ContactNets: Learning Discontinuous Contact Dynamics with Smooth, Implicit Representations

Samuel Pfrommer[*], **Mathew Halm**[*], **and Michael Posa**
GRASP Laboratory, University of Pennsylvania
{spfrom, mhalm, posa}@seas.upenn.edu

Abstract: Common methods for learning robot dynamics assume motion is continuous, causing unrealistic model predictions for systems undergoing discontinuous impact and stiction behavior. In this work, we resolve this conflict with a smooth, implicit encoding of the structure inherent to contact-induced discontinuities. Our method, *ContactNets*, learns parameterizations of inter-body signed distance and contact-frame Jacobians, a representation that is compatible with many simulation, control, and planning environments for robotics. We furthermore circumvent the need to differentiate through stiff or non-smooth dynamics with a novel loss function inspired by the principles of complementarity and maximum dissipation. Our method can predict realistic impact, non-penetration, and stiction when trained on 60 seconds of real-world data.

1 Introduction

To effectively perform a wide variety of manipulation tasks, intelligent robots must understand not only how they can affect the motion of objects in their environment, but also how objects in their environment interact with one another. While recent accomplishments in planning [1, 2] and control [3], suggest that a model of the robot-environment system's dynamics is a highly useful formalism for capturing this behavior, producing an accurate model from data is a challenging task for manipulation systems due to the complex behaviors induced by frictional contact.

Many methods for learning a dynamical system attempt to fit a universal function approximator to the system's equations of motion [4, 5, 6, 7] or inverse dynamics [8], yet often the underlying inductive biases conflict with the the nature of frictional contact. Two ubiquitous representations, fully-connected deep neural networks (DNNs) and Gaussian processes regression with squared-exponential kernels [5, 6, 7], are biased towards similar interpretations of Occam's razor: the best parameterization (i.e. simplest explanation) is the smoothest interpolator [9] or an infinitely-differentiable regressor [10] of the data. However, physics-based analysis predicts discontinuity [11], non-uniqueness [12], and/or extreme curvature [13] within the equations of motion for systems undergoing frictional contact, and furthermore areas of state-input space crucial to locomotion and manipulation (e.g. footfalls and grasping) are *precisely* where these irregularities arise. This conflict manifests as poor model predictions even in simple scenarios, as illustrated in Figure 1.

Some works attempt to capture discontinuity with multi-modality [14]. However, unstructured multi-modality is computationally intractable for multi-contact behaviors as the number of modes is extremely large, even in toy systems [15]. DNNs can alternatively be conditioned to generate contact-like behaviors by embedding a differentiable physics simulator directly into their structure [16, 17, 18]. However, differentiating through simulation is numerically challenging if discontinuity is approximated with high-curvature [19], and results from these methods have been limited to simulation and quasi-static real-world interaction with highly-compliant objects. Embedding tactile and force/torque sensors into the robot and/or environment can be effective for learning robot-objects interaction [20, 14], though using such methods to learn object-object interactions would necessitate embedding countless sensors in the robot's environment.

We present a novel approach to the problem of learning frictional contact behaviors, Contact-Nets, which eliminates these pervasive difficulties and leads to a well-conditioned problem that

---

[*]The first two authors contributed equally to this work.

1D System Predictions

1D System Loss

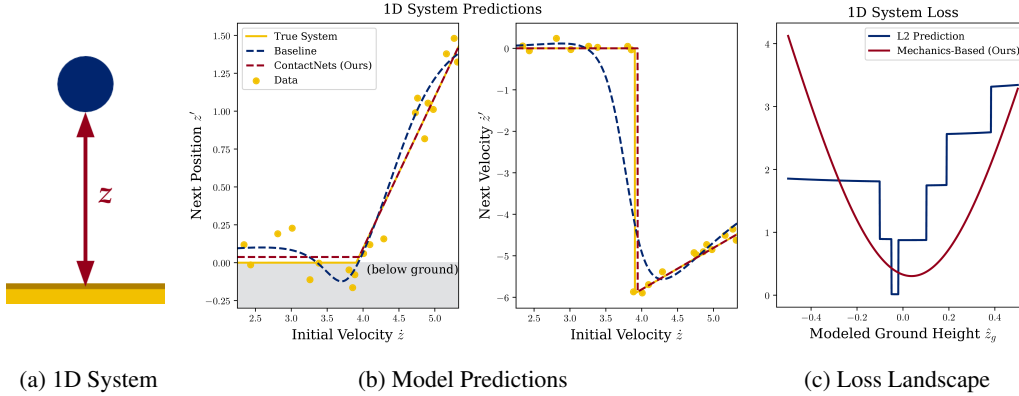(a) 1D System   (b) Model Predictions   (c) Loss Landscape

Figure 1: (a) A 1-D point mass system illustrates contact's pathological nature in model learning. (b) A projection (solid yellow) and noisy sampling (yellow dots) of the system's equations of motion onto the $z = 1$ plane is shown. Fitting a DNN (blue) directly to the data is unable to capture the velocity discontinuity well, and predicts significant ground penetration. Our method (red) instead uses gravitational acceleration as a prior, and learns the height of the contact surface $\hat{z}_g$; though noise prevents a perfect fit, the resulting dynamics qualitatively and quantitatively exceed the unstructured approach. (c) Supervised learning of $\hat{z}_g$ with L_2 loss and gradient descent is ill-posed, as discontinuities in the model are propagated to the loss landscape. We instead ensure reliable training via a smooth novel loss (red) based on contact mechanics. A detailed explanation of this example can be found in Appendix A.1.

is amenable to data-efficient learning. The main contribution of this paper is a reparameterization of the learning problem that effectively uses the inductive bias of DNNs for frictional contact without requiring *any* contact or force sensing. Inspired by both frictional contact mechanics [21, 22, 13, 11] and implicit representations in deep learning [23], we implicitly parameterize and learn discontinuous contact behaviors as continuous inter-body signed distance functions and contact-frame Jacobians. This representation is equivalent to the parameterization of contact internal to simulators for robotics (e.g. [24, 11, 21, 22, 13]), permitting generation of realistic discontinuous behaviors with efficient numerical optimization. At training time, we circumvent the numerical challenges of differentiating through discontinuous simulators with a novel loss function inspired by the principles of complementarity and maximum dissipation. We evaluate our method on a real-world, dynamic, 3D frictional contact scenario and compare its performance to an unstructured baseline.

