemented on CPU as shown in Fig. 4. However, the optimal estimation property of dense Kalman filter fails to hold. This decoupling idea in Kalman filters was initially proposed in [26]. Recent work proved the convergence of decoupled extended-Kalman filter under complex assumptions [27]. In our case, by using a memory replay technique introduced in the next section, we empirically found the diagonal Kalman filter can achieve similar accuracy compared to dense Kalman filter.

C. Memory replay

In practice, we found such a purely recursive update has a slow convergence speed. We propose a memory replay strategy to enhance the estimation results. We maintain a buffer of history observations $\mathcal{B}^t = \{(W^j, \tau^j)\}_{j=1}^t$. In each iteration, after the recursive update, a fixed number of history samples will be uniformly drawn from $\mathcal{B}^t$ and diagonal Kalman filter update the same as Eq. 7 is executed. The detailed procedure is shown in Alg. 1. The memory buffer will be reinitialized for each touch sequence, so its memory consumption will remain bounded.

TABLE I

ESTIMATION PERFORMANCE WITH / WITHOUT MEMORY REPLAY.

	$T_{\text{replay}} = 0$	$T_{\text{replay}} = 10$	$T_{\text{replay}} = 20$	Dense KF
Avg. $\|\hat{\tau} - \tau\|_2$ (Nm)	1.8 ± 0.4	1.3 ± 0.2	1.1 ± 0.1	1.1 ± 0.1
Frequency (Hz)	30.0 ± 3.0	28.3 ± 2.7	26.3 ± 2.2	6.3 ± 2.6

Table I shows that this memory replay method can achieve similar accuracy compared to the dense Kalman filter using a small T_{replay}. We compare the average torque prediction error $\sum_{j=1}^T \|\hat{\tau}^j(\mu^T) - \tau^j\|_2/T$ at the end of each sequence, where $\hat{\tau}^j(\mu^t) = W^{j\top}\mu^t$ denotes the mean predicted torque at time j by μ^t. Using $T_{\text{replay}} = 20$ can reduce estimation error by 40% while maintaining high update frequency.

D. Hybrid vision and touch stiffness estimation

Certainly, touching the object all over to obtain a complete stiffness map would be time-consuming. We examine a relatively simple approach that extrapolates stiffness from touched regions to untouched regions using correlations in visual features. Our visual extrapolation algorithm is composed of 3 steps: 1) We take a single depth image of the artificial plant before contact happens. 2) For touched points in the point cloud we look up the stiffness values of nearby touched voxels, creating a dataset that maps RGB and position (a 6-D vector) to estimated stiffness. 3) For all visible points, we use k-NN regression to predict the stiffness value from their color and position. We use $k = 200$ neighbors and an RBF kernel in our experiments. For invisible points, we extrapolate using position features only.

IV. Experiments

We perform a set of real-world experiments using a Kinova Gen3 robot arm touching four artificial plants. The experiments demonstrate the benefits of our approach from two aspects: (1) real time estimation speed and (2) generalization in joint torque prediction.

A. VSF initialization

The set of points $\{p_i\}_{i=1..N}$ in VSF is initialized from a single depth image for each object. First, we have a bounding box in the world frame that fully contains the object. The box dimensions are 0.6m×0.4m×0.4m for foxtail and 0.85m×0.75m×1.2m box for dracaena, orange tree, and cherry tree. We discretize this bounding box into a 100^3 uniform grid. Then, we perform ray tracing using calibrated camera parameters to find occluded points on the uniform grid from the depth image. The $\{p_i\}_{i=1..N}$ of VSF consists of both invisible volumetric points and surface point clouds within the bounding box.

We use a spatially uniform zero mean and equal variance Gaussian prior distribution $\mathcal{N}(\mu_i^0, (\sigma_i^0)^2)$ assuming no knowledge about object stiffness distribution, where $\mu_i^0 = 0\,\text{N/m}$, $\sigma_i^0 = 0.1\,\text{N/m}$. This prior allows stable and accurate estimation for all four objects. The observation noise of the joint torques is set to $\sigma_\tau = 1\,\text{Nm}$.

