Object-centric Representations for Interactive Online Learning with Non-Parametric Methods

Nikhil U. Shinde, Jacob Johnson, Sylvia Herbert, and Michael C. Yip

2023 IEEE 19th International Conference on Automation Science and Engineering (CASE) | 979-8-3503-2069-5/23/$31.00 ©2023 IEEE | DOI: 10.1109/CASE56687.2023.10260293

***Abstract*— Large offline learning-based models have enabled robots to successfully interact with objects for a wide variety of tasks. However, these models rely on fairly consistent structured environments. For more unstructured environments, an online learning component is necessary to gather and estimate information about objects in the environment in order to successfully interact with them. Unfortunately, online learning methods like Bayesian non-parametric models struggle with changes in the environment, which is often the desired outcome of interaction-based tasks. We propose using an object-centric representation for interactive online learning. This representation is generated by transforming the robot's actions into the object's coordinate frame. We demonstrate how switching to this task-relevant space improves our ability to reason with the training data collected online, enabling scalable online learning of robot-object interactions. We showcase our method by successfully navigating a manipulator arm through an environment with multiple unknown objects without violating interaction-based constraints.**

I. Introduction

Automated robot manipulation is rapidly being adopted throughout industry to improve efficiency and accuracy across several manufacturing tasks [1], [2]. For applications that require interaction with objects in the environment (e.g. assembling automobile components), current successful methods require highly structured and consistent environments, such as assembly lines. This structure enables established planning algorithms and offline learning-based models to work well. Unfortunately, these offline methods are usually parametric and have the drawback of being nearly impossible to update online as new data is observed. In many unstructured environments that are not perfectly modeled, this becomes a drawback, as there are many attributes of the environment that are difficult to learn without actively interacting with the environment.

Navigating and interacting with objects in less structured environments like warehouses, construction sites, or even a common household remains challenging. As an example, picture a household pantry with many opaque containers. Multiple parameters (e.g. center of mass, friction coefficients) are difficult to estimate without active interaction, and may drastically affect the results of the interaction. In these cases, it becomes important to have an online learning component that can help bridge this gap by learning through interaction.

Most work on online learning focuses on estimating particular model parameters or uncertainty pertinent to the robot model itself rather than interactions with objects. One popular approach is the use of Bayesian non-parametric models for online learning [3]. These can readily incorporate priors, these can be in the form of offline-learned base models, which enable predictions in low-data scenarios. They provide interpretable confidence metrics around their predictions in the form of a posterior distribution. These methods are data-driven and create Bayesian models on an effectively infinite-dimensional parameter space where the complexity of the model is allowed to grow with the size of the data [4]. In particular, in this work, we use a form of Bayesian modeling called Gaussian processes (GPs) to model the robot-object interaction attributes.

GPs are data-driven and rely on a new observation's similarity to the support set formed by its training data points. In interaction-based tasks, this can become challenging as the state of the objects and the robots, which form the datapoints, are constantly changing. The notion of finding the right data representation for a task has been very popular in offline learning methods [5]. Because the interaction between a robot and an object during manipulation tasks is largely

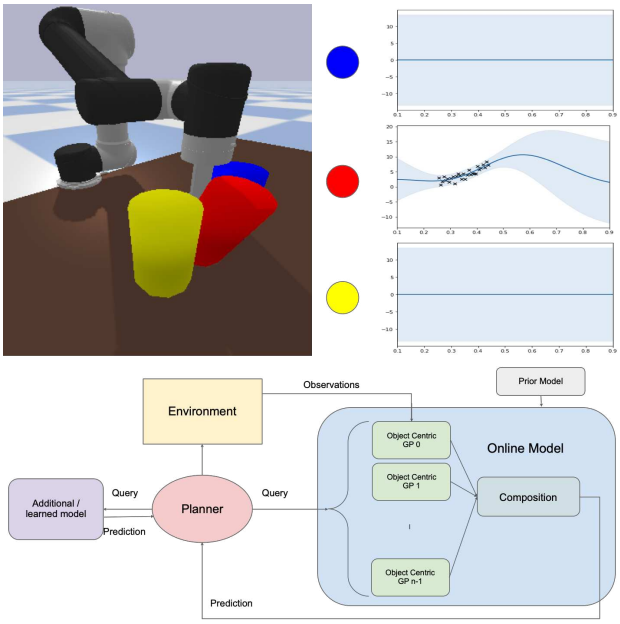

Fig. 1. Method Overview: Top: We apply our framework to the task of reaching a goal by pushing unknown objects on a table without knocking any over. The robot has no prior knowledge of how to interact with the objects, and reasons online using composable object-centric Gaussian process regression. A cartoon example of the robot's learned model is shown in the top figure. Bottom: The chart shows an overview of our method and how our online model would fit into a general planning framework.

