Learning Haptic Representation for Manipulating Deformable Food Objects

Mevlana C. Gemici and Ashutosh Saxena
Department of Computer Science, Cornell University, USA.
Email: {mevlana,asaxena}@cs.cornell.edu

***Abstract*— Manipulation of complex deformable semi-solids such as food objects is an important skill for personal robots to have. In this work, our goal is to model and learn the physical properties of such objects. We design actions involving use of tools such as forks and knives that obtain haptic data containing information about the physical properties of the object. We then design appropriate features and use supervised learning to map these features to certain physical properties (*hardness, plasticity, elasticity, tensile strength, brittleness, adhesiveness*). Additionally, we present a method to compactly represent the robot's beliefs about the object's properties using a generative model, which we use to plan appropriate manipulation actions. We extensively evaluate our approach on a dataset including haptic data from 12 categories of food (including categories not seen before by the robot) obtained in 941 experiments. Our robot prepared a salad during 60 sequential robotic experiments where it made a mistake in only 4 instances.**

I. Introduction

One important class of objects that a personal robot would need to learn about is deformable semi-solid materials such as food. It is important for the robot to model and estimate their physical properties to be able to manipulate them. Approaches based on vision (e.g., [1]) are limited because even the same types of food objects (e.g., two different bananas), which have similar visual properties, could have vastly varying material properties, depending on the conditions that they are treated at (e.g., temperature or humidity), their age (e.g., raw or ripe), or how they were processed (e.g., cooked). In many cases, these small differences can change the correct way to manipulate these objects. In this paper, we focus on using haptic data from the robot's internal sensors to perceive and manipulate food objects.

Although physical properties of objects are important for manipulation planning, mathematically modeling these properties in an explicit manner is very challenging for many object categories and requires a lot of design effort [2]. A feasible alternative is to learn representations from haptic data where the agent maps its sensory inputs from its actuators to physical properties of objects.

In this work, we propose a learning algorithm (see Fig. 2) that takes as input haptic sensory signals of a robot obtained while performing actions on the food objects. We then extract useful features from these haptic signals and map them to physical properties of the objects through supervised learning. Using this mapping, we then perform unsupervised learning and clustering based on Diritchlet Processes (DP) to represent the physical properties compactly. Through this representation, we integrate multiple observations about an object's properties obtained at different times into a compact *belief* and infer reliable strategies to manipulate it through appropriate task-oriented actions.

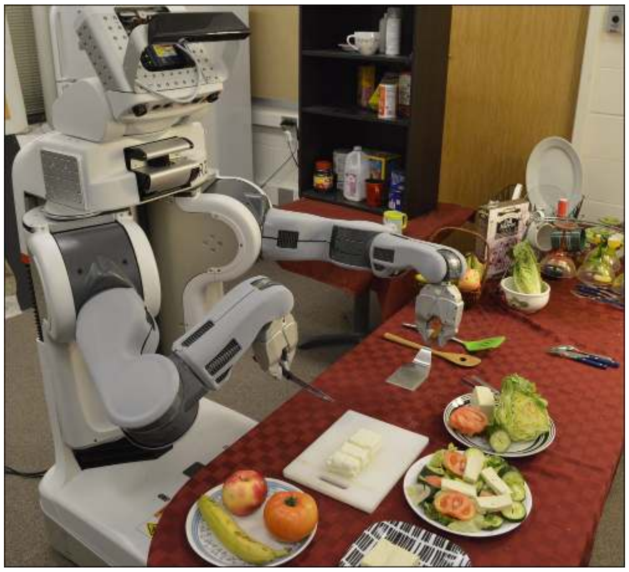


Fig. 1: We use haptic sensors of a personal robot to learn a data-driven representation of the physical properties of food objects. We use learning techniques to create manipulation strategies involving actions that use household tools in order to prepare meals.

We extensively test our approach on a large dataset of haptic information gathered in **941** experiments from **12** different categories of food (see Fig. 3), including categories that are quite similar (e.g., different types of bread) and challenging to differentiate. We show that actions have different information gathering abilities and it is possible to extract information from manipulation experiences of the robot. Our experiments show that the robot can obtain relevant information about the food items (both from seen and unseen object categories) in a safe way and is able to determine the correct way to manipulate those objects to reach simple goals. Finally, we use our learned models and manipulation strategies on the robot to build a simple meal.

II. Related Work

Robot Manipulation of Objects. While previous works on robot manipulation have focused on rigid objects (e.g., [3], [4]), there has been some recent work on manipulating articulated objects (e.g., [5], [6]) and foldable objects (e.g., towels [7]). Many of these works use tactile sensing for manipulation (e.g., [8]) but none of them consider the perception of internal properties of objects when they are critical for manipulation.

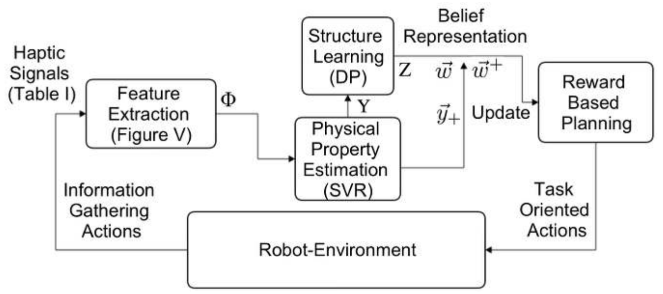


Fig. 2: **Overview of our system**, showing the different components.

Several previous works showed that haptic data is useful for predicting specific object properties, e.g., presence of obstacles [9], human contact forces [10], hardness of objects [11], and elasticity of objects [12]. Frank et al. [13] considers motion planning with deformable objects that are possible to explicitly model and simulate. However, perception of fine differences in physical properties of very complex objects through learning and task planning based on these properties has not yet been explored extensively.