B. Dataset collection setup

We set up three scenarios to evaluate the generalization ability of our method. Fig. 6 visualizes the initial configurations of the robot arm in each scenario. Each artificial plant is fixed to the support surface. We operate the robot arm in Cartesian control mode touching the artificial plant in a horizontal direction. Each touch *sequence* pushes the robot into the plant making contact with the end effector and other distal links, until 700 joint torque measurements are collected. Each sequence is approximately 10 s in duration. The pushed displacements vary from ~5 cm on the upper part of the foxtail down to ~1 cm for the cherry tree's branches. The average final torques $\|\tau^T\|_2$ vary from 10.1 Nm for foxtail to 17.1 Nm for cherry tree. We collect datasets with different touch locations conditions:

- FULL-OVERLAP: the same location is touched but with different configurations, 10 touch sequences per object,
- PARTIAL-OVERLAP: similar locations are touched, 10 touch sequences per object,
- NO-OVERLAP: different locations with no overlap are touched, 5 touch sequences per object.

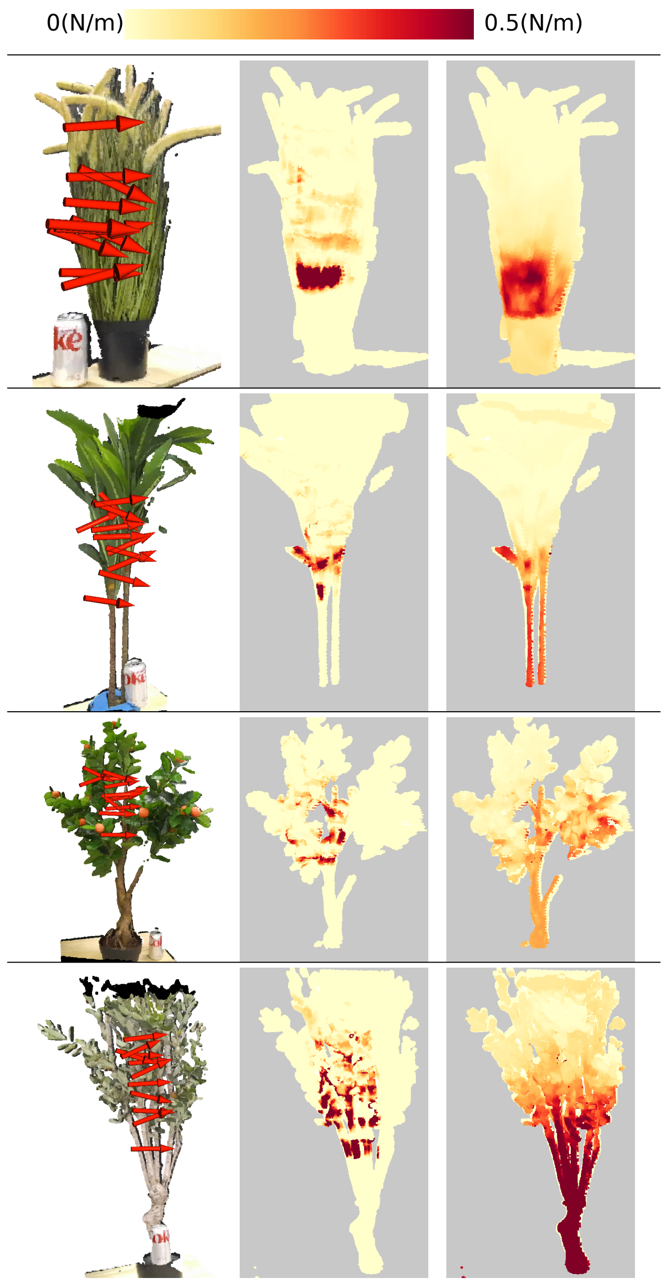
The NO-OVERLAP dataset has a smaller number of sequences because we require strictly non-overlapping touch regions and the reachable workspace of the robot arm is limited.