2 Background

A robotic manipulator interacting with rigid objects and environment can be modeled with inputs $\boldsymbol{u}$ (e.g. motor torques) and states $\boldsymbol{x} = [\boldsymbol{q}; \boldsymbol{v}]$, where $\boldsymbol{q}$ represents the robot's configuration and object poses and $\boldsymbol{v}$ represents velocities. The discrete-time dynamics $\boldsymbol{x}' = f(\boldsymbol{x}, \boldsymbol{u})$ of this system can be formulated as a generalization of Newton's second law emerging from Lagrangian mechanics:

$$\boldsymbol{M}(\boldsymbol{q})(\boldsymbol{v}' - \boldsymbol{v}) = \boldsymbol{F}_{net}(\boldsymbol{x}, \boldsymbol{u}). \tag{1}$$

$\boldsymbol{M}(\boldsymbol{q})$ represents inertial quantities, and $\boldsymbol{F}_{net}$ is the net generalized impulse over the timestep[2]. Configurations are updated via integration of the velocity[3], e.g.

$$\boldsymbol{q}' - \boldsymbol{q} = \boldsymbol{v}'\Delta t. \tag{2}$$

For a system experiencing up to m contact interactions, $\boldsymbol{F}_{net}$ can be decomposed as

$$\boldsymbol{F}_{net}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{F}_s(\boldsymbol{x}, \boldsymbol{u}) + \sum_{i=1}^{m} \boldsymbol{J}_i(\boldsymbol{q})^T \boldsymbol{\lambda}_i. \tag{3}$$

$\boldsymbol{F}_s$ aggregates smooth, non-contact impulses which emerge from potential (e.g. gravitational), gyroscopic, and input impulses; and for each i, $\boldsymbol{J}_i^T \boldsymbol{\lambda}_i$ is the net impulse due to the ith contact. Here,

---

[2]We use the letter $\boldsymbol{F}$ to denote behaviors emergent from contact forces for notational clarity, but note that $\boldsymbol{F}_{net}$ is the net *impulse* over the timestep, i.e. (Net Force) $\times \Delta t$.

[3]For 3D systems, $\boldsymbol{q}$ and $\boldsymbol{v}$ often use different coordinates (e.g. quaternions and angular velocities) which obey $\boldsymbol{\Gamma}(\boldsymbol{q})\boldsymbol{v} = \frac{d}{dt}\boldsymbol{q}$, where $\boldsymbol{\Gamma}$ is a Jacobian. (2) and (4) become $\boldsymbol{q}' - \boldsymbol{q} = \boldsymbol{\Gamma}(\boldsymbol{q})\boldsymbol{v}'\Delta t$ and $\boldsymbol{J}_{n,i} = \nabla_{\boldsymbol{q}}\phi_{n,i}\boldsymbol{\Gamma}$.

$\boldsymbol{J}_i = [\boldsymbol{J}_{n,i}; \boldsymbol{J}_{t,i}]$ is the configuration-dependent contact Jacobian which maps generalized velocities into Euclidean velocities in the ith contact frame normal ($\boldsymbol{J}_{n,i}$) and tangential ($\boldsymbol{J}_{t,i}$) directions. $\boldsymbol{\lambda}_i = [\boldsymbol{\lambda}_{n,i}; \boldsymbol{\lambda}_{t,i}]$ are the contact-frame normal impulses $\boldsymbol{\lambda}_{n,i}$, which resist interpenetration; and frictional impulses $\boldsymbol{\lambda}_{t,i}$, which resist sliding motion between the contacting surfaces.

The underlying mathematics in simulators for rigid robots and environments (e.g. MuJoCo [13], Bullet [11], Drake [24], and others [21, 22]) stray very little from the structure in (1)–(3), and are primarily differentiated in their methodology for calculating the contact impulses $\boldsymbol{\lambda}_i$. However, many models approximate the same two essential characterizations of contact behavior:

- **Normal complementarity**: The signed distance function $\boldsymbol{\phi}_n(\boldsymbol{q}) = [\phi_{n,1}; \ldots; \phi_{n,m}] \in \mathbb{R}^m$ captures contact geometry as inter-body distances. Because bodies cannot interpenetrate and normal forces only push bodies apart when they touch, for each contact i,

$$\boldsymbol{J}_{n,i} = \nabla_{\boldsymbol{q}} \phi_{n,i}, \qquad \phi_{n,i} \geq 0, \qquad \boldsymbol{\lambda}_{n,i} \geq 0, \qquad \phi_{n,i}\boldsymbol{\lambda}_{n,i} = 0. \qquad (4)$$

- **Maximal dissipation**: Many friction models pick $\boldsymbol{\lambda}_{t,i}$ from an admissible set $\Lambda_{t,i}$ such that mechanical power loss is maximized [22, 25]:

$$\boldsymbol{\lambda}_{t,i} \in \underset{\boldsymbol{\lambda}'_{t,i} \in \Lambda_{t,i}}{\arg\min} \quad \boldsymbol{\lambda}'_{t,i} \cdot \boldsymbol{J}_{t,i}(\boldsymbol{q})\boldsymbol{v}, \qquad (5)$$

For instance, Coulomb's friction model with coefficient of friction $\boldsymbol{\mu}_i$ uses

$$\Lambda_{t,i} = \left\{ \boldsymbol{\lambda}_{t,i} : \|\boldsymbol{\lambda}_{t,i}\|_2 \leq \boldsymbol{\mu}_i\boldsymbol{\lambda}_{n,i} \right\}. \qquad (6)$$

For nonzero velocities (sliding), (5) has the closed form solution

$$\boldsymbol{\lambda}_{t,i} = -\frac{\boldsymbol{J}_{t,i}\boldsymbol{v}}{\|\boldsymbol{J}_{t,i}\boldsymbol{v}\|_2} \boldsymbol{\mu}_i\boldsymbol{\lambda}_{n,i}. \qquad (7)$$

3 Related work

A large body of research ([1, 2, 14, 26, 15, 17, 18, 20] and others) in learning and robotics has recognized many challenges in identifying and controlling frictional contact behaviors. Even smooth contact-induced motion is complex due to partial observability, multi-modality, and stochasticity. Several recent works focus on objects sliding on flat surfaces via pushing; Zhou et al. [25] learn a set-valued representation of frictional forces via convex optimization and Bauza and Rodriguez [6] learn a pusher-slider system's dynamics via Gaussian processes regression. Fazeli et al. [7] learn the mapping from pre-impact velocity to post-impact velocity of a planar object falling onto a flat surface, and show superior performance to mechanics-based models. Ajay et al. [27] as well as [7] utilize *residual physics*, in which the gap between a physics simulator and real system's motion is learned. While these methods produce rich descriptions of frictional contact, they all critically assume *a priori* knowledge of what contacts are active. In this work we instead focus on the separate challenge of learning *where and when* discontinuous and non-smooth behaviors including impact and stick-slip transitions occur.