Some robotics work focused on food manipulation such as [14], [15] that reason about how objects are changed by actions through a rule-based ontology and build a knowledge base from natural language. Bollini et al. [16] presented a vision based cookie baking system that uses compliant controllers similar to ours. Kormushev et al. [17] and Beetz et al. [18] are focused on making pancakes by learning new motor skills trough reinforcement learning and using vision-guided task planning by utilizing natural language instructions respectively. None of these works directly learn object properties though exploratory actions and informative features computed from haptic sensors.

In Chitta et al. [19], the deformative properties of objects (e.g., empty bottles) are perceived through tactile information obtained from grippers. They also show that the recognition of non-visual properties of objects is challenging even for humans through a comparative study and motivate the use of general purpose features in a learning task similar to ours.

In Chu et al. [20], a robot learns the surface properties of objects by touching them with special-purpose hardware. However, we focus more on mechanically oriented properties directly relevant for manipulation purposes, which we learn using motor efforts, forces, torques and end-effector configuration in addition to fingertips sensors. We also use household tools in our information gathering actions which allows us to deal with object even when they are dangerous to contact with robotic hardware (e.g., wet). Lastly, we focus on learning internal physical properties of objects as compared to surface properties in this previous work.

Learning Algorithms for Task Planning. Dirichlet processes (DPs) have been applied to model different aspects of Markov models, such as in an HMM [21], POMDPs [22] or ILCRFs [23]. In other works, different probabilistic models have been used to model different aspects in the RL/MDP setting. For example, Gaussian processes for reinforcement learning [24] and Bayesian reinforcement learning [25], [26].

There are several works that considered the problem of compactly representing the states (or belief states) in POMDPs [25], [27], [28] . Representation of information is an important issue in computability of optimal task planning and motivates our approach for integrating multiple observations as compact beliefs represented through a generative model obtained from previous observations.

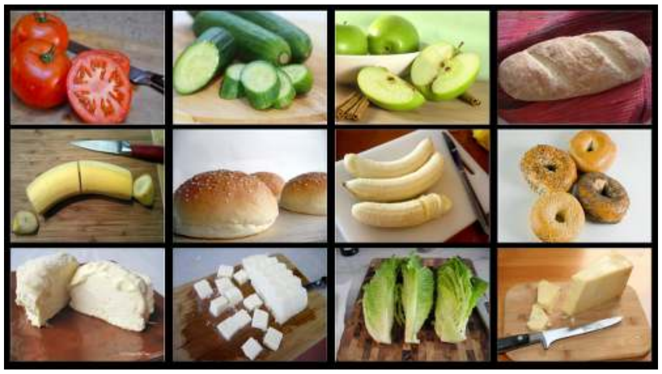


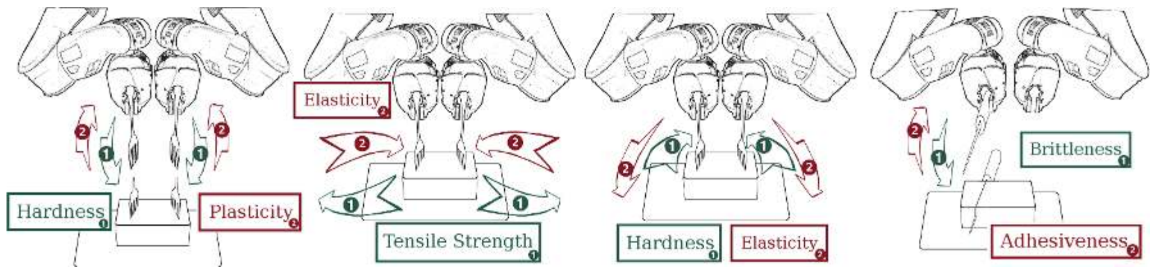
Fig. 3: **Food categories:** We consider manipulating 12 categories of food: tomatoes, cucumbers, apples, bread, bananas, soft buns, peeled bananas, bagels, cream cheese, tofu, lettuce and cheddar cheese. Our haptic dataset has a total of **941** instances extracted from these categories.

Visual identification of food. Several works consider food quality classification using computer vision (e.g., [29], [30]). Although these methods produce labels about food items from vision, these labels alone are not very informative from the standpoint of manipulation. Recent work done on inferring mechanical properties from visual properties, such as [31], are dependent on the object categories and limited to objects with visual cues that would give a good estimation of material properties. Other recent works [32], [33] label objects and their attributes in 3D point-clouds, but not their physical properties. Regardless, these works could be used as complementary inputs to future extensions of our algorithm combining haptic and visual perception.

III. Physical Properties of Food

For a robot to manipulate complex deformable objects such as food, it is necessary to begin with an estimate of the objects's physical properties because manipulation strategies can change significantly with differences in physical properties for many complex tasks [34]. For example, the force needed to insert a knife into a block of cheese would depend on the hardness of the cheese. If the cheese is soft and sticky enough, one may need to only insert the knife, then turn it on the way out, for getting the cheese. On the other hand, if the cheese is brittle and not sticky, then one needs to cut it first, twist the knife to detach the piece, then pick up the piece with a fork or fingers. With different types of food (Fig. 3), strategies for accomplishing useful tasks change drastically.

While food scientists can measure physical properties with special-purpose devices, a manipulation robot can only estimate them using the sensory information obtained from its joints and fingertips. For example, a puncture tester [35] advances a small punch into the food in order to obtain a force-distance curve that can be distinctive and diagnostic.



(a) Vertical Pierce with Forks (b) Lateral Split with Forks (c) Circular Bend with Forks (d) Vertical Cut with Knife

Fig. 4: **Actions designed to gather information about the physical properties of objects.** Arrows indicate the direction of the applied forces in the first and second motions for the actions. For each motion, we indicate the most important physical properties that the action is designed to extract. For example, in action (b), we desire to measure the tensile strength of an object as we apply an outward force (1) after piercing an object. Similarly, while removing a knife (2) from the object in action (d) we desire to determine whether the object is adhesive or not.

