# Validating Robotics Simulators on Real-World Impacts

Brian Acosta*, William Yang*, and Michael Posa

*Abstract*—**A realistic simulation environment is an essential tool in every roboticist's toolkit, with uses ranging from planning and control to training policies with reinforcement learning. Despite the centrality of simulation in modern robotics, little work has been done comparing robotics simulators against real-world data, especially for scenarios involving dynamic motions with high speed impact events. Handling dynamic contact is the computational bottleneck for most simulations, and thus the modeling and algorithmic choices surrounding impacts and friction form the largest distinctions between popular tools. Here, we evaluate the ability of several simulators to reproduce real-world trajectories involving impacts. Using experimental data, we identify system-specific contact parameters of popular simulators Drake, MuJoCo, and Bullet, analyzing the effects of modeling choices around these parameters. For the simple example of a cube tossed onto a table, simulators capture inelastic impacts surprisingly well, though generally fail to reproduce elasticity. For the higher-dimensional case of a Cassie biped landing from a jump, the simulators capture the bulk motion well but the accuracy is limited by model differences between the real robot and the simulators.**

*Index Terms*—**Contact Modeling, Simulation and Animation, Dynamics**

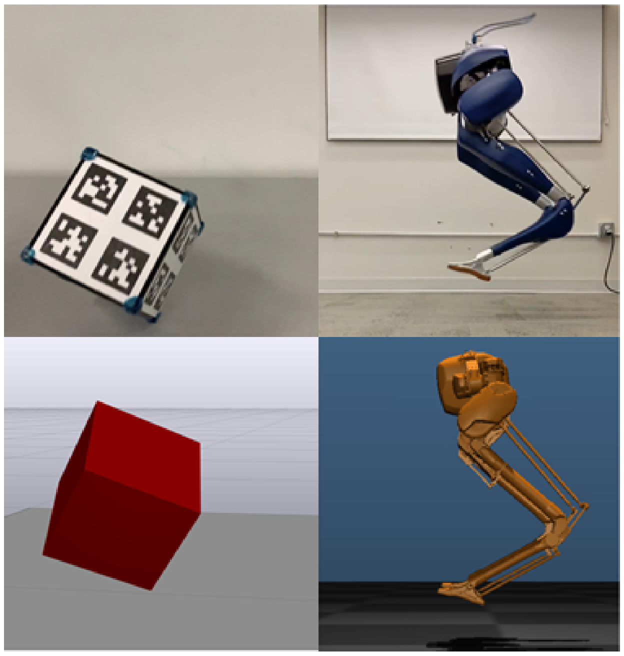
Fig. 1. Athletic behaviors such as jumping (right) involve high speed impacts which are difficult to model and simulate. Yet, optimal performance of controllers which are trained or verified in simulation relies on such impacts being faithfully captured by simulators. In this work, we evaluate simulators' ability to capture impact dynamics using real-world data from Cassie jumping and a cube tossed onto a wooden table.

## I. Introduction

Given the importance of simulation in planning and control, it is important to understand the physical realism of simulated impacts. Recent successes in sim-to-real reinforcement learning for legged locomotion [1], [2], [3] largely use domain randomization to mitigate model uncertainties. However, domain randomization works best when the range of randomized model parameters is small, and non-randomized dynamics are accurate. Contact parameters other than friction are rarely randomized, suggesting accurate impact simulation could be a contributing factor in successful sim-to-real transfer. Impact simulation accuracy also affects the ability to accurately compute regions of attractions for legged robot control [4] and verify the robustness of impact-aware control techniques [5], [6] without risking failure. Understanding which impact behaviors can be recreated in popular robotics simulators is an important step in applying these techniques to more impact-rich behaviors.

Most common robotics simulators use approximately rigid contact models [7], [8], with specific approximations chosen for the sake of computational speed and numerical stability. The physical realism of these contact models are largely validated by visual inspection or by tangential physical metrics [9], and there is evidence that rigid contact models sometimes poorly predict the dynamics of real collisions even for single impacts [10], [11]. Follow up work on improving the prediction ability of rigid body models with residual learning [12] demonstrates some success, though recent work suggests that stiff contact behavior may lead to poor amenability to learning based techniques [13].

While simulators may be able to accurately capture real impacts if tuned to a correct set of parameters, roboticists often choose contact parameters such as stiffness to maximize simulation speed and may only be peripherally aware of the tradeoffs being made with physical realism. As part of this paper, we examine the consequences of this approach by conducting a sensitivity analysis of simulators' physical realism to the contact parameters.

