IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Estimating Deformable-Rigid Contact Interactions for a Deformable Tool via Learning and Model-Based Optimization

Mark Van der Merwe, Miquel Oller, Dmitry Berenson, and Nima Fazeli[1]

***Abstract*—Dexterous manipulation requires careful reasoning over extrinsic contacts. The prevalence of deforming tools in human environments, the use of deformable sensors, and the increasing number of soft robots yields a need for approaches that enable dexterous manipulation through contact reasoning where not all contacts are well characterized by classical rigid body contact models. Here, we consider the case of a deforming tool dexterously manipulating a rigid object. We propose a hybrid learning and first-principles approach to the modeling of simultaneous motion and force transfer of tools and objects. The learned module is responsible for jointly estimating the rigid object's motion and the deformable tool's imparted contact forces. We then propose a Contact Quadratic Program to recover forces between the environment and object subject to quasi-static equilibrium and Coulomb friction. The results is a system capable of modeling both intrinsic and extrinsic motions, contacts, and forces during dexterous deformable manipulation. We train our method in simulation and show that our method outperforms baselines under varying block geometries and physical properties, during pushing and pivoting manipulations, and demonstrate transfer to real world interactions. Video results can be found at https://deform-rigid-contact.github.io/.**

***Index Terms*—Deep Learning in Grasping and Manipulation, Perception for Grasping and Manipulation, Contact Modeling**

I. Introduction

REASONING over the contact between a tool or end-effector and a target object is crucial to enabling performant, safe, and autonomous robotic manipulation. This extends naturally to the case where the contacts are not all well represented by rigid-body contact models. Deformable tools and objects are commonly found in human environments [1] and the inherent compliance may be advantageous for both maintaining contact and preventing excessive forces while interacting with, securing and controlling objects. Additionally, the advent of deformable tactile sensors [2], [3], [4] and soft robot manipulators [5] introduce new cases of non-rigid contact into robotic manipulation.

In this work, we consider an elastically deformable tool manipulating a rigid object, supported by the environment. This scenario could arise when utilizing deforming tools such as spatula or sponges [6] or manipulating objects with a soft robot [7] or end-effector [8]. Manipulation of such a system presents two important challenges: First, the deformation of the tool and motion of the rigid object are inherently linked - solving for one requires reasoning about both. However, modeling the deformation of the tool is challenging, due to the high-dimensional and non-linear dynamics [9]. Second, the system is only partially observable, and we must rely on sensing to intuit deformations and contact forces.

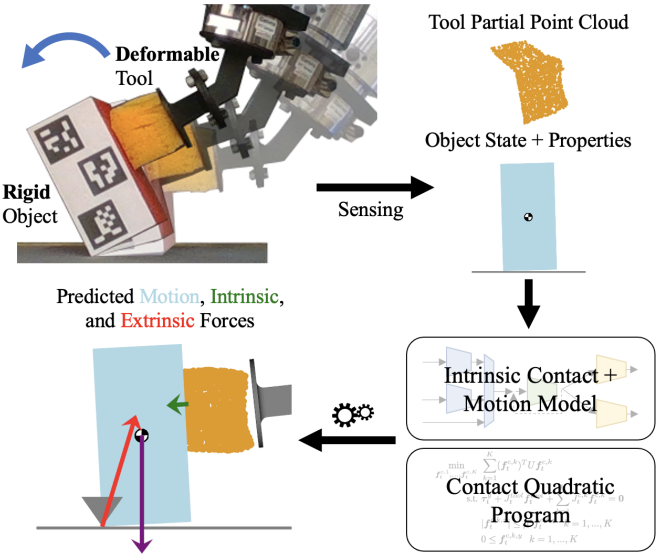


Fig. 1: We present a method that estimates motions and forces during dexterous manipulation with a deformable tool. Our proposed method takes in object information (geometry, center of mass, mass, and friction) and sensing from the deforming tool (partial point cloud). It estimates the block motion, tool force the deforming tool enacts on the block, and the rigid contact forces between block and (known) environment, given the robot actions. On the bottom left, we show one-step predictions from our method for a real pivoting execution.

To address these challenges, we propose a hybrid learning and first principles method for modeling the motion, contacts, and forces on the tool and extrinsic object during manipulation. Accurate motion models are crucial for driving the object toward goal configurations while recovering forces can help enforce desired contact modes and prevent excessive force transfer.

Our method leverages learning to address the complexity of the paired object motion and deformable tool contacts. It then turns to first-principles and physical priors to recover the extrinsic contacts and forces, enforcing quasi-static motion and Coulomb friction. This allows our method to overcome modeling challenges for the deformable object, while exploiting physical priors for modeling efficiency. We train our method on interactions between a tool and a variety of object geometries and physical properties using simulation [10]. We benchmark our method against baselines for modeling block

This work was supported in part by the Office of Naval Research Grant N00014-24-1-2036 and NSF grants IIS-2113401, IIS-2337870, and IIS-2220876.

[1]Robotics Department, University of Michigan, Ann Arbor, MI, USA 48109 {markvdm, oller, dmitryb, nfz}@umich.edu

motion and for recovering the environment-object forces. Finally, we deploy our model on a real robot, demonstrating sim-to-real transfer. In summary, our contributions are:

- a learning architecture for jointly resolving object motions and tool contacts and forces, given partial sensing and object information.
- a first-principles Contact Quadratic Program (CQP) that resolves environment contacts and forces, subject to quasi-static equilibrium and Coloumb friction.

II. Related Work

Enabling extrinsic dexterity of freely moving rigid bodies has long been a subject of research, from pushing [11] to pivoting [12] to chaining of multiple manipulating primitives [13]. Existing work on extrinsic dexterity largely assumes rigid-to-rigid interactions [12], [14], [15] and/or assumes sensing at the contact interface [13], [16]. We instead investigate deformable-to-rigid contact, where the robot directly controls the deformable tool and indirectly manipulates the rigid body.