A robot poking a fork into a chunk of food can implicitly obtain similar, albeit not as precise, information.

In our approach, we will describe the space of physical properties along the following axis: *hardness, plasticity, elasticity, tensile strength, brittleness* and *adhesiveness*. We give each axis a score between 0 and 1, obtaining a six-dimensional property space $[0,1]^6$, which we will call '*physical properties*' of objects. Although these properties are not completely independent of each other, they cover most of the material properties relevant for our manipulation purposes. A robot can obtain a (noisy) estimate of these properties from its sensors by performing information-gathering actions, as discussed in the next section.

IV. Action Design for Gathering Haptic Data

In order to estimate the physical properties of food objects, we designed robotic actions to extract useful physical information from them in a human-inspired fashion. Our robotic platform, the PR2, consists of two 7-DOF robotic arms attached to a mobile base (see Fig. 1). We use controllers [36] based on temporal force/torque feedback that work by dynamically adjusting desired forces and currents at the joints of the arm to reduce delays of reaching the goal configurations in a trajectory. Since we focus on food items, our actions utilize household tools that were designed to manipulate them, such as forks and knives. Fig. 4 shows our information-gathering actions.

Each of these actions is designed with a subset of the physical properties in mind. For example, a vertical piercing action with fork would obtain information about the hardness of an object during the downward motion due to resistance from harder objects. It would obtain information on the plasticity of the object as it tries to remove the fork with an upward motion because of the positive feedback from more elastic objects. Similarly, lateral splitting motion with two forks would have different force and torque response from objects with different tensile strength and brittleness as the gap is increased. As the motion is reversed, the robot would have to spend less effort with objects that have smaller plasticity. (See Fig. 4 for other examples.)

V. Features for Haptic Learning

We use a variety of haptic sensory signals (Table I) obtained during the execution of actions as input to our system in Fig. 2. In this section, we explain how we extract time-frequency histogram features from each haptic input's power spectrum and construct a feature vector for each object instance-action pair, $\mathbf{\Phi}_i^a$, to use in our learning algorithms.

TABLE I: Haptic Inputs and Feature Sets.

Description of Haptic Input	# Inputs
Configuration (C)	**7**
C1. Position of end effectors	3
C2. Orientation of end effectors	4
Effort (E)	**15**
E1. Desired (based on delays from object) joint efforts of arms	7
E2. Actual applied joint efforts of arms	7
E3. Deviation in joint efforts $\|\|E1 - E2\|\|$	1
Dynamics (D)	**29**
D1. Forces applied at end effectors	3
D2. Torques applied at end effectors	3
D3. Fingertip pressures (raw & average)	23

Our system uses three sets of haptic inputs to compute features — *configuration* (C), *effort* (E) and *dynamics* (D). *Configuration* features are computed from the configuration of the end effectors in 3-D space and captures the deviation of the end effector from the action trajectory due to object's properties. *Effort* features are determined from control-level current signals which represent the power used by each joint of the arms. While E1 represents the efforts (power) that are desired by the controller based on the delay caused by the object, E2 represents efforts that are actually applied to the joints while trying to reach E1. E3 represents the magnitude of the difference between E1 and E2. *Dynamics* features are obtained from forces and torques applied to the object at the end effector. This set also includes fingertip pressure changes (both raw and average), which are transferred from tools as the objects are manipulated. Table I summarizes the haptic input descriptions, the space each is represented in, and the number of inputs obtained per gripper (Vertical cut action uses only one gripper and others use both).

We compute each of our features by determining the power spectrum of the corresponding haptic input, which is in the time domain[1]. Fig. 5 presents an example haptic input (E3. Deviation in joint efforts) obtained while performing the *Lateral Split with Forks* action and how its power spectrum varies with different object categories that are being

[1]In order to determine the power spectrum we compute a Discrete Fourier Transform (DFT) at $0-16$Hz range for each haptic input.

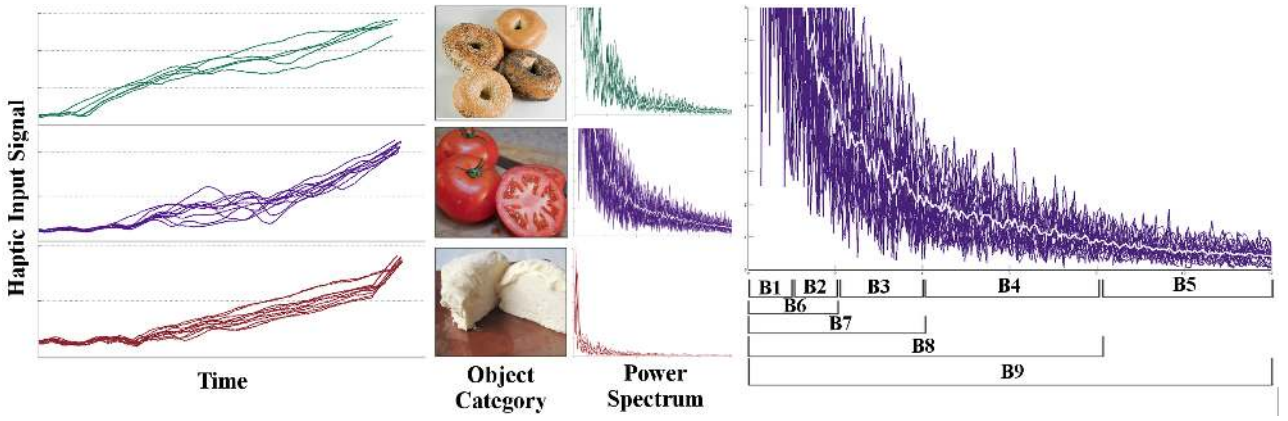


Fig. 5: **Feature Extraction**: On the left, the 'Deviation in joint efforts' (Table I entry E3) shown as *Lateral Split with Forks* action is executed on different object categories (middle). This haptic input has a higher low-frequency content for bagel category (top) and has a higher high-frequency content for tomato category (middle) compared to cream cheese category (bottom). On the right side, we show our binning strategy where each bin collects the power content by integrating its respective interval.

manipulated.