Each dataset with M sequences is split into train / test sets. For the FULL-OVERLAP dataset, a single touch sequence is sufficient to estimate the force response, so we train on 1 sequence and test on $M-1$ sequences. For the PARTIAL-OVERLAP and NO-OVERLAP datasets, since the touch location changes, we train on $M-1$ sequences and test on 1 sequence. Note that we do not update the stiffness estimation while evaluating on the test sequence.

C. Baseline methods

We set up three baseline learning-based methods that directly predict the torque measurements from the robot's configuration and end effector pose. The end-effector pose is a predictive feature because contact mostly occurs on the last link. Since we consider surface contacts, the exact contact locations cannot be found. Denoting robot's joint angles as $q \in \mathbb{R}^7$, the center position of the end-effector as $r \in \mathbb{R}^3$ and end-effector rotation axis $k \in \mathbb{R}^3$, the feature vector is $\phi = [q^\top r^\top k^\top]^\top \in \mathbb{R}^{13}$. Note that unlike in our method, training occurs once on the whole training sequence rather than in an online fashion. We feed the feature vectors into three baseline models:

- MLP: a two-layer multilayer perceptron with 128 hidden units trained with Adam [28] optimizer. The best performing hyperparameters were selected amongst 64, 128, and 256 units, and amongst two or three layers.
- GP: Gaussian Process regression predicts joint torques at each joint from joint angles using a Matern kernel with $\mu = 1.5$. The best performing hyperparameters were selected from Matern vs RBF kernel and several values of μ.



(a) Real object. (b) Raw VSF. (c) Extrapolated VSF.

Fig. 5. VSF estimation results of artificial plants on a scale from 0–0.5 N/m. (a) Image of the plant, with a 12 oz soda can as scale reference. Red arrows indicate the end-effector position and orientation for the initial configuration in a touch sequence. (b) Blind estimated VSF. (c) Touch+vision VSF that extrapolates stiffness using color and position information.

- k-NN: k-nearest neighbors regression using $k = 50$ nearest neighbors and an RBF kernel weights.

D. Results and discussion

a) Computation speed: Update rates of our algorithm on different objects are shown in Table II. All experiments are run on a machine with Intel(R) Xeon(R) W-2155 CPU and NVIDIA GP104GL Quadro P4000. This table shows our algorithm can achieve consistent and efficient update speeds across objects with different numbers of contact points.

(a) FULL-OVERLAP (b) PARTIAL-OVERLAP (c) NO-OVERLAP

Fig. 6. Initial configurations of the robot arm touching cherry tree for three datasets. Different colors indicate different touching sequences.

TABLE II
UPDATE SPEED FOR EACH OBJECT.

	Avg # contact points	Update speed(Hz)
Foxtail	3067 ± 1497	28.3 ± 6.2
Dracaena	4604 ± 1829	23.5 ± 6.0
Orange tree	3041 ± 1060	26.3 ± 4.9
Cherry tree	2470 ± 750	28.3 ± 4.9

b) Qualitative results: We visualize estimated VSFs for each object in Fig. 5. The three columns show real-world point clouds of artificial plants, blind VSF, and vision+touch VSF. We observe that using vision extrapolation, VSF can capture different semantic regions of the artificial plants: like soft leaves and harder branches. For foxtail in the second row, the stiffness near the base turns out to be much higher compared to its upper part. The VSF was able to effectively capture and generalize this heterogeneity even with a relatively naïve extrapolation approach.

c) Quantitative results: Tables III–V present the torque prediction accuracy of each method over the three datasets. The accuracy metric is the error $\|\hat{\tau}^t - \tau^t\|_2$ between predicted torques $\hat{\tau}^t$ and ground truth torques τ^t. We report the mean and standard deviation over all time steps in the testing sequences after being trained with the training sequences. No online updates are performed during testing. VSF has a consistently smaller prediction error compared to the baseline methods. By explicitly modeling a stiffness field, it more accurately predicts joint torques than the baselines and achieves very accurate results even with a single touch when the robot arm touches the same location (Tab. III). High accuracy is also achieved in the partial-overlap dataset (Tab. IV), in which $M-1$ sequences are used for training.