Much research forgoes model-based reasoning in favor of learning rigid-body motion from data. To predict the motion of a rigid body, existing work has investigated directly predicting SE3 transforms of objects [17], [18] or predicting point cloud or mesh vertex motion [19], [20]. Rigid body contact can yield discontinuities that are challenging for learned models, which generally favor smooth approximations. As such, Pfrommer et al. [21] propose to learn rigid body motion by parameterizing inter-body signed distances and contact Jacobians, enabling analytical physical simulation that can reflect rigid contacts. Our work, in-contrast, decomposes the contacts on the extrinsic rigid object to the deformable-to-rigid contact, which we model by learning to predict contact information, and the rigid-to-rigid contact between the object and environment. We note that our system does assume quasi-static motion, unlike in [21]. Calandra et al. [22] recovers contact forces and incorporates them into dynamics for a robot arm; we recover contact forces and incorporate them into quasi-static equilibrium for a rigid body. Other work seeks to directly predict the motion of heterogenous materials by applying Graph Neural Networks [23]. In contrast, we avoid directly modeling the motion of the deforming tool and only focus on the contact forces and rigid body motion.

Using deformation as an indicator for contact and force information has been demonstrated in continuum robots [24], [25] and volumetric deformables [9], [26]. In this work, we extend the modeling focus to also include the extrinsic object being manipulated.

III. Problem Formulation

Our goal is to reason over a) the motion of the extrinsic rigid object, and b) all the forces acting on it. We consider the case where the motion of the robot and rigid object are in the plane of gravity. We assume the following information from each part of the system:

1) **Object and Support Surface:** We assume knowledge of the mass m^e, the center of mass $\boldsymbol{p}^{e,CoM}$, the friction between object and rigid supporting surface μ^e, the geometry of the object, provided as a mesh $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, and the geometry of the support surface. Furthermore, we can track the motion of the rigid object to receive the current object pose $\boldsymbol{q}_t^e \in SE(2)$. By our assumed knowledge of the environment geometry, we additionally can use the known object pose and geometry to recover the current contact locations $\boldsymbol{p}_t^{c,1}, ..., \boldsymbol{p}_t^{c,K}$.
2) **Tool:** We assume access to a *segmented* partial pointcloud of the deformable tool $\boldsymbol{P}^{tool}$, provided by our sensors. Recent developments in category-agnostic open world segmentation [27] make recovering segmentation masks possible on the real system.

Given these inputs and assumptions, and the actions to be taken $\boldsymbol{a}_{t:t+H}$ for some horizon H, where each action is a planar translation and rotation, we wish to recover the future poses of the object $\boldsymbol{q}_{t:t+H}^e$ and all forces acting on the object. We split our consideration of the forces into those from the deforming tool and those from the supporting surface. Contact interface representations have take many forms, including points [28], lines [29], [30], [6] and patches [31], [26]. Here, the most important feature of the contact interaction is not its particular geometry, but rather its *effect*, namely, the resulting force on the block. As such, we consider a simple contact representation: a summary contact point $\boldsymbol{p}_t^{tool}$ and force $\boldsymbol{f}_t^{tool}$, whose resulting wrench on the block is equivalent to the actual and potentially extended contact. As many points and forces can yield an equivalent wrench on the object, we use the centroid of the contact between the tool and object as $\boldsymbol{p}_t^{tool}$, which prevents ambiguity in the representation. Finally, we wish to also recover the set of contact points $\boldsymbol{p}_t^{c,1}, ..., \boldsymbol{p}_t^{c,K}$ and forces $\boldsymbol{f}_t^{c,1}, ..., \boldsymbol{f}_t^{c,K}$ between the rigid object and the environment. We assume access to a labeled dataset with the specified inputs and outputs listed above, which we utilize to train our method and the baselines.

IV. Method

Our proposed method has two main components - a learned module for predicting a) the object motion and b) the contact location and forces between the deforming tool and object, and a model-based optimization problem for recovering the resulting frictional contacts with the environment. An overview of our approach can be found in Fig. 2.

A. Jointly Learning Object Motion and Deformable-Rigid Contact Interactions

The underlying mechanics of deforming objects is complex and can be expensive to compute [1], [10]. Simplifying models [32] can recover forces but lack fidelity to deforming surfaces. Learning-based methods have modeled geometries [9], [33] and contact interfaces [26] and crucially can recover information when only provided with partial information sensing, such as partial pointclouds, but have not recovered forces at the contacting surfaces. We adopt a learning-based approach to be able to recover the complex contacts and motions directly from partial sensing, and directly predict the resulting forces.