Many methods tackle this problem by inserting a mechanics-based or learned physics model with analytical gradients in an end-to-end optimization framework, such as a DNN. de Avila Belbute-Peres et al. [17], for instance, develop a piecewise-continuous, mechanics-based simulator, and differentiate through continuous motion assuming known, fixed, discontinuous impacts. Ignoring the discontinuities can adversely affect the prediction loss landscape (see Figure 1c), and makes learning contact geometries impossible as no gradients are propagated back to inter-body distance. Other methods represent all contact behaviors as a learned, fully differentiable DNN. Battaglia et al. [16] structure multi-object simulation as pairwise interactions, but even a single object experiencing impact is difficult to model as a DNN (See Sections 5–6), and their method is not tested on real-world data. Li et al. [18] extend this method to real-world deformable body manipulation, but the extreme compliance and quasi-static motion present in their setting does not exhibit the discontinuities that are fundamental to essential robotics tasks. Other works model discontinuity as multi-modality; Fazeli et al. [14] for instance learn a hierarchical model that does not embed strong physics-based priors, yet is capable of segmenting a handful of distinct contact modes when pushing Jenga blocks out of a tower. However, such methods typically scale in complexity with the number of smooth
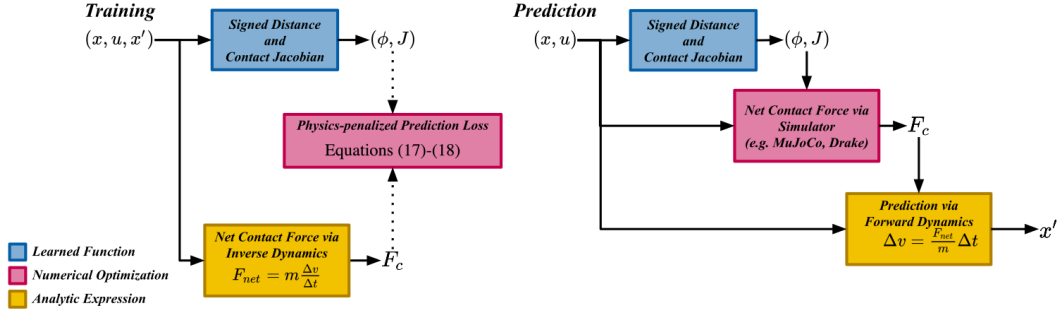
Figure 2: Computation graphs for the proposed model. At training time (left), contact-related terms $(\phi, \boldsymbol{J})$ are generated from a state transition, and a loss on the model is computed as the solution to an optimization problem that quantifies how well the the terms fit the transition. Prediction is performed by inputting the learned $(\phi, \boldsymbol{J})$ to a contact simulation environment, which typically solves a conic program (e.g MuJoCo [13]), unconstrained nonlinear program (e.g. Drake [24]), or linear complementarity program (e.g. Bullet [11] and others [22, 21]) to generate feasible contact forces.

modes, which grows combinatorially with the number of objects in the environment. Calandra et al. [20] instead leverage the continuity of the state-update equations (1), (3) in the contact forces $\boldsymbol{\lambda}_i$, and learn the mapping from current state *and* sensed contact forces to next state. This approach requires full observation of $\boldsymbol{\lambda}_i$, which makes multi-step prediction impossible and furthermore requires sensors to be embedded in the environment for multi-object manipulation.

Fazeli et al. [28] is perhaps the closest in spirit to our method, in which a nonlinear optimization problem (NLP) is developed to identify a handful of parameters for planar systems with contact, and furthermore establishes a complementarity-based loss similar to ours. However, computation of the NLP grows intractably with the number of contacts and datapoints. Our bilevel optimization instead uses efficient convex optimization in the inner loop, and scalable, gradient-based unconstrained optimization in the outer loop, enabling identification of complex geometries from a large dataset.

4 Approach

We now present *ContactNets*, our approach to learning a model explicitly capable of predicting discontinuous contact impulses. We consider systems containing rigid robots and environments, so that the model class (1)–(3) readily applies. For such systems, identification of the contact-free dynamics, parameterized by inertial quantities $\boldsymbol{M}$ and non-contact impulses $\boldsymbol{F}_s$, has been studied by roboticists for decades, and many algorithms exist for learning accurate models (e.g. in [29, 30]). Therefore, we additionally assume that $\boldsymbol{M}$ and $\boldsymbol{F}_s$ are known (either from a hand-designed or learned model), and focus on learning to predict the contact impulses, as in Calandra et al. [20].

An overview of our approach is given in Figure 2, which depicts how *ContactNets* models are trained and tested. All contact behaviors in (1)–(3) are determined solely by the inter-body distance $\phi_{n,i}(\boldsymbol{q})$, contact Jacobian $\boldsymbol{J}_i(\boldsymbol{q})$, and friction coefficient $\boldsymbol{\mu}_i$. Furthermore, while contact dynamics can be complex, discontinuous, and multimodal, $\phi_{n,i}$ and $\boldsymbol{J}_i$ are often modeled as simple, smooth functions. Therefore, at training time we learn functional approximations of these quantities using only state transitions $\mathcal{D} = (\boldsymbol{x}_j, \boldsymbol{u}_j, \boldsymbol{x}'_j)_{j \in 1, \dots, D}$. This process does not require *any* tactile or force/torque sensor information, which crucially does not require instrumenting each object with contact sensing hardware, an impractical requirement outside of a tightly-controlled research environment. This is in contrast to other methods (e.g. [20, 14]) which learn a mapping from the outputs of contact force sensing hardware to $\boldsymbol{F}_{net}$ directly. At test time, we can predict the net impulse $\boldsymbol{F}_{net}$ using the same mathematics from well-established simulation techniques, including MuJoCo [13], Bullet [11], Drake [24], and others [21, 22]. Furthermore, for generating intelligent behaviors, the learned parameterization is compatible with several dynamic planning algorithms [1, 2] and controllers [3] that have specifically been designed to handle the challenges of frictional contact.

4.1 Model parameterization

We construct an approximation of inter-body distances $\phi_n^{\boldsymbol{\theta}}(\boldsymbol{q})$ with parameters $\boldsymbol{\theta}$, and we calculate $\boldsymbol{J}_{n,i}^{\boldsymbol{\theta}}(\boldsymbol{q}) = \nabla_{\boldsymbol{q}} \phi_{n,i}^{\boldsymbol{\theta}}$ using back-propagation. $\phi_n^{\boldsymbol{\theta}}$ can embed strong geometric priors (e.g. a polytope

4

contacting flat ground), enabling learning of a handful of features from sparse training data. Alternatively, essentially arbitrary object and robot geometries can be represented if $\phi_n^{\boldsymbol{\theta}}$ is a DNN. For some systems (e.g. a polytope or strictly convex shape contacting flat ground), the tangential contact Jacobian $\boldsymbol{J}_{t,i}$ is also the gradient of some function of configuration $\boldsymbol{q}$, so we create another function approximator $\phi_t^{\boldsymbol{\theta}}$ and calculate $\boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}(\boldsymbol{q}) = \nabla_{\boldsymbol{q}} \phi_{t,i}^{\boldsymbol{\theta}}$.

In order to make our formulation tractable and well-posed, we also assume the following:

- **Configuration-dependent Coulomb friction:** To simplify the maximal dissipation constraint (5), we assume that, given a particular contact location, friction behaves according to Coulomb's model–i.e. for some configuration-dependent friction coefficient $\boldsymbol{\mu}_i(\boldsymbol{q})$, (6) and (7) hold. Due to scale invariance in the net force calculation, we equivalently assume $\boldsymbol{\mu}_i(\boldsymbol{q}) = 1$ and learn the lumped term $\boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}(\boldsymbol{q}) = \boldsymbol{\mu}_i(\boldsymbol{q})\boldsymbol{J}_{t,i}(\boldsymbol{q})$.