Vision-based VSF extrapolation improves on blind VSF in many cases particularly when touches do not overlap, but we observe that extrapolation can be a double-edged sword. If the first touches in a sequence lie on unusually soft or stiff regions of the object, then extrapolation will generalize those values across the entire object, possibly leading to worse predictions than the uninformed initialization. This causes a slight degradation of performance in the partial-overlap case (Tab. IV), but is overall worthwhile in the no-overlap case (Tab. V).

We also study prediction error as a function of the number of touches in Fig. 7. We see that the prediction error of our method is lower than baseline methods, particularly at small numbers of touches, and visual extrapolation helps substantially at the early stages of touching, as expected.

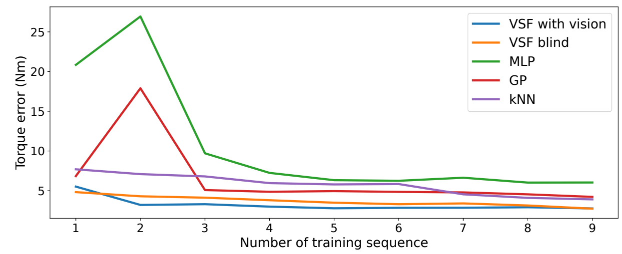


Fig. 7. Accuracy as the number of training sequences varies on PARTIAL-OVERLAP dataset of the cherry tree.

TABLE III
EVALUATION ON FULL-OVERLAP DATASET.

$\|\hat{\tau}-\tau\|_2$ (Nm)	VSF blind	MLP	GP	k-NN
Foxtail	1.0 ± 0.3	6.0 ± 6.6	3.2 ± 1.4	4.4 ± 3.4
Dracaena	1.6 ± 0.8	5.6 ± 5.0	4.6 ± 2.7	4.8 ± 3.5
Orange tree	2.6 ± 0.7	6.8 ± 6.8	3.8 ± 1.2	5.6 ± 3.5
Cherry tree	2.2 ± 0.7	11.8 ± 19.0	5.9 ± 2.1	6.3 ± 4.0

TABLE IV
EVALUATION ON PARTIAL-OVERLAP DATASET.

$\|\hat{\tau}-\tau\|_2$ (Nm)	VSF with vision	VSF blind	MLP	GP	k-NN
Foxtail	1.7 ± 1.4	1.5 ± 0.7	3.4 ± 1.8	2.6 ± 1.2	2.6 ± 1.6
Dracaena	1.7 ± 0.6	1.4 ± 0.5	11.1 ± 16.0	4.1 ± 2.1	6.0 ± 3.8
Orange tree	1.7 ± 0.6	1.8 ± 0.8	3.3 ± 1.7	3.0 ± 1.2	4.0 ± 3.0
Cherry tree	2.7 ± 0.9	2.3 ± 0.5	6.1 ± 5.4	4.2 ± 1.4	3.9 ± 1.6

TABLE V
EVALUATION ON NO-OVERLAP DATASET.