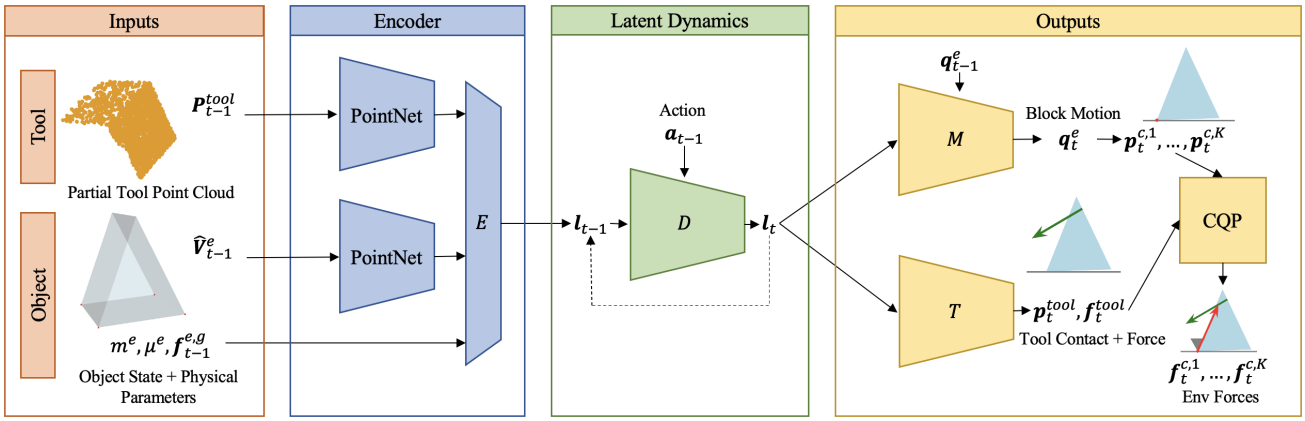


Fig. 2: Overview of our proposed method. Our method takes in partial views of the tool via point clouds along with the current object state and parameters and encodes to a learned latent space. Our latent dynamics module then rolls out in the latent space given actions. Finally, our model regresses a) object motion and b) tool summary contact point and force. We use the object motion to determine environment contacts and solve a Contact Quadratic Program (CQP) of our design which resolves the environment contact forces subject to friction and quasi-static equilibrium.

1) Architecture: An overview of our proposed architecture is shown in Fig. 2. Our architecture design encodes multimodal information from the object and the tool into a learned latent space. A latent dynamics backbone then propagates the latent information provided with the actions. Finally, we have multiple output heads that decode the object motion and tool contact information from the latent. We detail the architecture below.

Network Inputs: In order to accurately reason about the interactions, we input multimodal information from the rigid object and the tool. We represent spatial quantities (poses, forces, points, etc.) in the robot end effector frame provided by $\boldsymbol{q}_t^r$. We first encode the partial pointcloud of the tool $\boldsymbol{P}_t^{tool}$ using a PointNet encoder [34]. We encode the geometry and current pose by extracting the mesh vertices V and transforming them to the tracked object location $\boldsymbol{q}_t^e$. We additionally use our knowledge of the environment to detect which vertices are contacting the surrounding environment and append these binary contact indicators to each vertex location. We use another PointNet encoder [34] to encode the resulting unstructured point set $\hat{\boldsymbol{V}}_t^e \in \mathbb{R}^{|V|\times 4}$. Note, that this also allows us to use a single encoder for objects with varying geometries and differing numbers of vertices. We fuse the resulting latent embeddings from the tool and rigid object, along with the object mass m^e, object friction μ^e, and the gravity vector $\boldsymbol{f}_t^{e,g}$ expressed in the robot frame. This allows the model to reason about gravitational effects while keeping all learning in the robot frame. These inputs are fused and passed through a final multi-layer perceptron (MLP) to yield the latent code $\boldsymbol{l}_t$. We call this encoder E.

Latent Dynamics: Given the input actions $\boldsymbol{a}_{t:t+T}$, we rollout the dynamics in the latent space [35] as the latent space can capture how the resulting deformations, motions, and forces evolve. Our latent dynamics model D is given the current latent state and action and predicts the future latent state.

$$\boldsymbol{l}_{t+1} = D(\boldsymbol{l}_t, \boldsymbol{a}_t) \tag{1}$$

This allows us to recursively predict successive states given action sequences.

Output Heads: The final learned components are two output networks which take the latent state at a given time step and estimate the motion. One MLP M is used to estimate the motion of the object, given the current latent state. We predict the motion as a delta motion, utilizing the axis-angle representation [17], [18].[1]

$$\Delta \boldsymbol{q}_t^e = M(\boldsymbol{l}_t) \tag{2}$$

These delta poses can be combined with the tracked initial state $\boldsymbol{q}_t^e$ to predict the future rollout states of the rigid object.

A final MLP T is used to directly regress the tool contact point $\boldsymbol{p}_t^{tool}$ and force $\boldsymbol{f}_t^{tool}$.

$$\boldsymbol{p}_t^{tool}, \boldsymbol{f}_t^{tool} = T(\boldsymbol{l}_t) \tag{3}$$

2) Training Losses: We assume access to a labeled dataset $\mathcal{D}$ which contains example rollouts of the system with the necessary observations, actions, and labels provided. We train the model performance over a specified horizon H. We encode the starting observations using our encoder to yield our starting latent $\boldsymbol{l}_t = E(\boldsymbol{P}_t^{tool}, \hat{\boldsymbol{V}}_t^e, m^e, \mu^e, \boldsymbol{f}_t^{e,g})$ and recursively apply our dynamics D to yield the sequence of predicted latent vectors $\boldsymbol{l}_{t:t+T}$. We then apply our models M and T to recover our estimated object delta poses $\Delta \boldsymbol{q}_{t:t+H}^e$, tool contact points $\boldsymbol{p}_{t:t+H}^{tool}$, and tool contact forces $\boldsymbol{f}_{t:t+H}^{tool}$. We then apply the following supervised loss.

$$\mathcal{L}_{pred} = \sum_{i=t}^{t+H} ||\Delta \boldsymbol{q}_i^{e*} - \Delta \boldsymbol{q}_i^e||_2^2 + \alpha ||\boldsymbol{p}_i^{tool^*} - \boldsymbol{p}_i^{tool}||_2^2 + \tag{4}$$

$$\beta ||\boldsymbol{f}_i^{tool^*} - \boldsymbol{f}_i^{tool}||_2^2 \tag{5}$$

Where ground truth values come from our dataset $\mathcal{D}$ and α and β are loss weighting terms.

[1]Our model can handle full SE(3) poses and actions but all examples considered are planar motions.