Once we obtain the spectrum of a haptic input, we create the corresponding features by computing a 5-bin histogram integrated between different frequencies (uniform in log-scale between $0-16$Hz). Additionally, we compute another 4-bin histogram where the bins accumulate the power content until the frequency they represent (see right side of Figure 5). Cumulative binning helps in capturing various nonlinear functions of the original power content.

For the rest of this paper, we will represent this feature vector as $\mathbf{\Phi}_i^a \in \mathbb{R}^{9\times h}$, where a is the action executed to obtain haptic inputs from instance i and h is the number of inputs used in learning which depends on the input sets chosen (first column of Table I). For example, if we use all the haptic inputs (C+E+D), then $h = 2\cdot(7+15+29)$ and $\mathbf{\Phi}$ has 918 features (for actions that use both grippers). High dimensionality of the feature space for some sets of inputs motivates us to use learning algorithms that are able to cope with such data. We discuss our approach for mapping the computed features to physical properties of objects in the next section.

VI. Learning

A. Dataset

For our supervised learning task, we used three human experts (including one of the authors) to label the properties of the food categories involved in training (all categories from Fig. 3 except tofu). Although it is a hard task for a human to reliably estimate the absolute physical properties of food categories (e.g., average elasticity modulus of cream cheese in N/m^2), ordering these food categories according to their physical properties is relatively easier and disagreements on the labelings are arguably far less common.

In order to label the food categories, human experts are asked to order them for each pyhsical property independently. In this ordering, they are allowed to have equivalence relationships between two categories for individual physical properties as well. Once the ordering is decided for a property, it is projected to the $[0,1]$ interval linearly and the labels for that property are determined. For example, if the *elasticity* ordering is [*Soft buns* $\succ$ *Cheddar cheese* $\succ$ *Cucumbers* $\simeq$ *Bread*], then the corresponding elasticity labels would be $\mathbf{y}^{elas} = [1, 0.5, 0, 0]$ for those categories. In reality, the ordering includes all 12 categories mentioned.

During data collection, our robot performed information gathering actions discussed in Section IV, on all 12 categories of food in 941 experiments (more than 350 food objects). For many instances of objects, more than one info-gathering action was executed on it and features from different actions were obtained. Lastly, 80% of our dataset was used for training and 20% was used for evaluating our algorithms.

B. Learning Physical Properties

TABLE II: **Normalized RMSE results** (overall for all physical properties) for learning $\mathbf{y}$. Rows show the set of features used in learning, and columns show the different information-gathering actions presented in Section V and Figure 4 respectively.

Info-gathering Action	Vert. Piercing	Lat. Splitting	Circ. Bending	Vert. Cut w/ Knife	Best Action
C Only	.17	.14	.15	.19	.14
E Only	.14	.13	.13	.18	.13
D Only	.11	.11	.07	.09	.07
C+E	.11	.10	.09	.14	.09
C+D	.09	**.05**	**.06**	.09	.05
E+D	.09	.08	.09	.11	.08
C+E+D	**.06**	.06	.09	**.09**	.06
Best Set	C+E+D	C+D	C+D	C+E+D	

Given a feature vector $\mathbf{\Phi}_i^a$ computed from haptic inputs obtained while performing action a on an object i, our goal is to estimate the labels of the object, $\mathbf{y}_i = \{y_i^1 ... y_i^6\}$, where each $y^n \in [0,1]$ represents one of the physical properties. We take a large-margin approach to learning the parameters of each action's regression model H_a, using ν-SVR algorithm implemented in the LibSVM library [37] from training samples obtained during data collection.

In Table II, we present the test results for each action's regression model learned using linear kernel ν-SVR. We also present the performance of different sets of features described in Section V. Our metric is normalized RMS error corresponding to mean error rate aggregated for all properties. In Fig. 6, we present the estimation results projected

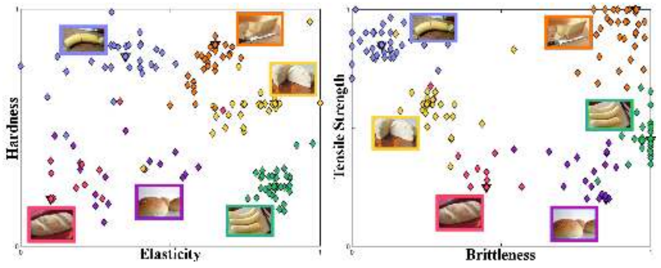


Fig. 6: **Category-wise estimates** projected to the space of physical properties, aggregated for all four actions using the best set of features of each action (See Table II). Diamonds show the estimates for the test set samples, triangles show ground truth label of a category. We show only a subset of categories in our dataset for better visualization. (Best viewed in color.)

onto the physical property space for different categories of food objects (only a subset of categories shown). Although the full property space is six dimensional, we show *hardness-elasticity* and *tensile strength-brittleness* subspaces only, due to space concerns.

C. Discussion on Results of Estimating Physical Properties

Our results in Table II show that for all actions, *dynamics* features are more informative than *configuration* and *effort* features. *Effort* features are control level features that represent raw actuator power used. From these results, we can conclude that using robot configuration or raw actuator information directly to learn the physical properties of objects is challenging compared to learning from dynamics (forces, etc.) at the point of contact with the objects. Nonetheless, for some actions, using all the features together results in the best model.