- **Discrete-time contact activation:** The complementarity condition (4) ideally holds for all times, but the dataset only contains a discrete-time sampling of the state. We therefore make the approximating assumption that if the ith contact is active at all during a time-step, then it is still active at the *end* of the step (a standard approach in simulation [22]). That is,

$$\phi_{n,i}(\boldsymbol{q}') \geq \boldsymbol{0}, \qquad \boldsymbol{\lambda}_{n,i} \geq \boldsymbol{0}, \qquad \phi_{n,i}(\boldsymbol{q}')\boldsymbol{\lambda}_{n,i} = 0. \qquad (8)$$

(8) holds approximately for inelastic impacts or high sampling frequencies.

While the contact forces in real systems do not behave exactly according to these assumptions, models in this class are accurate enough to produce agile and accurate motion in locomotion and manipulation tasks (e.g. Fallon et al. [31]).

4.2 Loss formulation

To train our models, we need a loss function $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}')$ that captures how well they explain a particular transition $(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}')$. We observe that, even in the absense of sensing for the *individual* contact impulses $\boldsymbol{\lambda}_i$, knowledge of the contact-free dynamics enables a ground-truth observation of the net contact impulse $\boldsymbol{F}_c$ from the data:

$$\boldsymbol{F}_{c,data}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}') = \boldsymbol{M}(\boldsymbol{q})(\boldsymbol{v}' - \boldsymbol{v}) - \boldsymbol{F}_s(\boldsymbol{x}, \boldsymbol{u}). \qquad (9)$$

A key insight, taking inspiration from multi-contact simulation [22] and planning [2], is to hypothesize a candidate set of contact impulses $\boldsymbol{\lambda}$. Given such a $\boldsymbol{\lambda}$, it is straightforward to capture how well the model explains $(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda})$ by a) determining how realistic $\boldsymbol{\lambda}$ is, quantifying violation of complementarity and maximum dissipation, and b) calculating how closely the force $\boldsymbol{\lambda}$ matches $\boldsymbol{F}_{c,data}$. We establish the following costs on $(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda})$:

- **Prediction quality:** $\boldsymbol{\lambda}$ should explain the observed contact forces $\boldsymbol{F}_{c,data}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}')$:

$$l_1(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda}) = \left\| \sum_i \boldsymbol{J}_i^{\boldsymbol{\theta}}(\boldsymbol{q})^T \boldsymbol{\lambda}_i - \boldsymbol{F}_{c,data} \right\|^2. \qquad (10)$$

While (10) is similar in spirit to L_2 loss on the output of a simulator, its mathematical behavior is fundamentally different due to its dependence on the unknown $\boldsymbol{\lambda}$.

- **Contact activation:** Forces should only be applied when contact is established ($\phi_{n,i}(\boldsymbol{q}')\boldsymbol{\lambda}_{n,i} = 0$):

$$l_2(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda}) = \sum_i \phi_{n,i}^{\boldsymbol{\theta}}(\boldsymbol{q}')^2 \|\boldsymbol{\lambda}_i\|_2^2. \qquad (11)$$

- **Non-penetration:** The motion generated by $\boldsymbol{\lambda}_i$ should not cause penetration. Manipulating (1)–(2), we can estimate how $\boldsymbol{\lambda}$ would affect $\phi_{n,i}^{\boldsymbol{\theta}}$ over the time-step as

$$\tilde{\boldsymbol{v}}'(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{v} + \boldsymbol{M}^{-1}(\boldsymbol{F}_s + \sum_i \boldsymbol{J}_i^{\boldsymbol{\theta}}(\boldsymbol{q})^T \boldsymbol{\lambda}_i), \qquad (12)$$

$$\tilde{\phi}_{n,i}'(\boldsymbol{x}, \boldsymbol{\lambda}) = \phi_i^{\boldsymbol{\theta}}(\boldsymbol{q}) + \boldsymbol{J}_{n,i}^{\boldsymbol{\theta}}(\boldsymbol{q})\tilde{\boldsymbol{v}}'(\boldsymbol{x}, \boldsymbol{\lambda})\Delta t. \qquad (13)$$

We then penalize negativity of the predicted signed distance $\tilde{\phi}_{n,i}'(\boldsymbol{x}, \boldsymbol{\lambda})$:

$$l_3(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda}) = \sum_i \min(0, \tilde{\phi}_{n,i}'(\boldsymbol{x}, \boldsymbol{\lambda}))^2. \qquad (14)$$

5

- **Maximal dissipation:** Friction forces $\boldsymbol{\lambda}_{t,i}$ must be chosen such that power loss is maximized. We penalize violation of (6), scaled by $\left\| \boldsymbol{J}_{t,i}^{\boldsymbol{\theta}} \boldsymbol{v}' \right\|_2$:

$$l_4(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda}) = \sum_i \left\| \left\| \boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}(\boldsymbol{q}) \boldsymbol{v}' \right\|_2 \boldsymbol{\lambda}_{t,i} + \boldsymbol{\lambda}_{n,i} \boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}(\boldsymbol{q}) \boldsymbol{v}' \right\|_2^2 . \tag{15}$$

Finally, we choose a positive loss $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}')$ that is equal to 0 if a *single*, feasible set of contact forces $\boldsymbol{\lambda}$ causes each of the costs l_k to be 0:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}') = \min_{\boldsymbol{\lambda}} \quad \sum_k l_k(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}', \boldsymbol{\lambda}), \tag{16}$$

$$\text{s.t.} \quad \boldsymbol{\lambda}_{n,i} \geq 0 , \|\boldsymbol{\lambda}_{t,i}\|_2 \leq \boldsymbol{\lambda}_{n,i} . \tag{17}$$

Each of the individual costs (10), (11), (14), (15) are convex piecewise-quadratic in $\boldsymbol{\lambda}$. Therefore, with appropriately chosen slack variables, we can pose (16) as a tractable, feasible, and convex program, allowing for its gradient to be efficiently computed through sensitivity analysis [32].

This structure is unexpected because the converse problem of predicting the transition via simulation is often formulated as a non-convex problem due to the complementarity constraint (8), as both the signed distance $\phi_n(\boldsymbol{q}')$ and forces $\boldsymbol{\lambda}$ must be solved for simultaneously [22]. It is because the transition is *already observed* and only $\boldsymbol{\lambda}$ is unknown that this difficulty is circumvented.
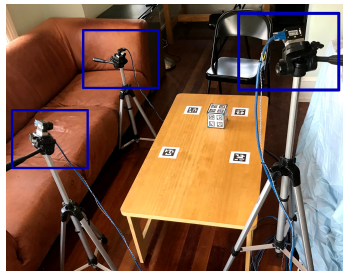
5 Experimental procedure

Figure 3: The experiment was designed to evaluate complex, three dimensional contact. A cube was tossed against the planar ground, producing impacts and both sticking and sliding frictional contact. Cube position and orientation are tracked by three cameras, outlined in blue.
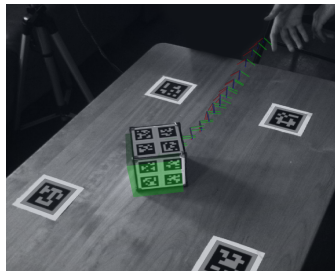
Figure 4: A trace of c.o.m. position and block orientation is superimposed on a typical tossing behavior learned by our method, *ContactNets*. In both real and predicted (green) behavior, the block impacts the ground on a corner and tumbles onto its side before coming to rest.