To encourage accurate latent dynamics, we use future observations to encode ground truth latent rollouts. That is we derive $\boldsymbol{l}^*_{t:t+H}$ by applying E on the actual observations found when rolling out the system. We then encourage our recursively generated latents to match these values. We additionally regularize the scale of the predicted latent vectors.

$$\mathcal{L}_{latent} = \sum_{i=t}^{t+H} \gamma ||\boldsymbol{l}^*_t - \boldsymbol{l}_t||_2^2 + \zeta ||\boldsymbol{l}_t||_2 \tag{6}$$

B. Model-Based Estimation of Frictional Object-Environment Contact

We next turn our attention to estimating the environment contacts. As outlined in Sec. III, we can recover the set of K active contact points at a given time step $\boldsymbol{p}_t^{c,1}, ..., \boldsymbol{p}_t^{c,K}$ given our estimated object poses and the known object and environment geometries. The task is to find the contact forces at those points $\boldsymbol{f}_t^{c,1}, ..., \boldsymbol{f}_t^{c,K}$. To estimate these forces, we exploit our knowledge of Coloumb friction and quasi-static motion to formulate an optimization problem that recovers a set of valid contact forces.

Assuming the system is in quasi-static equilibrium, and given the contact force applied by the tool $\boldsymbol{f}_t^{tool}$, we know that the set of environment contact forces must be the solution to the following Quadratic Program (QP) optimization problem [15].

$$\begin{aligned} \min_{\boldsymbol{f}_t^{c,1},...,\boldsymbol{f}_t^{c,K}} \quad & \sum_{k=1}^{K} (\boldsymbol{f}_t^{c,k})^T U \boldsymbol{f}_t^{c,k} \\ \text{s.t. } & \boldsymbol{\tau}_t^g + J_t^{tool} \boldsymbol{f}_t^{tool} + \sum_{k=1}^{K} J_t^{c,k} \boldsymbol{f}_t^{c,k} = \boldsymbol{0} \\ & |\boldsymbol{f}_t^{c,k,x}| \leq \mu^e \boldsymbol{f}_t^{c,k,y} \quad k = 1, ..., K \\ & 0 \leq \boldsymbol{f}_t^{c,k,y} \quad k = 1, ..., K \end{aligned} \tag{7}$$

Here $\boldsymbol{\tau}_t^g$ are the gravitational terms, J_t^{tool} and $J_t^{c,k}$ are the contact Jacobians mapping contacts from the contact frame to generalized forces on the object, and we use x and y superscripts to indicate the tangential and normal force components in the contact frame, respectively. The first constraint expresses the quasi-static equilibrium of the rigid object. The second and third constraint enforces Coloumb friction, namely that forces must lie within the friction cone defined by μ^e and that the forces can only apply pushing forces.

Were the system truly quasi-static and our tool force prediction perfectly accurate, we could utilize Eq. 7 to resolve our environment contact forces. In practice however, error in the model predictions, as well as our training data originating from a system that is not truly quasi-static can make Eq. 7 infeasible. To resolve this without sacrificing accuracy, we propose a relaxation of the QP.

TABLE I: Object Parameter Variations

Parameter	Rectangle	Triangle	Pentagon
Geometry	$w \in [0.05, 0.1]$ $h \in [0.08, 0.15]$	$w \in [0.05, 0.1]$ $h \in [0.08, 0.15]$	$l \in [0.05, 0.1]$
Mass	$m \in [0.3, 0.43]$ kg		
Friction	$\mu^e \in \mathcal{B}([0.2, 0.8], \alpha = 2, \beta = 5)$		

$$\begin{aligned} \min_{\hat{\boldsymbol{f}}_t^{tool}, \boldsymbol{f}_t^{c,1},...,\boldsymbol{f}_t^{c,K}} \quad & (\Delta \boldsymbol{f}_t^{tool})^T U (\Delta \boldsymbol{f}_t^{tool}) + \rho \sum_{k=1}^{K} (\boldsymbol{f}_t^{c,k})^T U \boldsymbol{f}_t^{c,k} \\ \text{s.t. } & \boldsymbol{\tau}_t^g + J_t^{tool} \hat{\boldsymbol{f}}_t^{tool} + \sum_{k=1}^{K} J_t^{c,k} \boldsymbol{f}_t^{c,k} = \boldsymbol{0} \\ & |\boldsymbol{f}_t^{c,k,x}| \leq \mu^e \boldsymbol{f}_t^{c,k,y} \quad k = 1, ..., K \\ & 0 \leq \boldsymbol{f}_t^{c,k,y} \quad k = 1, ..., K \end{aligned} \tag{8}$$

Here, $\Delta \boldsymbol{f}_t^{tool} = \boldsymbol{f}_t^{tool} - \hat{\boldsymbol{f}}_t^{tool}$ In our relaxation, we introduce a new decision variable $\hat{\boldsymbol{f}}_t^{tool}$ which we use to achieve force balance. We introduce a cost to minimize the difference between the new decision variable and our original estimated tool force $\boldsymbol{f}_t^{tool}$. We interpret the resulting $\hat{\boldsymbol{f}}_t^{tool}$ as the closest tool force to the estimated tool force, for which the system is in equilibrium. We utilize the resulting $\boldsymbol{f}_t^{c,1}, ..., \boldsymbol{f}_t^{c,K}$ as our estimate for the environment forces. We set ρ to a small value (1e-3) to prevent excessive drift from the estimated force while still realizing realistic contact forces.