We see in Fig. 6 that, although different object properties are predicted well for all actions, some of the actions were not able to differentiate well the elasticity of two very similar categories (*Bread*: red- *Soft buns*: purple). One action that did particulary good in estimating plasticity of these categories (and others) was *Circular Bend* action, which was designed to measure elasticity specifically. Since we have determined in this section that certain sets of features are more useful for certain actions in terms of predictive ability, we use the best set from Table II for each action in order to discover useful *haptic categories* in the next section.

D. Discovering Haptic Categories through Dirichlet Process

In this section, we present a method to learn the structure of the distribution of food categories in the physical property space. Our motivation for this task is two-fold: First, we wish to identify the regions of the property space that are occupied by different categories of food objects and exploit the similarities between these regions for planning task-oriented manipulation actions to accomplish useful goals. Second, we wish to compactly represent *beliefs* about the object properties. A belief integrates multiple *observations* in the property space obtained from performing different information gathering actions. However, the complete belief space over continuous variables, as in our case, is infinite dimensional. This requires us to constrain the set of possible

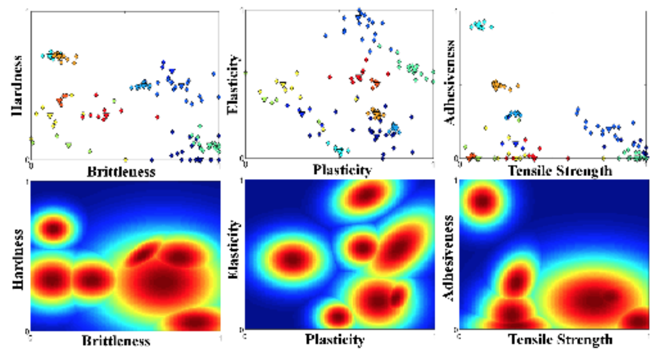


Fig. 7: **Haptic categories** learned (bottom) from estimated properties of training samples (top) through DP for the *Lateral Split* action. Best viewed in color.

beliefs to reflect the overall structure in the property space (see [27]) due to computational limitations.

In our terminology, the unit of structure is called a *haptic category*. A haptic category may be representative of a specific object category (e.g., cucumber) that can be easily differentiated from other categories by an information gathering action. It can also represent a number of object categories that the action perceives as similar. In our learning approach, we wish to represent each food object as being generated from a mixture of haptic categories.

In order to learn these categories, we do unsupervised learning on estimated properties, $\mathbf{y}$, of the samples from our training set. Since the number of haptic categories underlying the real structure is unknown, and potentially infinite, we use Diritchlet Processes [22], [23]. Discovering new haptic categories from the observations in the property space follows three steps:

- Perform an action a, obtain i^{th} feature vector $\mathbf{\Phi}_i^a$, and estimate the property labels $\mathbf{y}_i$ (called an *observation*) using the regression model H_a obtained through supervised learning. (Section VI-B).
- Assign $\mathbf{y}_i$ to an old haptic category or start a new one based on a Dirichlet Process (DP).
- Update the parameters of each haptic category $\forall z$: $\mu_z, \Sigma_z \in Z$ after the new assignment.

We describe the assignment step more formally below. Let z_i^a be the assignment of the i^{th} sample to haptic category z of action a. Now the assignment of this sample to existing category in Z (or a new category) for action a is as follows:

$$z_i^a = z|\boldsymbol{z}^{-i} = \begin{cases} \frac{n_z^{-i}}{N-1+\alpha} \cdot \mathrm{P}(\mathbf{y}_i;\xi_z) & z \text{ already} \in Z \\ \frac{\alpha}{N-1+\alpha} & \text{otherwise} \end{cases} \tag{1}$$

where superscript $-i$ denotes everything except the i^{th} instance, n_z^{-i} equals the number of data points assigned to category z excluding i^{th} for action a, and α is a Dirichlet concentration parameter. We use Gaussians for the distributions $P(\mathbf{y}_i;\xi_z) = \mathcal{N}(\mathbf{y}_i;\mu_z,\Sigma_z)$. Similarly, we consider the structure obtained as an infinite Gaussian Mixture Model and represent our *beliefs* as a linear mixture of these categories. Fig. 7 shows an example set of haptic categories obtained for the *Lateral Split* action.

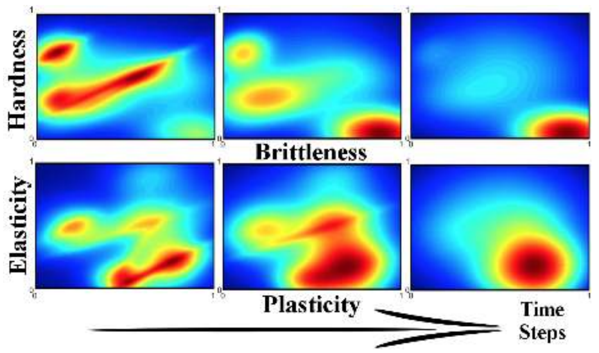


Fig. 8: **Belief update** as more observations are obtained from information gathering actions that are being performed on a tofu object during robotic experiments (Section VII-E). Only two of the subspaces are shown. Best viewed in color.

E. Belief Representation and Update

We represent a belief over an object's properties as a linear combination of haptic categories, Z, obtained from the Diritchlet Process for all actions: $b(\mathbf{y}) = \sum_{z \in Z} w_z \cdot \mathcal{N}(\mathbf{y}; \mu_z, \Sigma_z)$, where the weight vector $\mathbf{w}$, represents the mixture and $\{\mu_z, \Sigma_z\}$ represents the parameters of haptic category z. We integrate multiple observations for a given object by updating the belief weights using a new observation $\mathbf{y}_+$ in property space:

$$w_z^+ = (\lambda w_z)^{\frac{m}{(m+1)}} \cdot \left[\frac{\sqrt{|\Sigma_z|}}{\lambda^m} e^{(\sqrt{[\mathbf{y}_+ - \mu_z]\Sigma_z^{-1}[\mathbf{y}_+ - \mu_z]^T})} \right] \quad (2)$$

where m is the number observations used for determining the previous belief b, λ is the decay parameter for previous observations (chosen to be 0.9) and $\mathbf{w}^+$ is the new mixture weight vector after the update which we normalize to maintain $1^T\mathbf{w}^+ = 1$.