This work was supported by NDSEG and NSF award #2045803. The authors are with the University of California, San Diego. {nshinde, jjj025, sherbert, yip}@ucsd.edu.

object-centric in nature, recent literature in offline learning for manipulation have used object-centric representations to improve automation [6], [5], [7].

Inspired by these recent developments, we apply object-centric representations to improve *online* learning for manipulation tasks. Our framework is shown in Fig. 1. We show that using an object-centric representation for online learning can be beneficial for capturing task-relevant features in our input representation and allow the model to learn better and be used online for the task at hand.

II. Related Works

Different sensor modalities using images [8], sound [9], and touch [10] have been proposed in the literature to capture object-centric properties like deformation, relative pose, mass, friction, and texture. But these sensor values are subject to noise and need to be actively tracked. Numerous works have shown the benefits of tracking these errors for downstream robotic tasks [11]–[13]. In [14], the authors use GPs to estimate object deformation. The model uses prior data to fit the GP model and cannot generalize to new objects with different material characteristics. In [15], the authors actively track deformations using GPs, but it is not generalizable to other modalities. An application of this work looks at how environment uncertainty can be reduced by moving objects occluding the sensors [16]. Still, it is not object-centric and doesn't consider how to interact with the environment. The authors in [17] track the noise in pose estimates using an ensemble of learning models, but these cannot capture object properties like friction and mass.

Recent works have looked at object representation, all specifically using neural networks. Zhu et al. [7] propose an object-centric learned representation using different camera views and proprioceptive data and uses the fused features to accomplish downstream manipulation tasks. Similarly, SORNet [5] uses a transformer-based architecture to generate latent embeddings for different objects that generalize to objects with similar shapes and textures. Still, since it uses images, it can't capture physical properties like friction and mass. Kofinas et al. [18] propose object representation that is rotation and translation invariant, which makes learning more efficient.

Most similar to our work with regards to learnable representations, the authors in [19] describe an object-centric embedding specifically for task and motion planning. The authors train an encoder-decoder structure by optimizing for task representations and pixel-wise segmentation of images. These models require large datasets for training, and the generated latent representations are difficult to interpret. In unstructured environments, we are data deprived, and data is costly to acquire, rendering these neural network representations unrealistic to train and inadapatable.

In summary, there continue to exist many challenging unstructured environments where interaction dynamics are unknown and must be estimated through online interaction. Most prior object representation work have focused on neural networks that are data-hungry to train and difficult to adapt online. On the other hand, methods that leverage online adaptation tend to be specialized towards tracking a singular or very small set of parameter errors in a given system model, with fewer considering the involved challenge of learning the potentially nonlinear, potentially stochastic, parameter function online. To address this gap, we described the paired use of (i) non-parametric methods that can learn, online, a statistical model of the measureable outcomes of interaction, and (ii) a task-relevant, object-centric representation that result in more scalable online learning.

III. Gaussian Process Regression

GPs are Bayesian non-parametric models that capture the distribution over continuous functions using a set of Gaussian random variables. The distribution over all functions $f : \mathbb{R}^m \rightarrow \mathbb{R}$ is parameterized with a mean function, $\mu_p(x)$, and a covariance or kernel function, $k(x, z)$, written as

$$f(x) \sim \mathcal{GP}(\mu_p(x), k(x, z)) \tag{1}$$

$$\mu_p(x) = \mathbb{E}[f(x)] \tag{2}$$

$$k(x, z) = \mathbb{E}[(f(x) - \mu_p(x))(f(z) - \mu_p(z))] \tag{3}$$

where $\mathbb{E}[\cdot]$ represents the expectation operation.