V. Implementation

A. Data Collection

As described in Sec. IV-A2, we train our learned model using supervised learning. This requires us to generate ground truth labels of the contact force. As such, we use simulation to generate example interactions and train our model. We use the Drake simulator [36], as it supports deformable-rigid contacts [10]. We attach a simple deformable tool to the end of a gripper. We use a 46mm cube to match our real tool (Fig. 1) and utilize Drake's Finite Element Method simulation, setting Youngs Modulus to 1.1e4, Poisson's ratio to 0.1, and density to $30kg/m^3$. For the extrinsic object, we use three object primitives for which we generate variations: a rectangle, triangle and pentagon. All shapes are given a depth of 9cm. For each class, we perform geometric variations as well as varying the mass and friction parameters. The geometry and physical parameter variations used are shown in Table I.

We use a simple scripted policy to collect the data in simulation. We randomize the starting location of the wrist so that initial contact with the object is randomized along the contacting edges. We then perform a simple forward push or a pivot. In the push case, we command a translation along the supporting plane. In the pivot case, we press into the object, then pivot by following an arc around a selected pivot vertex on the rigid object. In both cases, we then add noise to the action to add more variation to the motions. We execute these policies and record the resulting object motion, contact forces

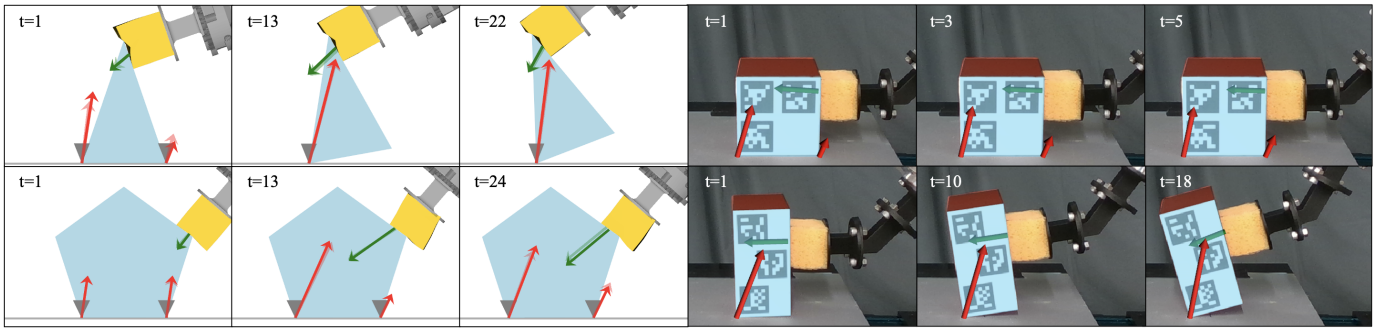


Fig. 3: Qualitative Results. **Left:** We compare our methods one-step predictions (solid) to the ground truth (semi-transparent) object motion, tool force, and environment forces across several manipulation trajectories. The friction cones for each environment contact is shown in gray at the environment point of contact. Our method shows high-fidelity predictions for a) varying object geometries and physical parameters, and b) different interaction primitives (pivoting vs. pushing). **Right:** We show one-step predictions for real robot executions. The predictions match the observed motion and show qualitatively realistic forces (see Sec. VI-B for quantitative results).

TABLE II: Real World 1-Step Object and Force Error (↓)

Metric		Pos. Error (mm)	Rot. Error (deg)	Force Error (N)
Blk. 1	Pivot	0.364 (0.128)	0.262 (0.125)	0.745 (0.292)
	Push	1.163 (0.420)	1.144 (0.613)	0.977 (0.445)
Blk. 2	Pivot	0.466 (0.196)	0.071 (0.054)	1.033 (0.382)
	Push	2.044 (0.211)	0.069 (0.032)	1.180 (0.160)

from the environment and the tool, and partial point clouds using a simulated camera. We use this to construct our training dataset $\mathcal{D}$.

B. Model Details

We collect a training dataset by collecting 5000 push and 5000 pivot trajectories in our simulation with each geometry class we used. Each trajectory samples new geometric and physical properties and runs up to 40 time steps or until the object breaks contact with the tool. We filter the transitions to only include examples in contact. We additionally collect a validation dataset of 500 push and 500 pivot trajectories in the same manner. This yields a final training dataset of 262,780 train transitions and 66,199 validation transitions. During training, we use an action horizon of 4. We implement our network in PyTorch and solve our CQP using CvxPy. Our model is trained for 50 epochs and we use the validation dataset to select the best model. For all experiments, we set our constants from Sec. IV as $T = 4, \alpha = 1.0, \beta = 1.0, \gamma = 0.1, \zeta = 1e{-}3$.

VI. Results

A. Simulated Results

We start by examining our method's performance on a test dataset. We follow the same procedure outlined in Sec. V and collect 500 push and 500 pivot trajectories per object class. We examine our methods ability to predict tool contact information and object motion and forces.

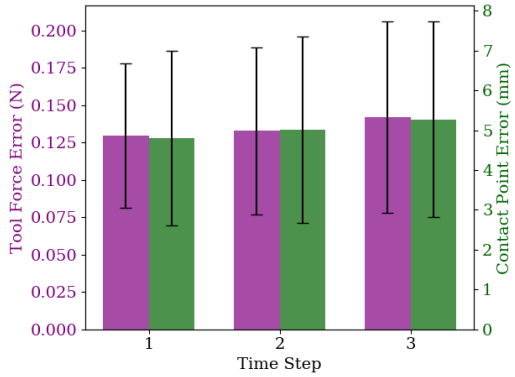
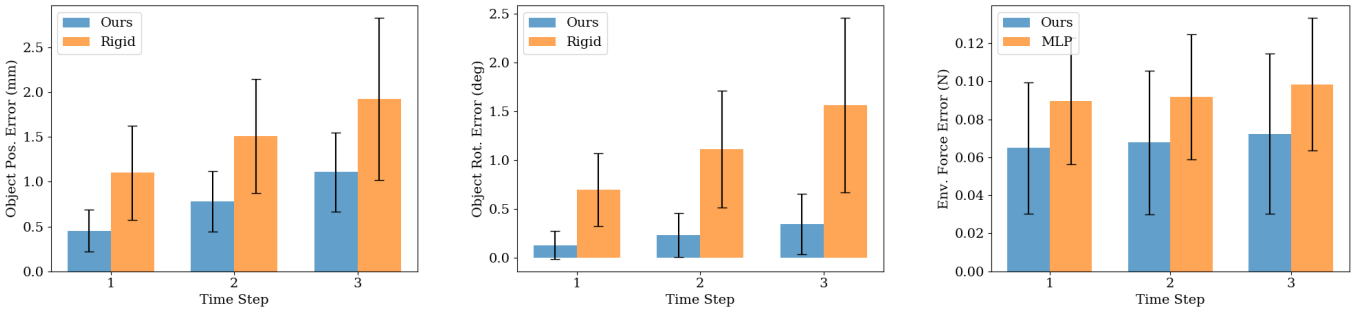