In order to evaluate our formulation's ability to learn the dynamics of a real system undergoing frictional contact, we conduct experiments on the setup shown in Figure 3; the associated source code[4] is available online. A 10 cm acrylic cube is tossed onto a wooden surface, which produces nearly instantaneous (i.e., sub-timestep) impact behaviors. Cube corners are covered with a thin layer of gelatinous, low-friction material to produce more regular contact behavior. Each side of the cube features four unique AprilTags tracked by three 148 Hz PointGrey cameras using TagSLAM [33]. Object configurations are represented as $\boldsymbol{q} = [\boldsymbol{p}; \ \boldsymbol{R}]$ consisting of the world-frame center of mass position $\boldsymbol{p}$ and rotation matrix $\boldsymbol{R}$, and velocities are represented as $\boldsymbol{v} = [\dot{\boldsymbol{p}}; \boldsymbol{\omega}]$, where $\boldsymbol{\omega}$ is the cube's body-frame angular velocity. After post-processing the original 750 tosses, the collected dataset contains 570 unique, high-quality tosses, whose initial toss directions are then randomly rotated about the world z axis.

5.1 Comparison and metrics

We compare the data efficiency of *ContactNets* versus an unstructured baseline for long-term prediction and physical realism. For different dataset sizes, we train the models until loss converges on validation data, then we evaluate the models on a separate test dataset. We evaluate long-term

---

[4] https://github.com/DAIRLab/contact-nets

prediction via temporally-averaged absolute error over a model rollout; for a particular ground truth trajectory $(\boldsymbol{p}_j^*, \boldsymbol{R}_j^*)_{j \in 1, \ldots, N}$ we construct a predicted trajectory $(\hat{\boldsymbol{p}}_j, \hat{\boldsymbol{R}}_j)_{j \in 1, \ldots, N}$ using *only* the initial condition $(\boldsymbol{p}_1^*, \boldsymbol{R}_1^*)$ by recursing through the model's dynamics, and compute our metrics as

$$e_{pos} = \frac{1}{N} \sum_{j=1}^{N} \left\| \hat{\boldsymbol{p}}_j - \boldsymbol{p}_j^* \right\|_2, \qquad e_{rot} = \frac{1}{N} \sum_{j=1}^{N} \left| \text{angle}(\boldsymbol{R}_j^*, \hat{\boldsymbol{R}}_j) \right|, \tag{18}$$

where $\text{angle}(\boldsymbol{R}^*, \hat{\boldsymbol{R}})$ is the angle of the rotation between $\boldsymbol{R}^*$ and $\hat{\boldsymbol{R}}$. To evaluate physical realism, we additionally examine how much the rollout prediction penetrates the real surface on average:

$$e_{pen} = \frac{1}{N} \sum_{j=1}^{N} \text{penetration}(\hat{\boldsymbol{p}}_j, \hat{\boldsymbol{R}}_j). \tag{19}$$

5.2 Models

5.2.1 ContactNets Polytope (Ours)

This model fits a low dimensional representation of the cube, highlighting *ContactNets*' ability to learn simple, discontinuous dynamics from sparse data. A common approximation of polytopes contacting a flat surface is that only the vertices make contact with the ground. For this model, $\boldsymbol{\theta}$ contains the body-frame locations of the vertices of the cube and the surface normal. Each $\phi_{n,i}^{\boldsymbol{\theta}}$ transforms the ith vertex into the world frame and projects it onto the surface normal. A similar geometric construction is conducted for $\boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}$. Detailed equations can be found in Appendix A.2.

Parameters are optimized using the loss (16), and osqpth [32, 34] is used to compute its gradient. Forward rollouts are computed using a common contact simulation formulation (Stewart and Trinkle [22]). Some additional regularizers are added to prevent simulation artifacts due to the particular behaviors of [22], and are discussed in detail in Appendix A.3.

5.2.2 ContactNets Deep (Ours)

This model extends the polytopic model by adding a DNN to $\phi_n^{\boldsymbol{\theta}}$ and $\phi_t^{\boldsymbol{\theta}}$, e.g.

$$\phi_n^{\boldsymbol{\theta}}(\boldsymbol{q}) = \phi_n^{poly,\boldsymbol{\theta}}(\boldsymbol{q}) + \phi_n^{DNN,\boldsymbol{\theta}}(\boldsymbol{q}), \tag{20}$$

enlarging the model class to include essentially arbitrary object and ground geometries. It is trained in the same fashion as *ContactNets Polytope*.

5.2.3 End-to-end

Here we consider a slight modification of a typical unstructured learned dynamics model $\boldsymbol{x}' = \boldsymbol{f}^{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{u})$. As *ContactNets* only learns to predict the contact impulse, rather than burden the unstructured model with the additional task of identifying continuous dynamics, we instead fit a DNN $\boldsymbol{F}_{c,DNN}^{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{u})$ *directly* to the observed contact forces $\boldsymbol{F}_{c,data}(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}')$, calculated as in (9); this model is trained end-to-end on single-step prediction with L_2 loss:

$$\mathcal{L}_{e2e}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{x}') = \left\| \boldsymbol{F}_{c,data} - \boldsymbol{F}_{c,DNN}^{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{u}) \right\|_2^2. \tag{21}$$

At test time, motion is predicted using equations (1)–(3).

Network architectures and training hyperparameters are discussed in detail in Appendix A.2.

6 Results

We compare the above models in Figure 5. For a range of data sizes, both *ContactNets* methods outperform the *End-to-end* baseline in positional and rotational accuracy, most strikingly for rotational error with ample training data. *End-to-end* rollouts struggle to capture hard face-to-ground contacts and typically drift rotationally; in contrast, *ContactNets* models are capable of capturing such interactions and are primarily limited by stochastic contact behavior and noisy data.
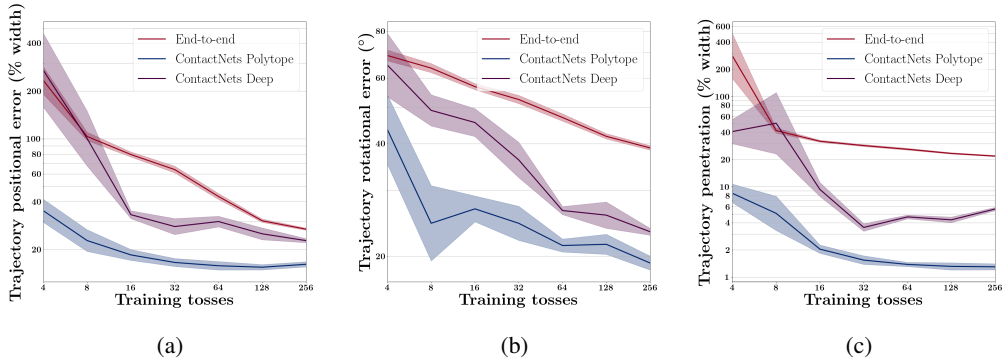
Figure 5: (a, b) *End-to-end* model performance is compared to our two parameterizations of contact: *Contact-Nets Polytope* (low-dimensional) and *ContactNets Deep* (overparameterized). The log-scale graphs show both the mean and 95% confidence interval for a minimum of 4 samples per point, assuming log-normality of the data. Both *ContactNets* methods achieve at least comparable positional and superior rotational performance to the best 256-toss *End-to-end* model after just 32 training tosses. Since the system's geometry is nearly poly-topic, *ContactNets Polytope* performs strongly; for more complex interactions we would expect *ContactNets Deep* to have an advantage. (c) Learning representations of inter-body signed distance leads to trajectories which match physical intuition of realistic contact. Despite having no explicit knowledge of ground height or orientation, *ContactNets* model rollouts produce very little penetration, while *End-to-end* methods fail to capture this important behavior.