$\|\hat{\tau}-\tau\|_2$ (Nm)	VSF with vision	VSF blind	MLP	GP	k-NN
Foxtail	3.9 ± 2.5	4.5 ± 2.5	6.0 ± 3.5	5.3 ± 2.3	6.5 ± 3.3
Dracaena	2.6 ± 1.3	3.2 ± 1.2	9.1 ± 10.5	3.3 ± 1.1	7.0 ± 4.9
Orange tree	2.5 ± 1.1	3.3 ± 1.3	4.7 ± 2.0	3.7 ± 1.8	3.8 ± 2.2
Cherry tree	5.2 ± 4.4	7.4 ± 4.8	12.6 ± 8.1	8.2 ± 6.2	11.4 ± 6.8

V. CONCLUSION

In this paper, we introduce VSF, a new deformable object stiffness model which can be estimated online using a projected diagonal Kalman filter. Our real world experiments show an estimated VSF can be used to predict joint torques. We also show the VSF estimated from a small number of touches can be extrapolated to untouched regions using visual features. In future work, we intend to tackle the modeling of a more diverse set of deformable objects and also model different kinds of deformation behaviors, such as visco-elastic and plastic deformation. Another avenue of investigation is to incorporate other types of tactile sensors, such as Gelsight [29], in our pipeline to improve contact estimation.

ACKNOWLEDGMENT

This project was supported by USDA/NIFA Grant #2020-67021-32799. We appreciate UIUC Intelligent Motion Lab members for their helpful comments and suggestions.

REFERENCES

[1] C.-A. Smarr, T. L. Mitzner, J. M. Beer, A. Prakash, T. L. Chen, C. C. Kemp, and W. A. Rogers, “Domestic robots for older adults: Attitudes, preferences, and potential,” *Int. J. Social Rob.*, vol. 6, pp. 229–247, 2014.

[2] E. Kayacan, Z. Zhang, and G. Chowdhary, “Embedded high precision control and corn stand counting algorithms for an ultra-compact 3d printed field robot,” in *Robotics: Science and Systems*, 2018.

[3] N. Haouchine, J. Dequidt, I. Peterlik, E. Kerrien, M.-O. Berger, and S. Cotin, “Image-guided simulation of heterogeneous tissue deformation for augmented reality during hepatic surgery,” in *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Adelaide, Australia: IEEE, Oct. 2013, pp. 199–208. [Online]. Available: http://ieeexplore.ieee.org/document/6671780/

[4] C. Matl, J. Koe, and R. Bajcsy, “StRETcH: a Soft to Resistive Elastic Tactile Hand,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. Xi’an, China: IEEE, May 2021, pp. 11 782–11 789. [Online]. Available: https://ieeexplore.ieee.org/document/9561822/

[5] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp, “Reaching in clutter with whole-arm tactile sensing,” *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 458–482, 2013.

[6] B. Frank, C. Stachniss, R. Schmedding, M. Teschner, and W. Burgard, “Learning object deformation models for robot motion planning,” *Robotics and Autonomous Systems*, vol. 62, no. 8, pp. 1153–1174, 2014.

[7] V. E. Arriola-Rios, P. Guler, F. Ficuciello, D. Kragic, B. Siciliano, and J. L. Wyatt, “Modeling of deformable objects for robotic manipulation: A tutorial and review,” *Frontiers in Robotics and AI*, vol. 7, 2020.

[8] V. E. Arriola-Rios and J. L. Wyatt, “A multimodal model of object deformation under robotic pushing,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 2, pp. 153–169, 2017.

[9] H. Shi, H. Xu, Z. Huang, Y. Li, and J. Wu, “Robocraft: Learning to see, simulate, and shape elasto-plastic objects with graph networks,” *Robotics: Science and Systems (RSS)*, 2022.

[10] B. Shen, Z. Jiang, C. Choy, L. J. Guibas, S. Savarese, A. Anandkumar, and Y. Zhu, “Acid: Action-conditional implicit visual dynamics for deformable object manipulation,” *Robotics: Science and Systems (RSS)*, 2022.

[11] Y. Zhu, K. Lu, and K. Hauser, “Semi-empirical simulation of learned force response models for heterogeneous elastic objects,” in *IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020, pp. 1646–1652.

[12] H.-y. Chen, E. Tretschk, T. Stuyck, P. Kadlecek, L. Kavan, E. Vouga, and C. Lassner, “Virtual elastic objects,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 15 827–15 837.