Given a set of training data inputs $X = \{x_0, x_1 \dots x_n\}, x_i \in \mathbb{R}^m$ and their corresponding noisy target values $Y = \{y_1, y_2, \dots, y_n\}$ where $y_i = f(x_i) + \boldsymbol{N}, \boldsymbol{N} \sim \mathcal{N}(0, \sigma_n^2)$, the posterior mean and variance for a new point x^* is given by:

$$f(x^*)|X, Y, x^* \sim \mathcal{N}(\mu^*, \sigma^*) \tag{4}$$

$$\mu^* = \mu_p(X) + K(X, x^*)^T K_y^{-1}(Y - \mu_p(X)) \tag{5}$$

$$\sigma^* = k(x^*, x^*) - K(x^*, X)K_y^{-1}K(X, x^*) \tag{6}$$

where $K_y = K + \sigma_n^2 I, K \in \mathbb{R}^{n \times n}$ and $K(X, x^*)^T = K(x^*, X) \in \mathbb{R}^n$. The matrix K is constructed by comparing all pairs of points in the given dataset using the kernel function i.e., $K(i, j) = k(x_i, x_j)$. Similarly, the vector $K(X, x^*)$ is constructed by comparing all values in X with x^*. The prior mean function, $\mu_p(x)$, can be set to 0 without loss of generality.

For this paper, we used the radial basis function (RBF) kernel function for k given by

$$k(x, z) = \alpha \exp\left(-\tfrac{1}{2} l^{-2}(x - z)^T(x - z)\right) \tag{7}$$

where scaling factor α and lengthscale l are hyperparameters that can be tuned. Any kernel function that belongs to the class of covariance functions can be chosen. An example of an alternate kernel is the forward kinematics kernel [20].

Due to the non-parametric nature of this method, this model is able to fit diverse data types. With enough data, the posterior distribution will overcome the model priors.

IV. Object-Centric Representations for Online Learning

We model the attributes of the interaction dynamics between the robot and the objects in the environment using GPs by leveraging data collected online, purely through interactions. We need interaction in unstructured environments

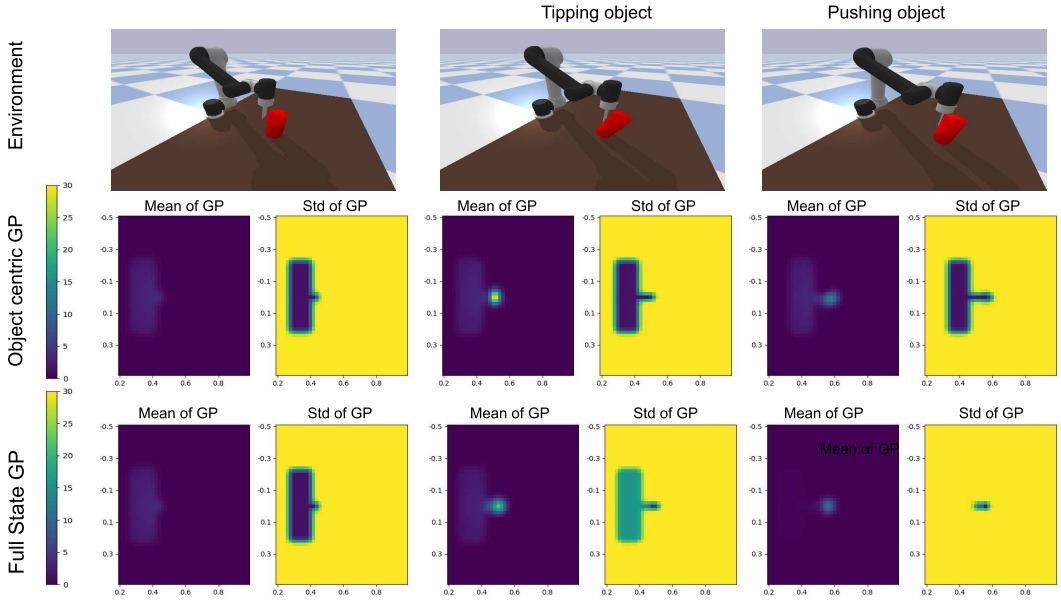

Fig. 2. Single object-centric GP: This figure demonstrates the benefit of using our object-centric GPs vs. a Full State GP baseline. In this experiment, we are probing a single object. We utilize the learned GP at different stages of interaction to predict the mean and variance of the tipping angle as a function of the robot state in the world frame and the current position of the object. We show this in two setups, one where the interaction largely causes the object to tip: "Tipping object" and one where the interaction largely causes the object to translate "Pushing object". Our object-centric representation allows our model to retain its predictive abilities despite changes in the object state, which cause the baseline GP quickly loses its ability to predict.