The penetration metric supports qualitative observations that *ContactNets* rollouts appear more physically plausible. *End-to-end* models are incapable of producing discontinuous impulses to prevent penetration, and often continue moving after the ground-truth block motion is at rest. This leads to average penetrations of over 20% block width which fail to improve significantly even with more tosses. *ContactNets* rollouts rarely have penetrations of more than 6% block width, with *Contact-Nets Polytope* averaging at just under 2% block width. Penetrative behaviors exacerbate DNNs' poor ability to extrapolate beyond the training distribution; since these states are non-physical, the training data distribution will never include nearby states, leading learned models to perform poorly. Explicitly encoding complementarity into *ContactNets* eliminates this pathological behavior.

The superior performance of *ContactNets Polytope* compared to *ContactNets Deep* can be attributed to the polytopic geometry of the cube. Further experimentation with curved objects that feature rich, non-isotropic frictional behavior should highlight the more flexible parameterization provided by *ContactNets Deep*. We note that directly parameterizing ϕ_n and ϕ_t as DNNs without a polytopic component proved difficult to train due to the possibility of highly unphysical initializations (i.e., ϕ_n representing a "ceiling" above the block, instead of a ground below it). We hope to overcome these difficulties with future work incorporating additional initialization and regularization techniques.

7 Conclusion

Discontinuous and non-unique impact and stiction underpin essential robotics tasks—thus capturing these phenomena in learned models is crucial for their effective use in the real world. Our method, *ContactNets*, presents a novel approach to resolving fundamental problems in representing these behaviors with neural networks, and produces realistic dynamics from sparse training data.

The primary limitation of our model is the constrictive nature of its priors: namely that the analytical contact-free dynamics are exact, collisions are inelastic, and objects are rigid. In future work, we will extend the method to learn continuous forces, and examine models of elastic impact that are consistent with our parameterization, e.g. Anitescu and Potra [21]. Additionally, as real-time data of object poses is unavailable in some applications, natural extensions could involve embedding our formulation into dynamical models based on visual data. Recent advances in keypoint-based approaches [35] suggest a promising intermediate representation for inferring contact geometry from video. Further experimentation involving a manipulator interacting with several objects would allow us to evaluate our formulation's ability to capture multi-body contact, and we will verify the quality of our learned models for executing robotic tasks by utilizing them in planning and control algorithms.

8 Acknowledgements

References

[1] I. Mordatch, Z. Popović, and E. Todorov. Contact-Invariant Optimization for Hand Manipulation. *SCA '12 Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 137–144, 2012. doi:10.2312/SCA/SCA12/137-144.

[2] M. Posa, C. Cantu, and R. Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2013.

[3] A. Aydinoglu, V. Preciado, and M. Posa. Contact-aware controller design for complementarity systems. In *International Conference on Robotics and Automation (ICRA)*, 2020.

[4] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems 31*, pages 4754–4765. Curran Associates, Inc., 2018.

[5] M. Deisenroth, C. Rasmussen, and D. Fox. Learning to control a low-cost manipulator using data-efficient reinforcement learning. In *Robotics: Science and Systems (RSS)*, 2011.

[6] M. Bauza and A. Rodriguez. A probabilistic data-driven model for planar pushing. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3008–3015, 2017.

[7] N. Fazeli, S. Zapolsky, E. Drumwright, and A. Rodriguez. Learning data-efficient rigid-body contact models: Case study of planar impact. *arXiv preprint arXiv:1710.05947*, 2017.

[8] F. Meier, D. Kappler, N. Ratliff, and S. Schaal. Towards robust online inverse dynamics learning. In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016.

[9] M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116 (32):15849–15854, 2019. doi:10.1073/pnas.1903070116.

[10] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning series)*. The MIT Press, 2005. ISBN 9780262182539.

[11] E. Coumans. Bullet physics simulation. In *ACM SIGGRAPH 2015 Courses*, page 7.

[12] M. Halm and M. Posa. Modeling and analysis of non-unique behaviors in multiple frictional impacts. In *Robotics: Science and Systems (RSS)*, 2019.

[13] E. Todorov. Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in mujoco. In *International Conference on Robotics and Automation (ICRA)*, 2014.

[14] N. Fazeli, M. Oller, J. Wu, Z. Wu, J. B. Tenenbaum, and A. Rodriguez. See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion. *Science Robotics*, 4 (26), 2019. doi:10.1126/scirobotics.aav3123.

[15] E. Huang, X. Cheng, and M. T. Mason. Efficient contact mode enumeration in 3d. In *International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2020.

[16] P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende, and k. kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *Advances in Neural Information Processing Systems 29*, pages 4502–4510. Curran Associates, Inc., 2016.

[17] F. de Avila Belbute-Peres, K. Smith, K. Allen, J. Tenenbaum, and J. Z. Kolter. End-to-end differentiable physics for learning and control. In *Advances in Neural Information Processing Systems 31*, pages 7178–7189. Curran Associates, Inc., 2018.

[18] 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.

[19] S. Kolev and E. Todorov. Physically consistent state estimation and system identification for contacts. In *International Conference on Humanoid Robots (Humanoids)*, Nov 2015.

[20] 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)*, pages 3186–3191, 2015.

[21] M. Anitescu and F. A. Potra. Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics*, 14(3):231–247, 1997.

[22] D. Stewart and J. J. Trinkle. An implicit time-stepping scheme for rigid body dynamics with Coulomb friction. *International Journal for Numerical Methods in Engineering*, 39(15):2673–2691, 1996.

[23] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

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

[25] J. Zhou, R. Paolini, J. A. Bagnell, and M. T. Mason. A convex polynomial force-motion model for planar sliding: Identification and application. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 372–377, 2016. doi:10.1109/ICRA.2016.7487155.

[26] M. Halm and M. Posa. A Quasi-static Model and Simulation Approach for Pushing, Grasping, and Jamming. In *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018.

[27] A. Ajay, J. Wu, N. Fazeli, M. Bauza, L. P. Kaelbling, J. B. Tenenbaum, and A. Rodriguez. Augmenting physical simulators with stochastic neural networks: Case study of planar pushing and bouncing. In *International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[28] N. Fazeli, R. Kolbert, R. Tedrake, and A. Rodriguez. Parameter and contact force estimation of planar rigid-bodies undergoing frictional contact. *The International Journal of Robotics Research*, 36(13-14):1437–1454, 2017. doi:10.1177/0278364917698749.