Fig. 4: Tool Contact Force (purple) and Contact Point Location (green) error for our proposed model, plotted by prediction horizon time step. Error bars indicate one half standard deviation.

1) Tool Contact Prediction: We report our method's mean performance over a prediction horizon of three steps for tool contact point and contact force accuracy across all our simulated test trajectories in Fig. 4. For all interactions and across our prediction horizon, our method showed the ability to accurately estimate future contact points between the deforming tool and extrinsic block, to within 6mm on average (against a tool of size 46mm), and showed the ability to estimate future contact forces to within 0.2N on average.

2) Object Motion Prediction: To benchmark our method's object motion performance, we introduce a *Rigid* baseline which assumes that the block moves "rigidly" with respect to the tool. This assumes that the deformation of the tool is negligible for the object motion and doesn't model collisions.

We report our method and the rigid method's object position and rotation accuracy in Figs. 5a and 5b. We show that across all interaction types, our method outperformed the *Rigid* baseline, with sub-millimeter position accuracy and less than 0.5 degrees of rotational error on average across the full prediction horizon.



(a) Object Position Error (↓) (b) Object Rotation Error (↓) (c) Extrinsic Force Error (↓)

Fig. 5: Baseline comparison of our method on test simulated interactions. For (a) and (b) we compare to a Rigid baseline that predicts block motion as if rigidly attached to the tool. For (c) we compare our model-based optimization for extrinsic contact recovery to directly predicting it from an MLP. Error bars indicate one half standard deviation.

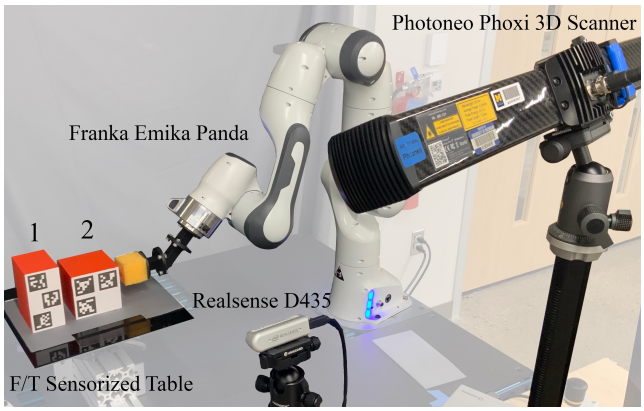


Fig. 6: Our real robot setup, with the test objects. We use the Realsense D435 to track the block pose and the Phoxi 3D Scanner to recover partial pointclouds of the tool. We use a Force/Torque sensorized table to indirectly evaluate our force predictions.

3) Environment-Object Force Prediction: A key part of our proposed method is utilizing our physical priors and the quasi-static assumption to solve for the force between the environment and extrinsic object. We hypothesized that this allows for accurate modeling without requiring additional training effort. To benchmark this approach, we create a variation which replaces the CQP with another MLP following the form of our other network output heads. In particular, it takes in the latent state $\boldsymbol{l}_t$ and directly predicts the contact forces $\boldsymbol{f}_t^{c,1}, ..., \boldsymbol{f}_t^{c,K}$. In practice, the baseline predicts forces at set candidate points on the object and we use the predicted object motion to detect which points lie in contact.

We report our method and baseline method's environment-object force accuracy across all interaction types in Fig. 5c. We find that our proposed method outperforms the baseline, yielding the best average force prediction across all scenarios. Our method consistently performed to within 0.08N of error on average, across the prediction horizon. Our results indicate that our proposed CQP method for recovering environment forces outperforms direct learning.

Finally, Fig. 3 (Left) shows qualitative examples of our methods predictions on simulated data with different objects and actions (pivoting and pushing). The predicted object motion, tool contact and force, and environment forces are consistent with the ground truth, as indicated by the significant overlap with the ground truth (shown as semi-transparent).

B. Real Robot Results

We demonstrate our methods performance on a real robot system. We use a Franka Emika Panda robot and interact with two rectangular objects. Our setup and the test objects are shown in Fig. 6. We collect 5 push and 5 pivot trajectories with each object. We use a Realsense D435 and AprilTags to track the object state and a Photoneo Phoxi 3D Scanner sensor and the Segment Anything Model [27] to get our segmented partial pointcloud of the tool. Finally, we mount an ATI Gamma Force/Torque sensor under the tabletop. This allows us to compare the cumulative force experienced by the table via the extrinsic contacts, which we compare to our predicted extrinsic contacts.

In Table II, we show our one-step object motion prediction and extrinsic force prediction on the real data. Despite sensor noise and the sim-to-real gap, we are still able to predict object motion to within roughly 2mm and 2 degrees error. Our extrinsic contact prediction is within roughly 1N on average. In Fig. 3 (Right), we show full qualitative predictions of block motion, and intrinsic/extrinsic forces. We see that our method yields qualitatively realistic force estimates given real sensor feedback on the physical system.