because the physical properties of these objects are not well-defined or may change with time. For example, consider an environment with 5 opaque bottles on a tabletop. The robot would like to reach a goal without knocking over these bottles. The properties of the bottles such as the center of mass and friction coefficients are challenging to model from visual sensors alone and are better understood by interactions. In our experiments, we model the tipping angle of each object because it encapsulates different physical parameters of the objects.

Consider a simple illustrative running example shown in Fig. 2 with one object, O_0, and the robot. The robot has an end effector which is primarily responsible for interactions with the object. For this paper, we consider fairly homogeneous environments that do not have a large variation in their terrain for the bounds of the environment. The object state is denoted by $O_0 : p_{O_0} \in \mathbb{R}^3, q_{O_0} \in SO(3)$. Here p_{O_0} and q_{O_0} denotes the position and orientation of O_0 respectively. The robot end effector state is denoted by $A : p_A \in \mathbb{R}^3, q_A \in SO(3)$. Here p_A and q_A denote the position and orientation of the robot. All orientations are represented using quaternions in this paper. These states are with respect to a world coordinate frame.

The interaction attribute that we model is denoted by y. This interaction attribute is a function of the object state in the world frame, the robot state in the world frame and the robot action in the world frame. For the purposes of this paper, we consider a quasi-static environment so that the interactions can be modeled using only the states without loss of generality. We can readily extend this to consider additional action parameters in our inputs and by transforming them with the same or similar object-centric representation. The true value of y can be computed using $f(O_0, A, \theta)$ where θ denotes an unknown number of independent parameters that specify the real system. In practice, we only get noisy observations of the interaction attribute y^t paired with noiseless observations of the object and robot state: O_0^t, A^t, y^t at any timepoint.

A naive implementation of the GP uses the object state and the robot state $\{O_0^t, A^t\}$, or some subset of each of the states, as an input and outputs y in attempt to model the true function f. GP regression is highly reliant on its similarity metric or kernel function and the observed data points in order to model behavior at a new unseen datapoint. As seen in Eq. 5, 6, it uses the similarity metric to compare the new input to the support set of inputs contained in its training dataset. In tasks involving interacting with and manipulating objects, the object states will change as a result of the robot motion. Changes in the object state will cause all future inputs to the GP to appear to be further from the support set of training points that may not contain the object state in question. As a result, when the object state is altered by the robot, the GP Regression model's predictive capabilities will falter, resulting in prediction values close to the GP's prior with a very high variance indicating a low measure of confidence. We can see this behavior in Fig. 2. As the robot interacts with the object and alters its state, the predictive variance on previously seen robot states increases drastically, and the predictive mean falls back to the GP prior.

Despite the change in the object state in the world frame, the model should still be able to rely on its support set of datapoints from past interactions and allow reasoning about new interactions in this updated object state. This is because these manipulation/interaction-based tasks are fairly object-centric in nature, as discussed in section II. The interaction between the robot and the object is a function of

the state/action of the robot in the relative frame of the object. Thus despite the object state changing, the model should be able to continue to reason about certain robot-object interactions. We leverage this understanding by learning the interaction attribute using an object-centric GP. The object-centric GP uses the state of the robot in the object frame $A^t_{O_0}$, or some subset of the state, as an input and outputs a prediction on the interaction attribute y.

To compute $A^t_{O_0}$ we start by using O_0 to create a transformation matrix $T_w^{O_0}$ to convert coordinates in world frame to coordinates in the frame of the object 0_0. To get $T_w^{O_0}$ we start with the pose O_0. We first compute the rotation matrix $R^W_{O_0}$ using the quaternion specifying the orientation of pose $O_0 : q_{O_0} = [q_0, q_1, q_2, q_3]$.

$$R^W_{O_0} = \begin{bmatrix} 1-2q_1^2-2q_2^2 & 2q_0q_1-2q_3q_2 & 2q_0q_2+2q_3q_1 \\ 2q_0q_1+2q_3q_2 & 1-2q_0^2-2q_2^2 & 2q_1q_2-2q_3q_0 \\ 2q_0q_2-2q_1q_3 & 2q_1q_2+2q_3q_0 & 1-2q_0^2-2q_1^2 \end{bmatrix} \quad (8)$$