[29] P. K. Khosla and T. Kanade. Parameter identification of robot dynamics. In *1985 24th IEEE Conference on Decision and Control*, pages 1754–1760, Dec 1985.

[30] S. Traversaro, A. Del Prete, R. Muradore, L. Natale, and F. Nori. Inertial parameter identification including friction and motor dynamics. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 68–73, 2013.

[31] M. Fallon, S. Kuindersma, S. Karumanchi, M. Antone, T. Schneider, H. Dai, C. P. D'Arpino, R. Deits, M. DiCicco, D. Fourie, T. Koolen, P. Marion, M. Posa, A. Valenzuela, K. T. Yu, J. Shah, K. Iagnemma, R. Tedrake, and S. Teller. An architecture for online affordance-based perception and whole-body planning. *Journal of Field Robotics*, 32(2):229–254, 2015.

[32] B. Amos and J. Z. Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning (ICML)*, 2017.

[33] B. Pfrommer and K. Daniilidis. Tagslam: Robust slam with fiducial markers. *arXiv preprint arXiv:1910.00679*, 2019.

[34] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: An operator splitting solver for quadratic programs. *arXiv preprint arXiv:1711.08013*, Nov. 2017.

[35] L. Manuelli, W. Gao, P. Florence, and R. Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. In *International Symposium on Robotics Research (ISRR)*, 2019.

A Appendix

A.1 1D Toy Example

(a) 1D System   (b) Model Predictions   (c) Loss Landscape
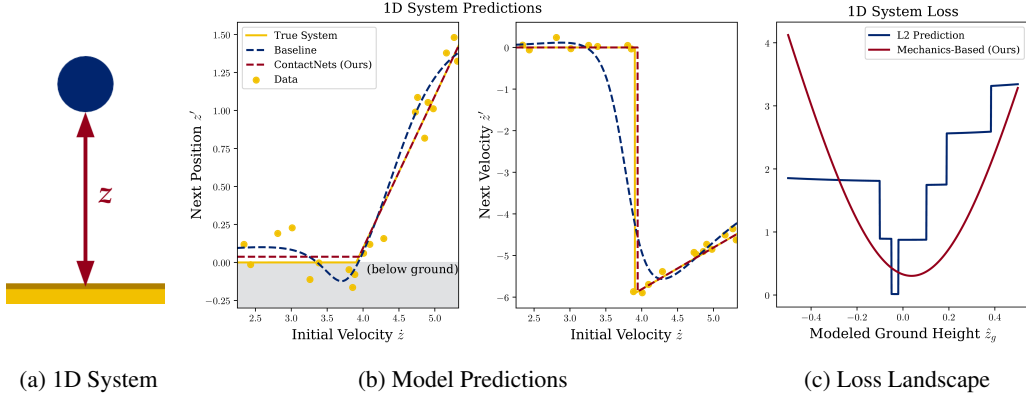
Here, we describe the toy example first shown in Figure 1, which is reproduced for reference.

A.1.1 System Dynamics

In (a), we display a simple, 1D system with contact: a point mass which makes inelastic impact with the ground at height $z = z_g = 0$. The system has state $\boldsymbol{x} = [z; \dot{z}]$ which has freefall motion

$$z'_{free}(z, \dot{z}) = z + \dot{z}\Delta t - \frac{9.81}{2}\Delta t^2 \,, \tag{22}$$

$$\dot{z}'_{free}(z, \dot{z}) = \dot{z} - 9.81\Delta t \,. \tag{23}$$

For a ground height z_g, the next state $\boldsymbol{x}' = [z'; \dot{z}]$ either obeys freefall motion, or impacts the ground and comes to rest:

$$\begin{bmatrix} z' \\ \dot{z}' \end{bmatrix} = f_{z_g}(z, \dot{z}) = \begin{cases} \begin{bmatrix} z'_{free}(z, \dot{z}) \\ \dot{z}'_{free}(z, \dot{z}) \end{bmatrix} & z'_{free}(z, \dot{z}) \geq z_g \,, \\ \begin{bmatrix} z_g \\ 0 \end{bmatrix} & z'_{free}(z, \dot{z}) < z_g \,. \end{cases} \tag{24}$$

In (b), we fix the ground height to $z_g = 0$ and the initial position to $z = 1$ and plot $f_0(1, \dot{z})$ in yellow for $\Delta t = 1$. Note the velocity discontinuity due to impact near $\dot{z} = 4$.

A.1.2 Dynamics Learning

To illustrate the difficulty of fitting a DNN to discontinuity, we consider a simple, unstructured end-to-end dynamics learning setting for the system (24) where we fix $z = 0$ and learn the mapping $\dot{z} \rightarrow \boldsymbol{x}'$. Specifically, we generate training state transitions $(\dot{z}_i, \boldsymbol{x}'_i)_{i \in 1, \dots, 20}$, by uniformly sampling points from the graph of $f_0(1, \dot{z})$ in (b), and perturbing each element of $\dot{z}$ and $\boldsymbol{x}'$ with Gaussian white noise with variance 0.01. The data is shown in (b) as yellow dots.

We train a fully connected DNN $f_{\boldsymbol{\theta}}$ to predict $\boldsymbol{x}'_i \approx f_{\theta}(\dot{z}_i)$. The DNN has 2 hidden layers of width 128 and $\tanh$ activations, and is trained using Adam with PyTorch default parameters using L_2 loss

$$\mathcal{L}(\boldsymbol{\theta}, \dot{z}, \boldsymbol{x}') = \|\boldsymbol{x}' - f_{\boldsymbol{\theta}}(\dot{z})\|_2^2 \,. \tag{25}$$

Training is terminated when the loss converges on a separate, identically distributed validation set. The fully trained network's output is plotted in blue in (b). The trained DNN is unable to capture the velocity discontinuity well, and predicts significant ground penetration. While stopping early on validation loss prevents the model from overfitting to the noisy training data, the result of this regularization merely produces a smooth regressor, and does not recover important qualitative features of the true system. By a similar notion, any naive regularization that encourages smoothness (e.g. weight decay) will generate similar learned models. Furthermore, an unregularized training

process is likely to produce an interpolator of the data, which would exacerbate ground penetration and generate erratic behavior near the discontinuity.

Next, we consider a simple application of *ContactNets* to the 1D system. Without tangential motion along the surface, there are no frictional behaviors in the system; we therefore forego learning related quantities. We consider the simple case of learning an approximation of the ground height $\hat{z}_g$. As in Section 4, we construct the inter-body signed distance, $\phi^{\hat{z}_g}(z) = z - \hat{z}_g$, and contact impulse estimate $\boldsymbol{F}_{c,data}(\dot{z}, \boldsymbol{x}') = (\dot{z}' - \dot{z}) - (-9.81)$. Finally, as there are no frictional behaviors in the system, we construct a simplified version of our mechanics-inspired loss (16):

$$\mathcal{L}(\hat{z}_g, \dot{z}, \boldsymbol{x}') = \min_{\boldsymbol{\lambda}_n \geq 0} \quad \phi^{\hat{z}_g}(z')^2 \boldsymbol{\lambda}_n^2 + (\boldsymbol{F}_{c,data} - \boldsymbol{\lambda}_n)^2 . \tag{26}$$

The average of this loss over the data is plotted in (c) in red for different $\hat{z}_g$. We learn $\hat{z}_g$ by minimizing (26) using Adam with identical training hyperparameters and termination conditions as the DNN model. After recovering a good estimate for ground height, we can predict the next state as $\hat{\boldsymbol{x}}' = f_{\hat{z}_g}(1, \dot{z})$, shown in red in (b). As we embed the key behaviors of contact directly into our model, we both quantitatively and qualitatively outperform the unstructured baseline model. Despite significant noise in the training and validation data, our method produces a ground height $\hat{z}_g$ which closely approximates the true z_g in the underlying system.