We show the belief update in Fig. 8 as actions *Vertical Pierce* and *Vertical Cut* are performed on a tofu object consecutively. The initial belief (column 1) has a uniform weight for each class obtained in DP. This corresponds to a prior probability obtained from previously seen examples. Therefore, the belief update starts with this prior and not the naive uniform distribution over the whole property space. As observations are obtained in 2^{nd} column and 3^{rd} column in order, the belief changes towards an object with very low *hardness*, high *brittleness* and high *plasticity* which is correct for tofu. Once the robot reaches the belief in column 3, it can use this information for choosing the correct way to manipulate this object. For example, it can move the object with spatulas instead of its grippers, since an object with high plasticity and brittleness may be deformed when grasped or damage the grippers.

In the example above, the tofu category is not seen during the training stage and therefore Z cannot include a haptic category directly representing the tofu category. However, because there are similar categories of food objects seen during training such as cream cheese and peeled banana, it is possible to obtain a good representation even for tofu as a combination of haptic categories representing the properties of these other food categories.

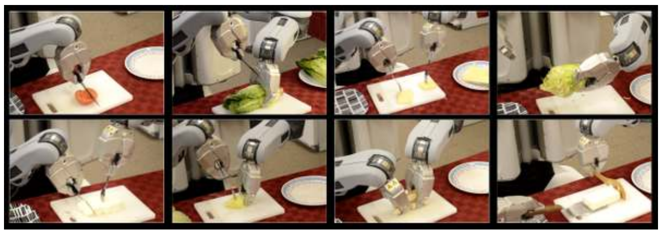


Fig. 9: **Task-oriented actions** used by our robot, involving use of grippers, a knife, spatulas and forks, in different fashions. Eight (out of ten) are shown, due to space constraints.

VII. Experiments and Results

A. Experimental Setup

We test our perception and belief integration approach on the task of preparing a simple meal. We have four *subgoals* that we wish to accomplish: *Cut food object*, *split object into two pieces*, *flip-turn object*, *pick up and move object*. All of these subgoals require the robot to first perceive the physical properties of object and then use appropriate tools to perform task-oriented actions to reach the subgoals.

We design 10 additional task-oriented actions (see Fig. 9)—designed to help reach specific subgoals in different ways. For the subgoals of *cutting* and *splitting*, we created three actions that use different tools (such as using a fork or a gripper to stabilize the object while cutting) and different strategies of execution such as cutting with a downward motion versus cutting with a saw-like motion. For the other subgoals, *flipping* and *moving*, we created two actions each that either use spatulas or a gripper to accomplish the task.

B. Manipulation Planning

In order to archive manipulation goals such as cutting a food object, we wish to create strategies that balance the *exploration* of object properties and the *exploitation* of actions that can accomplish useful tasks. In this work, we use a reward based method, similar to the Reinforcement Learning setting (e.g., [17]), that integrates our approach for belief representation introduced in the previous section. For a given goal and a belief about the object, we wish to determine the action that will produce the highest expected immediate reward. Although it is possible to plan ahead with larger or infinite horizons (see POMDPs [22], [25]), it is beyond the scope of this work.

In order to determine the expected immediate reward for an action-belief pair, we construct a reward function defined over the property space for each action based on the knowledge of our human experts. Although it is possible to determine the reward functions through training, such as Inverse Reinforcement Learning [38], this requires human demonstration on the task we wish to accomplish. Instead, in this work, for a given action, the reward for a given point in the property space is determined from a distance-weighted interpolation of rewards of neighbouring object categories (determined by the experts).[2] Then we generate Monte Carlo

[2]We leave out the details due to space concerns. Please refer to additional material on our website for examples of reward functions.

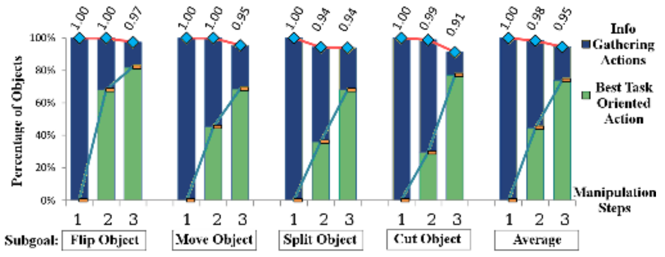


Fig. 10: **Performance of our reward based approach in time.** For a given subgoal, each column represents the percentage of info-gathering actions and the best task-oriented action chosen in different steps of the manipulation for the objects in our test set.

samples from the belief distribution and choose the action with the highest total reward for the samples in the next step of manipulation to greedily maximize total expected reward.

When defining the reward functions, we have to ensure that the rewards for information gathering actions are constructed such that, when the robot is not very sure of the properties of the object, it prefers to gather more information before performing risky, task-oriented actions. In this work, we manually tuned the reward functions for our manipulation task for a reasonable level of exploration and exploitation.

C. Baseline Algorithms for Comparison

We consider two other methods as baselines:

SVR-NN Algorithm: In this approach, the algorithm estimates the material properties of an object by performing a random information gathering action. Based on the estimation, it determines the nearest object category in the space of properties. Then the task-oriented action with the best expected reward for that object category is chosen to reach the subgoal. Therefore, if the object properties are predicted well, it always chooses the best action. However, if the distance between the nearest category label and the estimate in the property space is greater than half the mean distance between category labels, then a new information gathering action is performed in the next step of manipulation to replace the previous estimate.