This rotation matrix is used with the position of $O_0 : p_{O_0}$ to compute the desired transformation matrix $T_w^{O_0}$ at time t.

$$T^W_{O_0} = \begin{bmatrix} R^W_{O_0} & p_{O_0} \\ \vec{0} & 1 \end{bmatrix}, \vec{0} = [0,0,0] \quad (9)$$

$$T_w^{O_0} = [T^W_{O_0}]^{-1} \quad (10)$$

This transformation matrix is utilized to transform coordinates, such as the state of the robot, into the object frame. $A^t_{O_0} : p_{A^t_{O_0}} \in \mathbb{R}^3, q_{A^t_{O_0}} \in SO(3)$. $R^W_{A^t}$ is the rotation matrix corresponding to the orientation q_{A^t}, computed similar to 8.

$$\begin{bmatrix} R^W_{A^t_{O_0}} & p_{A^t_{O_0}} \\ \vec{0} & 1 \end{bmatrix} = T_w^{O_0} \begin{bmatrix} R^W_{O_0} & p_{A^t} \\ \vec{0} & 1 \end{bmatrix}, \vec{0} = [0,0,0] \quad (11)$$

$R^W_{A^t_{O_0}}$ is the rotation matrix describing the orientation of $A^t_{O_0}$ in the frame of O_0, which can be converted back to a quaternion $q_{A^t_{O_0}}$. These transformed representations are used with the object-centric GP. The transformation can be augmented or simplified by leveraging geometric symmetries of the object or other task-specific simplifications.

This object-centric representation consolidates the object and robot state and switches the input space of the GP to be more task-relevant, allowing better online modeling. Since the interaction attribute should only be a function of the robot state relative to the object state this representation provides a better space to compare new datapoints to the support set of datapoints collected online. Thus even if the object state has moved in the world frame, the model can leverage previous datapoints and reason about new robot states. By consolidating the two state spaces, we also reduce the dimensionality of the input space of the GP. We can see this behavior in Fig. 2. As the robot interacts with the object and alters its state we still maintain a high confidence over previous states that have been sampled.

In the case of multiple objects $\{0, 1, \ldots n-1\}$ we can model the pairwise interactions between the robot and each object using a separate object-centric GP for each object. This can be done when the robot accounts for a majority of what is measured in the interaction attribute. Each of these GPs can be updated intelligently, based on the proximity of the robot, to improve efficiency. The outputs of these GPs can be combined coherently, based on the task. This combined output can be used by a high-level planner to make intelligent decisions on how to move through the environment and interact with objects. One example of how multiple object-centric GPs can be composed is illustrated in section V.

V. Experiments and Results

To showcase our method, we consider a problem with multiple objects on a tabletop with different attributes such as mass and center of mass. The interaction dynamics of the objects are unknown and estimated through online interaction. We consider a task involving non-prehensile manipulation where the robot must push the objects to reach a specified goal region. The robot wants to get to the goal without tipping the objects beyond a certain angle and knocking them over. We use GPs to directly learn the interaction by mapping the object-centric robot state to the tipping angle of the object, for each object. For these experiments the robot is constrained to motion in the x, y plane at a fixed height, thus we only visualize the GP over robot states in the x, y plane. We initialize our GPs with samples in the empty space around the robot for all experiments.

In Fig. 2 we demonstrate what our model learns when interacting with a single object. To do this, we run our robot in open-loop to probe a single object. We add the samples gathered during this open loop maneuver to update our GP model online. Each sample datapoint consists of the robot end effector state in the object-centric frame and the observed tip angle of the object. To visualize the GP at a point in time we plot the mean and standard deviation (std) of the predicted tipping angle for robot end effector positions in the world coordinate frame for the current object state. We compare this to the "Full State GP" baseline. For this GP a sample datapoint's input contains the robot and object state in the world frame, with the corresponding object's tip angle as the output.