Given that we predict state transitions using $f_{\hat{z}_g}(1, \dot{z})$, it might seem natural employ L_2 loss $\left\| \boldsymbol{x}' - f_{\hat{z}_g}(1, \dot{z}) \right\|_2^2$, shown in blue in (c), during training. However, because $f_{\hat{z}_g}$ is discontinuous in $\hat{z}_g$, the L_2 loss is not differentiable or even continuous, leading to numerical challenges. By contrast, our loss is smooth, allowing higher-order methods like Adam to perform well.

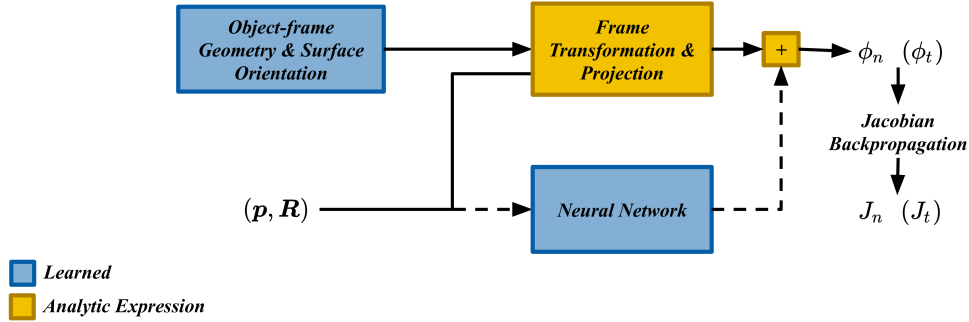## A.2   Learning setup

Model Architecture

Figure 7: Proposed model architectures for learning signed distance and contact-frame Jacobian $\phi, \boldsymbol{J}$. *ContactNets Polytope* is represented by the solid lines above, and parameterizes ϕ_n/ϕ_t in terms of the object-frame block geometry and surface orientation. *ContactNets Deep* adds, in parallel, an overparameterized neural network directly mapping from configuration to ϕ, as indicated by the dashed lines. Jacobians $\boldsymbol{J}_n$ and $\boldsymbol{J}_t$ are computed using Jacobian backpropagation, as described in Appendix A.2.

The optimal network structure for *End-to-end* was empirically determined by varying network width and depth, resulting in 4 hidden layers with 256 neurons and ReLU activations. Network inputs are normalized to have zero mean and unit variance. Training is executed using the PyTorch AdamW optimizer with a learning rate of $3 \cdot 10^{-5}$ and weight decay of 10^{-3}.

The *ContactNets* models are parameterized as depicted in Figure 7. Object-frame geometry and surface orientation vectors are initialized randomly to their ground-truth values with significant added noise (standard deviation of 40% of their original values). For *ContactNets Deep*, a separate network is summed in parallel, featuring two hidden layers of 256 neurons with $\tanh$ activation. Following the addition of regularizers as described in Appendix A.3 with coefficients 0.3, AdamW was used for optimization with a learning rate of $5 \cdot 10^{-4}$ and 0 weight decay.

For the *ContactNets* methods, we require an additional procedure for computing $\boldsymbol{J}_n^{\boldsymbol{\theta}}(\boldsymbol{q})$ from the parameterization of $\phi_n^{\boldsymbol{\theta}}(\boldsymbol{q})$ and a given configuration. This is accomplished by first forwards propagating an input $\boldsymbol{q}$ through the network, keeping note of its value before each operation (activation, weight multiplication, etc.), and then backpropagating a Jacobian matrix using the chain rule. Coupling $\phi_n^{\boldsymbol{\theta}}$ and $\boldsymbol{J}_n^{\boldsymbol{\theta}}$ is critical to ensuring that our learned model produces physically reasonable behavior. We similarly parameterize $\boldsymbol{J}_t^{\boldsymbol{\theta}}$ as the Jacobian of a learned function $\phi_t^{\boldsymbol{\theta}}$.

For all models the train-validation-test split is 50-30-20. Each model is trained until its loss fails to improve on the validation set for atleast 12 epochs (smaller datasets were permitted additional epochs) and is subsequently evaluated on the test dataset in Figure 5.

A.3 Learning regularizers

The LCP-based, semi-implicit method of Stewart and Trinkle [22] is used to simulate rollouts with the learned $(\phi_{n,i}^{\boldsymbol{\theta}}, \boldsymbol{J}_i^{\boldsymbol{\theta}})$. To prevent unrealistic simulation artifacts, the following regularizers $\mathcal{R}_1, \mathcal{R}_2$ were added to the loss (16):

A.3.1 Normal–tangent perpendicularity

For each contact, we expect that contact-frame forces applied to the body due to normal and frictional contact forces to be orthogonal by definition; hence, we encourage the corresponding elements of the normal and tangential contact Jacobians $\boldsymbol{J}_{n,i}^{\boldsymbol{\theta}}$ and $\boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}$ to be perpendicular by penalizing their normalized dot products:

$$\mathcal{R}_1 = \sum_i \left\| \left(\frac{\boldsymbol{J}_{n,i}^{\boldsymbol{p}}}{\left\| \boldsymbol{J}_{n,i}^{\boldsymbol{p}} \right\|_2} \right) \left(\frac{\boldsymbol{J}_{t,i}^{\boldsymbol{p}}}{\left\| \boldsymbol{J}_{t,i}^{\boldsymbol{p}} \right\|_2} \right)^T \right\|_2^2 .$$

Here, $\boldsymbol{J}_{n,i}^{\boldsymbol{p}} = \frac{\partial \phi_{n,i}^{\boldsymbol{\theta}}}{\partial \boldsymbol{p}}$ and $\boldsymbol{J}_{t,i}^{\boldsymbol{p}} = \frac{\partial \phi_{t,i}^{\boldsymbol{\theta}}}{\partial \boldsymbol{p}}$ denote the columns of $\boldsymbol{J}_{n,i}^{\boldsymbol{\theta}}$ and $\boldsymbol{J}_{t,i}^{\boldsymbol{\theta}}$ that relate to the center of mass position.

A.3.2 Position Jacobian unit norm

Regardless of object or table geometry, geometric analysis would imply that $\boldsymbol{J}_{n,i}^{\boldsymbol{p}}$ has unit norm. We therefore additionally penalize

$$\mathcal{R}_1 = \sum_i \left(\left\| \boldsymbol{J}_{n,i}^{\boldsymbol{p}} \right\|_2 - 1 \right)^2 .$$