Random Strategy: Given a subgoal, this algorithm chooses either an information gathering action or one of the actions that are designed to reach that subgoal. It disregards actions designed to reach other subgoals.

In the next section, we evaluate the performance of our reward based approach on our test set and make a comparison against the baseline methods.

D. Manipulation Results

Fig. 10 shows the performance of our reward based algorithm for the objects in our test set in different time steps of manipulation. In the absence of knowledge about the properties, our algorithm first performs an info-gathering action (randomly chosen). As it makes an estimate about the properties of the object and reaches a decisive belief in later steps, the algorithm performs a task-oriented action to reach the subgoal. In this graph, we see that a belief represented by a mixture of haptic categories is able to integrate multiple

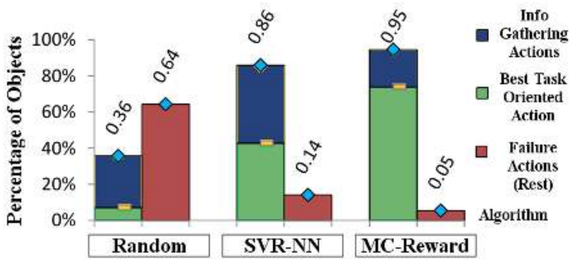


Fig. 11: **Our approach vs. baseline algorithms.** This graph compares the performance of our manipulation approach with some baseline algorithms. The algorithms are allowed to perform three steps of manipulation (similar to Figure 10) and distribution of chosen actions are presented for all the objects in our test set.

observations to make reliable decisions as more information is gathered in time.

We compare the performance of our reward based manipulation approach against the baseline algorithms in Fig. 11 where the algorithms are allowed to perform the first three steps of manipulation according to their strategy. While using the Random Strategy results in a failure 64% of the time, using the SVR-NN algorithm reduces failure rate to 14%. SVR-NN works very well when we predict the physical properties of the object very precisely. However, in some cases, the uncertainty associated with the prediction is significant and the observation in the property space is closer to a wrong object category. These cases lead to wrong manipulation decisions in this algorithm.

Our reward based approach significantly reduces the failures to 5%. Furthermore, the number of times that the algorithm has already performed the best action and reached a subgoal by the third step of execution is almost doubled compared to SVR-NN algorithm. This means that for most of the objects, one or two information gathering actions was enough to determine the best task oriented action to reach the subgoal. We associate this large improvement with the integration of new observations about an object with previous knowledge by maintaining a belief that is structured through *haptic categories* that are learned. In comparison, SVR-NN algorithm does not have a way to integrate multiple observations to balance exploration and exploitation since it greedily decides on the task-oriented action for the closest category without considering the structure of observations in the property space.

TABLE III: **Overall success rate of our robot** as it prepares a salad by performing different subgoals in a total of 60 trials. While using a random strategy would result in the best choice (success) 37% of the time, our algorithm chooses the best action for the objects in **93%** of all the trials.

Subgoal	Flipping	Moving	Splitting	Cutting	All
Trials	9	7	15	29	60
Success	8	6	15	27	56
Failure	1	1	0	2	4

E. Robotic Experiments and Results

We used our learning algorithms and our reward based manipulation approach to test the final performance of executing different subgoals in sequence to prepare a salad with

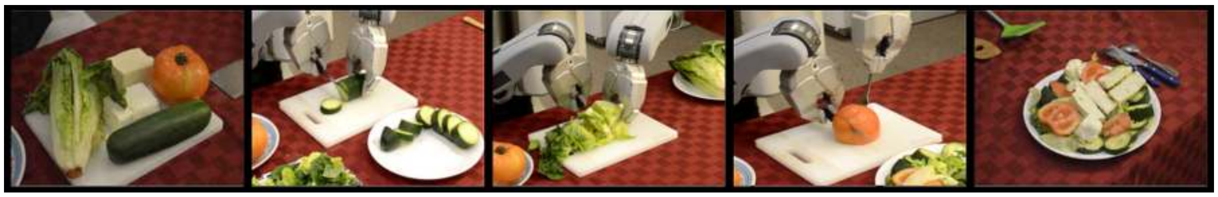


Fig. 12: **Results showing PR2 making a salad.** (Left) Unprepared food objects, (Middle) PR2 performing manipulation actions after learning about objects, and (Right) prepared salad.

PR2 (subgoals were given to the robot). In order to test the generalization of our algorithm to new object categories, we also included a new category (tofu) not seen during training.

Our algorithms performed well in most trials, failing only 4 times out of 60 trials. Table III shows the number of times each subgoal had to be executed and the overall success rate for each of them. Fig. 12 shows a few snapshots as the salad is prepared. Explanatory videos of different task-oriented actions given in Fig. 9 and the robot using our algorithms while preparing the salad is available at: `http://pr.cs.cornell.edu/hapticmanipulation/`

VIII. Conclusions

In this work, we have presented a learning algorithm for manipulating food objects with complex physical properties. Our robot models the properties through haptic inputs, computes the features, learns haptic categories and executes correct manipulation actions to reach useful goals. Using the learned belief representation and manipulation strategies, our robot attained an overall success rate of 93% in robotic experiments.

Acknowledgements

This work was supported by ARO award W911NF- 12-1-0267, and by Microsoft Faculty Fellowship and NSF Career Award (to Saxena).

References

[1] S. Yang, M. Chen, D. Pomerleau, , and R. Sukthankar, “Food recognition using statistics of pairwise local features,” in *CVPR*, 2010.

[2] L. Kunze, M. E. Dolha, E. Guzman, and M. Beetz, “Simulation-based temporal projection of everyday robot object manipulation,” in *AAMAS*, 2011.