In Fig. 2, we show the results of probing two different types of objects: A "tipping object" whose interaction parameters cause it to mainly tip on contact, and a "pushing object" whose parameters allow it to be pushed by the robot. As the object state changes, the full state GP fails to properly leverage support points from its training dataset to make valuable predictions about the tipping angle beyond the immediate position of the robot. This occurs when the object state changes slightly due to tipping and is much more exaggerated when the object is pushed. In contrast, our object-centric representation enables the model to continue to make meaningful predictions by leveraging its support set in spite of changes in the object state.

Multiple object open loop experiment: We repeat the same experiment to show this methodology scaling to multiple objects, as shown in Fig. 3 The Full State GP takes the state of the robot and objects in the world frame as its input and maps it to the maximum tip angle in the environment. With the object-centric, approach we split the

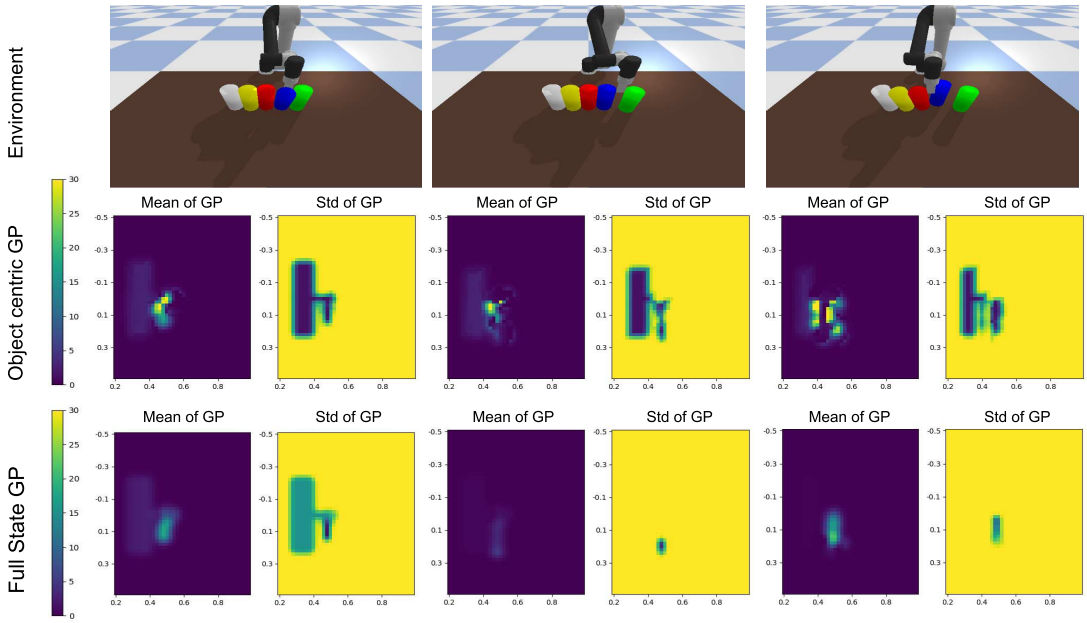

Fig. 3. Multiple object-centric GPs: This figure demonstrates the benefit of using a composition of object-centric GPs to model robot-object interactions in environments with multiple objects. The robot runs an open loop path and interacts with multiple objects. We show frames of the robot as well as the mean and standard deviation of the tipping angle predicted by the online learned GP models as a function of the robot state in the world frame and the object states at that time point. Even in complex environments with multiple objects, our object-centric representation allows us to maintain a good model of the robot object interactions, while the full state GP fails once the objects have been interacted with.

problem up to consider pairwise interactions between the robot and each object separately. We learn each interaction online with a separate object-centric GP. These GPs can then be used to predict pairwise interactions, and their outputs can be combined for the task at hand.

This scenario focuses on predicting the worst-case tipping angle in the environment in terms of the probabilistic upper bound of our understanding. We first predict the mean and variance of the tipping angle with each of our GPs: $\{(\mu_0, \sigma_0), (\mu_1, \sigma_1), \ldots, (\mu_{n-1}, \sigma_{n-1})\}$. The upper bound $u(\mu, \sigma) = \mu + \beta\sigma$ is then computed for each output: $\{u_0, \ldots u_{n-1}\}$. The mean and variance prediction corresponding to the highest upper bound: $(\mu_i, \sigma_i), i = \arg\max_i u_i$, is then used. From Fig. 3 we can see that the object-centric representations enable useful predictions, in stark contrast to the full state, even after multiple objects have begun to move from their original states.