C. Force Tracking

Finally, we demonstrate a task utilizing our force predictions. Regulating force on the extrinsic object can be important in cases where the object or environment are fragile, such as when handling food, or when the robot or end effector can break or tear. Here, we demonstrate force tracking using our proposed extrinsic contact estimates. We use Block 1 from Sec. VI-B and execute a pivoting motion, pivoting around the contact point furthest from the robot. After an initial press into the object, we have the robot follow an arc, such that the robot wrist is a specified distance d from the pivot point. We setup a

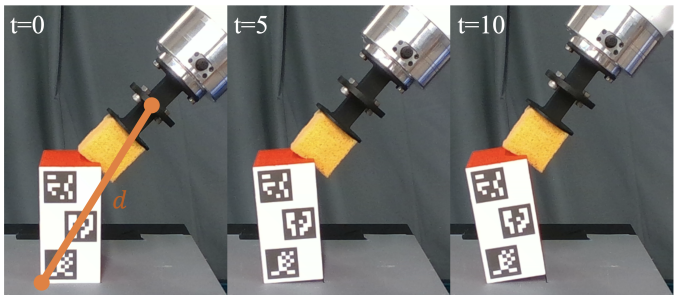


Fig. 7: Our pivot force tracking task, with the decision variable d shown in leftmost frame.

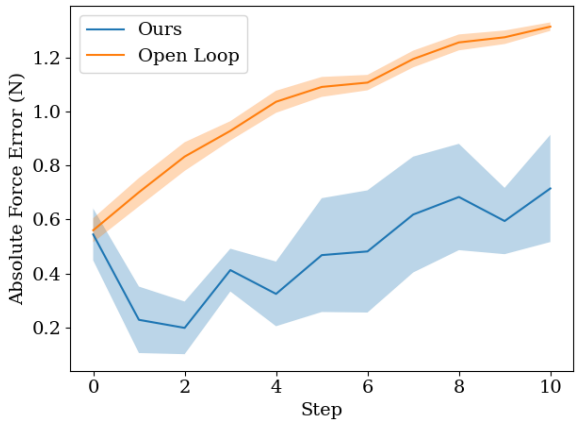


Fig. 8: We plot, over five trials, the evolution of the absolute force magnitude error as a function of the trajectory step. Our method is able to better track the desired force, compared to a method without force feedback executing a similar pivot.

simple greedy controller which samples various distances d for the next step, thereby allowing it to press harder or release as it pivots. We use our proposed model and predict the motion and forces for the sampled actions (with a single-action horizon). We then select the action with the resulting cumulative force magnitude closest to the desired force setting ν. We set the target force magnitude ν to be the gravitation force given the mass of the block plus 1.5N. We can then use the F/T sensor mounted on the environment to determine approximately how well we tracked the desired force. An example execution is shown in Fig. 7.

We ran five trials using this simple greedy controller to track the desired force profile. We then executed our baseline, which started at the same starting configuration, but kept the initial d fixed. We used our environment Force/Torque sensor to measure the actual composite force transfer and plot how the errors evolved over the trials in Fig. 8. Our method is able to better track the desired force magnitude, while the open loop plan gradually drifts further from the desired force profile.

VII. Discussion

We present a method capable of jointly recovering object motion and intrinsic/extrinsic contact information during dexterous deformable manipulations. We found that our first-principles approach outperformed learning methods when recovering environment forces. In future work, we hope to leverage this work for the planning of dexterous deformable motion, where we aim to utilize force estimates to reason about contact modes and force targets.

A limitation of our method is the assumption of quasi-static motion. While this is a common assumption made during dexterous manipulation [12], [15], we may wish for more dynamic motions during certain tasks [37]. A potential direction to investigate is incorporating predicted forces into the underlying rigid body dynamics [22].

We have limited our investigation here to planar motions of the extrinsic object. While our method can be extended to full SE(3) motions, this requires extending the data collection in simulation to consider those motions and extending our CQP to consider 3D forces.

Additionally, our method assumes knowledge of the geometry of the extrinsic object and the environment, which may not be readily available. One potential remedy would be utilizing a shape reconstruction algorithm to estimate these geometries [38].

Finally, our method relies on supervised learning, requiring simulated data to train and successful transfer from simulation to the real robot. Using object motion consistency with predicted forces could be a way to directly train on real system rollouts [21].

References

[1] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Science Robotics*, vol. 6, no. 54, p. eabd8803, 2021. [Online]. Available: https://www.science.org/doi/abs/10.1126/scirobotics.abd8803

[2] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez, "Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1927–1934.

[3] A. Alspach, K. Hashimoto, N. Kuppuswamy, and R. Tedrake, "Soft-bubble: A highly compliant dense geometry tactile sensor for robot manipulation," in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, 2019, pp. 597–604.

[4] H. Sun, K. J. Kuchenbecker, and G. Martius, "A soft thumb-sized vision-based sensor with accurate all-round force perception," *Nature Machine Intelligence*, vol. 4, no. 2, pp. 135–145, 2022.

[5] U. Yoo, N. Dennler, M. Mataric, S. Nikolaidis, J. Oh, and J. Ichnowski, "Moe-hair: Toward soft and compliant contact-rich hair manipulation and care," in *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, 2024, pp. 1163–1167.

[6] M. V. d. Merwe, D. Berenson, and N. Fazeli, "Learning the dynamics of compliant tool-environment interaction for visuo-tactile contact servoing," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 2052–2061. [Online]. Available: https://proceedings.mlr.press/v205/merwe23a.html

[7] A. Goncalves, N. Kuppuswamy, A. Beaulieu, A. Uttamchandani, K. M. Tsui, and A. Alspach, "Punyo-1: Soft tactile-sensing upper-body robot for large object manipulation and physical human interaction," in *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, 2022, pp. 844–851.