[3] A. Saxena, J. Driemeyer, and A. Ng, “Robotic grasping of novel objects using vision,” *IJRR*, vol. 27, no. 2, p. 157, 2008.

[4] K. Hsiao, P. Nangeroni, M. Huber, A. Saxena, and A. Ng, “Reactive grasping using optical proximity sensors,” in *ICRA*, 2009.

[5] D. Katz, Y. Pyuro, and O. Brock, “Learning to manipulate articulated objects in unstructured environments using a grounded relational representation,” in *RSS*, 2008.

[6] A. Jain and C. Kemp, “Improving robot manipulation with data-driven object-centric models of everyday forces,” *Autonomous Robots*, vol. 35, no. 2-3, pp. 143–159, 2013.

[7] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel, “Cloth Grasp Point Detection based on Multiple-View Geometric Cues with Application to Robotic Towel Folding,” in *ICRA*, 2010.

[8] K. Hsiao, L. Kaelbling, and T. Lozano-Perez, “Task-driven tactile exploration,” in *RSS*, 2010.

[9] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp, “Reaching in clutter with whole-arm tactile sensing,” *IJRR*, vol. 32, no. 4, 2013.

[10] K. Hawkins, C. King, T. Chen, and C. Kemp, “Informing assistive robots with models of contact forces from able-bodied face wiping and shaving,” in *RO-MAN*, 2012.

[11] H. Yussof, M. Ohka, J. Takata, Y. Nasu, and M. Yamano, “Low force control scheme for object hardness distinction in robot manipulation based on tactile sensing.,” in *ICRA*, 2008.

[12] B. Frank, R. Schmedding, C. Stachniss, M. Teschner, and W. Burgard, “Learning the elasticity parameters of deformable objects with a manipulation robot,” in *IROS*, 2010.

[13] B. Frank, C. Stachniss, N. Abdo, and W. Burgard, “Efficient motion planning for manipulation robots in environments with deformable objects,” in *IROS*, 2011.

[14] M. Tenorth and M. Beetz, “A unified representation for reasoning about robot actions, processes, and their effects on objects,” in *IROS*, 2012.

[15] D. Nyga and M. Beetz, “Everything robots always wanted to know about housework (but were afraid to ask),” in *IROS*, 2012.

[16] M. Bollini, J. Barry, and D. Rus, “Bakebot: Baking cookies with the pr2,” in *IROS PR2 Workshop,*, 2011.

[17] P. Kormushev, S. Calinon, and D. Caldwell, “Robot motor skill coordination with em-based reinforcement learning,” in *IROS*, 2010.

[18] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, and M. Tenorth, “Robotic roommates making pancakes,” in *Humanoids*, 2011.

[19] S. Chitta, M. Piccoli, and J. Sturm, “Tactile object class and internal state recognition for mobile manipulation,” in *ICRA*, 2010.

[20] V. Chu, I. Mcmahon, L. Riano, C. G. Mcdonald, Q. He, J. M. Perez-tejada, M. Arrigo, N. Fitter, J. C. Nappo, T. Darrell, and K. J. Kuchenbecker, “Using Robotic Exploratory Procedures to Learn the Meaning of Haptic Adjectives,” in *ICRA*, 2013.

[21] K. Heller, Y. The, and D. Gorur, “Infinite hierarchical hidden markov models,” in *ICML*, 2009.

[22] F. Doshi-Velez, “The infinite partially observable markov decision process,” in *NIPS*, 2009.

[23] Y. Jiang and A. Saxena, “Infinite latent conditional random fields for modeling environments through humans,” in *RSS*, 2013.

[24] Y. Engel, S. Mannor, and R. Meir, “Reinforcement learning with gaussian processes,” in *ICML*, 2005.

[25] S. Ross, B. Chaib-draa, and J. Pineau, “Bayesian reinforcement learning in continuous pomdps with application to robot navigation,” in *ICRA*, 2008.

[26] S. Ross, J. Pineau, B. Chaib-draa, and P. Kreitmann, “A bayesian approach for learning and planning in partially observable markov decision processes,” *JMLR*, 2011.

[27] N. Roy and G. Gordon, “Exponential family pca for belief compression in pomdps,” in *NIPS*, 2003.

[28] J. Porta, M. Spaan, and N. Vlassis, “Robot planning in partially observable continuous domains,” in *RSS*, 2005.

[29] M. Z. Abdullah, J. Mohamad-Saleh, A. S. Fathinul-Syahir, and B. M. N. Mohd-Azemi, “Discrimination and classification of fresh-cut starfruits (Averrhoa carambola L.) using automated machine vision system,” *J Food Engineering*, vol. 76, pp. 506–523, 2006.

[30] J. Blasco, N. Aleixos, and E. Molt, “Machine vision system for automatic quality grading of fruit,” *Biosys Eng.*, vol. 85, no. 4, 2003.

[31] S. Matiacevich, D. Mery, and F. Pedreschi, “Prediction of mechanical properties of corn and tortilla chips by using computer vision,” *Food and Bioprocess Technology*, vol. 5, pp. 2025–2030, 2012.

[32] C. Wu, I. Lenz, and A. Saxena, “Hierarchical semantic labeling for task-relevant rgb-d perception,” in *RSS*, 2014.

[33] H. Koppula, A. Anand, T. Joachims, and A. Saxena, “Semantic labeling of 3d point clouds for indoor scenes,” in *NIPS*, 2011.

[34] K. Matheus and A. Dollar, “Benchmarking grasping and manipulation: Properties of the objects of daily living.,” in *IROS*, 2010.

[35] S. Sahin and S. Sumnu, *Physical Properties of Foods*. Springer, 2006.

[36] J. Barry. `http://wiki.ros.org/ee_cart_imped`, accessed in 2013.

[37] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Trans Intel Sys Tech*, vol. 2, pp. 27:1–27:27, 2011.

[38] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *ICML*, 2004.