Object-centric model integration with planner: Our online learned models can be integrated with planners, as shown in Fig. 4. The environment contains 5 objects with unknown centers of mass, mass, and completely unknown interaction dynamics. The robot is attempting to get to a goal region at the other side of the table without knocking over the objects. The robot state space is bounded so that it can not trivially go around the objects. If the robot attempts to naively push through the objects, some objects will tip excessively and fall over. To showcase our models, the planner lacks any prior on how the objects will move in response to interactions. Additionally, to showcase the learning capabilities of our GPs, they are initialized with a naive prior mean function, $\mu_p(x) = 0$, that indicates that the robot can move without affecting the objects. The planner uses the online learned models to ensure that the robot will not cause the bottles to fall over. The predicted mean and variance of the tipping angles are used, by the planner, to determine where the robot should sample to learn more about the environment, while balancing exploration and exploitation to get to the goal. Using our algorithm the robot is able to successfully learn about the environment through multiple sampling maneuvers. The robot is able to leverage small gaps created between the objects as they are manipulated to squeeze between the objects in a manner that doesn't tip them over beyond a set threshold, to get to the goal region. The sampling maneuvers can be seen between timesteps $[0.04\alpha, 0.9\alpha]$ in Fig. 4, before the robot exploits the gap between the bottles between timesteps $[0.9\alpha, 0.94\alpha]$

VI. Conclusion

We enable better online learning for robot-object interaction tasks in unknown environments through our use of task-relevant object-centric representations. We showcase the potential of our method by integrating it with a planner to navigate through complex obstacle-filled environments.

This work's restriction to learning online limited our choice of predictive models for learning. Though GPs provide an efficient, non-parametric way to learn online they have limitations. These limitations include cubic computational complexity with increased samples, difficulty predicting in high dimensional spaces and difficulty in choosing hyperparameters. The drawbacks posed by these limitations can often be mitigated. Sparse GPs, local GPs or techniques such as thresholding the number of used predictive samples can help reduce predictive computational complexity. Creating an efficient kernel to leverage geometric similarities in

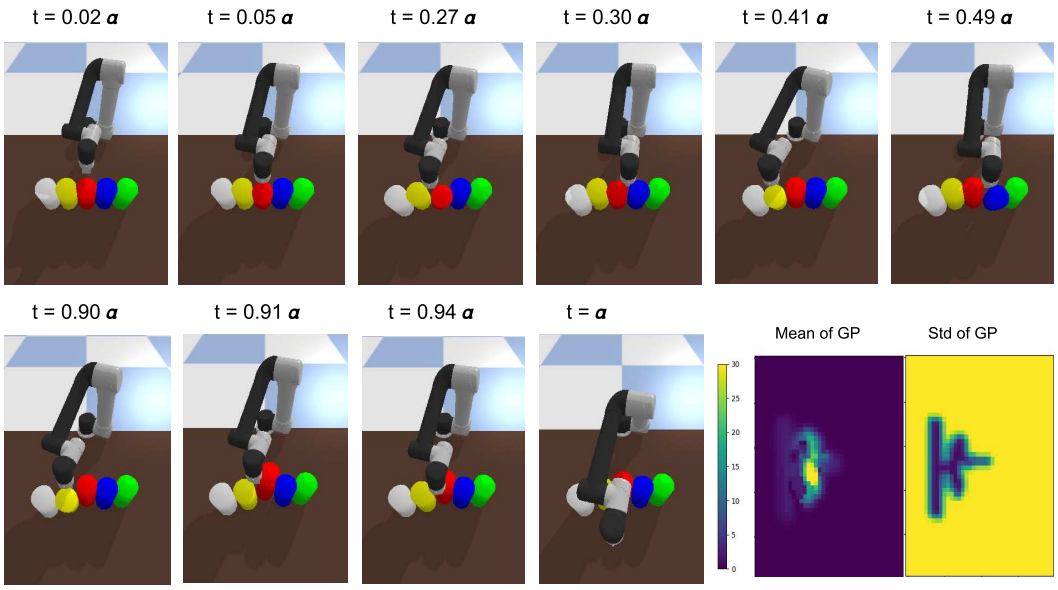

Fig. 4. Online learning with object-centric GPs for Planning: The robot is trying to get to a goal on the other side of several objects with unknown interaction dynamics without knocking any of them over. The robot uses a composition of object-centric GPs to learn about its effect on the objects through online interaction. The robot's planner queries the GP models and uses their predictions and confidence bounds to balance exploring the environment and exploiting what it has learned to get to the goal, the other side of the table. We show the plan at different frames between timesteps $[0, \alpha]$. We also show the mean and variance predictions of the worst-case tipping angle generated with our method at the final timestep. This prediction is done over the robot states in the world frame with respect to the object states at the final timestep.