[13] V. L. Popov, “Method of reduction of dimensionality in contact and friction mechanics: A linkage between micro and macro scales,” *Friction*, vol. 1, no. 1, pp. 41–62, Mar. 2013. [Online]. Available: https://link.springer.com/10.1007/s40544-013-0005-3

[14] K. Salisbury, D. Brock, T. Massie, N. Swarup, and C. Zilles, “Haptic rendering: Programming touch interaction with virtual objects,” in *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, ser. I3D ’95. Association for Computing Machinery, 1995, p. 123–130. [Online]. Available: https://doi.org/10.1145/199404.199426

[15] W. Yuan, M. A. Srinivasan, and E. H. Adelson, “Estimating object hardness with a GelSight touch sensor,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea: IEEE, Oct. 2016, pp. 208–215. [Online]. Available: http://ieeexplore.ieee.org/document/7759057/

[16] T. Bhattacharjee, P. M. Grice, A. Kapusta, M. D. Killpack, D. Park, and C. C. Kemp, “A robotic system for reaching in dense clutter that integrates model predictive control, learning, haptic mapping, and planning,” in *IEEE/RSJ Int. Conf. Intel. Robots and Systems (IROS)*, 2014.

[17] A. Albu-Schäeffer, S. Haddadin, C. Ott, A. Stemmer, T. Wimböck, and G. Hirzinger, “The dlr lightweight robot – design and control concepts for robots in human environments,” *Industrial Robot-An International Journal*, vol. 34, pp. 376–385, 08 2007.

[18] T. Pang and J. Umenberger, “Identifying external contacts from joint torque measurements on serial robotic arms and its limitations,” *IEEE Int. Conf. Robotics and Automation*, pp. 6476–6482, 2021.

[19] L. Manuelli and R. Tedrake, “Localizing external contact using proprioceptive sensors: The contact particle filter,” in *IEEE/RSJ Int. Conf. Intel. Robots and Systems (IROS)*, 2016, pp. 5062–5069.

[20] J. Bimbo, C. Pacchierotti, N. G. Tsagarakis, and D. Prattichizzo, “Collision detection and isolation on a robot using joint torque sensing,” in *IEEE/RSJ Int. Conf. Intel. Robots and Systems (IROS)*, 2019, pp. 7604–7609.

[21] B. Saund, S. Choudhury, S. S. Srinivasa, and D. Berenson, “The blindfolded robot: A bayesian approach to planning with contact feedback,” in *ISRR*, 2019.

[22] J. Bimbo, A. S. Morgan, and A. M. Dollar, “Force-based simultaneous mapping and object reconstruction for robotic manipulation,” *IEEE Robotics and Automation Letters*, vol. 7, pp. 4749–4756, 2022.

[23] K. Hauser, *Robust Contact Generation for Robot Simulation with Unstructured Meshes*. Cham: Springer International Publishing, 2016, pp. 357–373. [Online]. Available: https://doi.org/10.1007/978-3-319-28872-7_21

[24] D. Simon, “Kalman filtering with state constraints: a survey of linear and nonlinear algorithms,” *IET Control Theory & Applications*, vol. 4, pp. 1303–1318(15), August 2010. [Online]. Available: https://digital-library.theiet.org/content/journals/10.1049/iet-cta.2009.0032

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[26] G. Puskorius and L. Feldkamp, “Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks,” *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 279–297, Mar. 1994. [Online]. Available: https://ieeexplore.ieee.org/document/279191/

[27] N. M. Vural, S. Ergüt, and S. S. Kozat, “An efficient and effective second-order training algorithm for lstm-based adaptive learning,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 2541–2554, 2021.

[28] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[29] W. Yuan, S. Dong, and E. Adelson, “GelSight: High-Resolution Robot Tactile Sensors for Estimating Geometry and Force,” *Sensors*, vol. 17, no. 12, p. 2762, Nov. 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/12/2762