[8] M. Oller, M. P. i. Lisbona, D. Berenson, and N. Fazeli, "Manipulation via membranes: High-resolution and highly deformable tactile sensing and control," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1850–1859. [Online]. Available: https://proceedings.mlr.press/v205/oller23a.html

[9] Y. Wi, P. Florence, A. Zeng, and N. Fazeli, "Virdo: Visio-tactile implicit representations of deformable objects," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 3583–3590.

[10] X. Han, J. Masterjohn, and A. Castro, "A convex formulation of frictional contact between rigid and deformable bodies," *IEEE Robotics and Automation Letters*, 2023.

[11] K.-T. Yu, M. Bauza, N. Fazeli, and A. Rodriguez, "More than a million ways to be pushed. a high-fidelity experimental dataset of planar pushing," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 30–37.

[12] Y. Hou, Z. Jia, and M. T. Mason, "Fast planning for 3d any-pose-reorienting using pivoting," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 1631–1638.

[13] F. R. Hogan, J. Ballester, S. Dong, and A. Rodriguez, "Tactile dexterity: Manipulation primitives with tactile feedback," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 8863–8869.

[14] Y. Shirai, D. K. Jha, A. U. Raghunathan, and D. Hong, "Tactile tool manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 597–12 603.

[15] M. Oller, D. Berenson, and N. Fazeli, "Tactile-Driven Non-Prehensile Object Manipulation via Extrinsic Contact Mode Control," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

[16] S. Suresh, H. Qi, T. Wu, T. Fan, L. Pineda, M. Lambeta, J. Malik, M. Kalakrishnan, R. Calandra, M. Kaess *et al.*, "Neuralfeels with neural fields: Visuotactile perception for in-hand manipulation," *Science Robotics*, vol. 9, no. 96, p. eadl0628, 2024.

[17] A. Byravan and D. Fox, "Se3-nets: Learning rigid body motion using deep neural networks," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 173–180.

[18] A. Byravan, F. Leeb, F. Meier, and D. Fox, "Se3-pose-nets: Structured deep dynamics models for visuomotor control," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3339–3346.

[19] D. Seita, Y. Wang, S. J. Shetty, E. Y. Li, Z. Erickson, and D. Held, "Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1038–1049. [Online]. Available: https://proceedings.mlr.press/v205/seita23a.html

[20] K. R. Allen, T. L. Guevara, Y. Rubanova, K. Stachenfeld, A. Sanchez-Gonzalez, P. Battaglia, and T. Pfaff, "Graph network simulators can learn discontinuous, rigid contact dynamics," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1157–1167. [Online]. Available: https://proceedings.mlr.press/v205/allen23a.html

[21] S. Pfrommer, M. Halm, and M. Posa, "Contactnets: Learning discontinuous contact dynamics with smooth, implicit representations," in *Proceedings of the 2020 Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Kober, F. Ramos, and C. Tomlin, Eds., vol. 155. PMLR, 16–18 Nov 2021, pp. 2279–2291. [Online]. Available: https://proceedings.mlr.press/v155/pfrommer21a.html

[22] R. Calandra, S. Ivaldi, M. P. Deisenroth, E. Rueckert, and J. Peters, "Learning inverse dynamics models with contacts," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 3186–3191.

[23] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," in *International Conference on Learning Representations*, 2019.

[24] J. M. Ferguson, D. C. Rucker, and R. J. Webster, "Unified shape and external load state estimation for continuum robots," *IEEE Transactions on Robotics*, vol. 40, pp. 1813–1827, 2024.

[25] P. Wang, Z. Xie, W. Xin, Z. Tang, X. Yang, M. Mohanakrishnan, S. Guo, and C. Laschi, "Sensing expectation enables simultaneous proprioception and contact detection in an intelligent soft continuum robot," *Nature Communications*, vol. 15, no. 1, p. 9978, 2024.

[26] M. J. V. der Merwe, Y. Wi, D. Berenson, and N. Fazeli, "Integrated Object Deformation and Contact Patch Estimation from Visuo-Tactile Feedback," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

[27] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollar, and R. Girshick, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 4015–4026.

[28] L. Manuelli and R. Tedrake, "Localizing external contact using proprioceptive sensors: The contact particle filter," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 5062–5069.

[29] S. Kim and A. Rodriguez, "Active extrinsic contact sensing: Application to general peg-in-hole insertion," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 10 241–10 247.

[30] D. Ma, S. Dong, and A. Rodriguez, "Extrinsic contact sensing with relative-motion tracking from distributed tactile measurements," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 11 262–11 268.

[31] C. Higuera, S. Dong, B. Boots, and M. Mukadam, "Neural contact fields: Tracking extrinsic contact with tactile sensing," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 576–12 582.

[32] J. Masterjohn, D. Guoy, J. Shepherd, and A. Castro, "Velocity level approximation of pressure field contact patches," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 593–11 600, 2022.

[33] Y. Wi, A. Zeng, P. Florence, and N. Fazeli, "Virdo++: Real-world, visuo-tactile dynamics and perception of deformable objects," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 1806–1816. [Online]. Available: https://proceedings.mlr.press/v205/wi23a.html

[34] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[35] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International conference on machine learning*. PMLR, 2019, pp. 2555–2565.

[36] R. Tedrake and the Drake Development Team, "Drake: Model-based design and verification for robotics," 2019. [Online]. Available: https://drake.mit.edu

[37] C. Wang, S. Wang, B. Romero, F. Veiga, and E. Adelson, "Swingbot: Learning physical features from in-hand tactile exploration for dynamic swing-up manipulation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5633–5640.

[38] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.