the scene or focus on more relevant features can help extend this method to more complex high dimensional spaces. While certain hyperparameters can be set by maximizing the likelihood of the observed data, setting these parameters can also serve as a way to enforce principled priors on extending the learned model to uncertain, unsampled regions of the state space.

This work provides an initial step in interactive online learning to improve interaction-based tasks in unknown environments. Such a object-centric representation can be used to incorporate richer state information that can ultimately lead to more intelligent interactions with deformable objects, for tasks such as tissue manipulation for surgical automation.

REFERENCES

[1] T. Brogårdh, "Present and future robot control development—an industrial perspective," *Annual Reviews in Control*, vol. 31, no. 1, pp. 69–79, 2007.

[2] A. K. Gupta and S. K. Arora, *Industrial automation and robotics*. Laxmi publications, 2009.

[3] P. Orbanz and Y. W. Teh, *Bayesian Nonparametric Models*. Boston, MA: Springer US, 2010, pp. 81–89.

[4] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[5] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "SORNet: Spatial object-centric representations for sequential manipulation," in *5th Annual Conference on Robot Learning*, 2021.

[6] O. Kroemer, S. Niekum, and G. D. Konidaris, "A review of robot learning for manipulation: Challenges, representations, and algorithms," *CoRR*, vol. abs/1907.03146, 2019.

[7] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Imitation learning for vision-based manipulation with object proposal priors," *6th Annual Conference on Robot Learning (CoRL)*, 2022.

[8] Z.-Y. Chiu, A. Z. Liao, F. Richter, B. Johnson, and M. C. Yip, "Markerless suture needle 6d pose tracking with robust uncertainty estimation for autonomous minimally invasive robotic surgery," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 5286–5292.

[9] M. S. Li, T. M. Huh, C. R. Yahnker, and H. S. Stuart, "Resonant pneumatic tactile sensing for soft grippers," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 105–10 111, 2022.

[10] T. Hellebrekers, O. Kroemer, and C. Majidi, "Soft magnetic skin for continuous deformation sensing," *Advanced Intelligent Systems*, vol. 1, no. 4, p. 1900025, 2019.

[11] J. J. Johnson and M. C. Yip, "Chance-constrained motion planning using modeled distance- to-collision functions," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 1582–1589.

[12] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017, pp. 1517–1522.

[13] J. van den Berg, P. Abbeel, and K. Goldberg, "Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 895–913, 2011.

[14] B. Frank, C. Stachniss, N. Abdo, and W. Burgard, "Using gaussian process regression for efficient motion planning in environments with deformable objects," ser. AAAIWS'11-09. AAAI Press, 2011, p. 2–7.

[15] S. Caccamo, P. Güler, H. Kjellström, and D. Kragic, "Active perception and modeling of deformable surfaces using gaussian processes and position-based dynamics," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 530–537.

[16] S. Caccamo, Y. Bekiroglu, C. H. Ek, and D. Kragic, "Active exploration using gaussian random fields and gaussian process implicit surfaces," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 582–589.

[17] G. Shi, Y. Zhu, J. Tremblay, S. Birchfield, F. Ramos, A. Anandkumar, and Y. Zhu, "Fast uncertainty quantification for deep object pose estimation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 5200–5207.

[18] M. Kofinas, N. S. Nagaraja, and E. Gavves, "Roto-translated local coordinate frames for interacting dynamical systems," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.

[19] C. Wang, D. Xu, and L. Fei-Fei, "Generalizable task planning through representation pretraining," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8299–8306, 2022.

[20] N. Das and M. C. Yip, "Forward kinematics kernel for improved proxy collision checking," *IEEE Robot. and Automat. Lett.*, vol. 5, no. 2, pp. 2349–2356, 2020.