Manuscript received: February 24, 2022; Accepted April 24, 2022. This paper was recommended for publication by Editor H. Kurniawati upon evaluation of the Associate Editor and Reviewers' comments. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1845298. Toyota Research Institute also provided funds to support this work.

*These authors contributed equally. Names are in alphabetical order.

The authors are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA {bjacosta, yangwill, posa}@seas.upenn.edu

Digital Object Identifier (DOI): see top of this page.

The primary contributions of this work are:

- Empirical evaluation of multiple simulators on complex 3D impacts using real world data collected from cube tosses and jumping with a bipedal robot.
- Identification of optimal contact parameters for each system in Drake, MuJoCo, and Bullet and sensitivity analysis to assess the importance of properly specifying each parameter.
- Analysis of the relative strengths and weaknesses of each simulator when applied to the given scenarios.
- A publicly available dataset containing Cassie jumping trajectories.

## II. Background

The forward simulation of rigid body motion without contact is assumed to be identical across all three simulators. For this reason, this paper focuses on the predictive fidelity of simulator contact models when the system undergoes impact. Contact models define a mathematical relationship between the relative position and velocity of two rigid bodies and the contact force between them. Rigid contact models assume that bodies are perfectly rigid (i.e. infinite stiffness) and undergo inelastic collisions, which can be posed as a linear complementarity problem (LCP) [7]. Compliant models acknowledge that real objects experience some deformation during contact, and approximate the inter-body forces produced by deformation as a restoring force against the interpenetration of objects. The constitutive laws of compliant contact models are numerically stiff, requiring the use of small time steps, while solving LCPs is computationally expensive. Therefore, simulators take different approaches to improving simulation speed by choosing approximations and/or solution tactics. The following section details the contact model and solution strategy for each of the chosen simulators, which represent three distinct strategies for modeling and solving for contact forces.

### A. Simulator Contact Models

*1) Drake:* Drake [14] uses a compliant model of contact with Hunt & Crossley dissipation [15] which penalizes interpenetration using a nonlinear spring-damper law. Using a smooth approximation of Coulomb friction, Drake expresses contact forces as a function of state. Given a normal penetration distance $\delta$ and penetration rate $\dot{\delta}$, the normal contact force is

$$f_n = k(1 + b\dot{\delta})_+\delta_+, \tag{1}$$

where $k$ is the stiffness, $b$ is the dissipation, and $(\cdot)_+ = \max(0, \cdot)$ ensures positive normal forces. When incorporated into the equations of motion, this leads to a nonlinear system of equations. Drake uses the custom TAMSI solver to properly resolve contact transitions [16]. Like many other contact-model solvers, TAMSI uses an iterative algorithm; however, TAMSI enforces convergence at each time step. Resulting contact forces are therefore consistent with the original modeling equations of compliant contact with regularized friction.

*2) MuJoCo:* MuJoCo uses a convex approximation of rigid contact first introduced in [17] with regularization introduced in [18]. This regularization softens contact by relaxing the strict complementary of contact force and distance between objects. The amount of regularization is determined by user specified stiffness and damping parameters, which are transformed internally into reference accelerations for the contact constraint dynamics, which approximately obey the following linear spring-damper law [1]:

$$a_n \approx (-k\delta - b\dot{\delta})d(\delta) + (1 - d(\delta))a_0, \tag{2}$$

where $a_n$ is the normal acceleration, $a_0$ is the acceleration in the absence of contact, and $d(\delta) \approx 1$ is a position-dependent interpolation between the constrained and unconstrained acceleration. MuJoCo allows for tuning the shape of $d$, though the results presented here use the default values, as we find no improvement in the prediction error by tuning $d$. MuJoCo implements friction by solving a convex optimization problem which balances achieving the constraint dynamics in (2) with contact activation and energy dissipation.

This convex formulation guarantees a unique solution at each time step and yields fast, differentiable computations. This speed has made MuJoCo a popular simulator in the robotics community, especially among RL practitioners, though MuJoCo's contact constraint regularization has been observed to cause non-physical artifacts during slip. Objects can glide at a distance from each other [19], and large amounts of regularization can lead to viscous slip [20], requiring the use of additional stabilization techniques. Like Drake, after posing its regularized dynamics problem, MuJoCo is guaranteed to find an accurate solution to this problem.

*3) Bullet:* Bullet is a physics engine originally introduced for simulation in computer graphics and animation but widely used for robotics simulations. Bullet formulates the contact problem as a mixed LCP (MLCP), a generalization of the LCP which allows for inequality constraints. Bullet's MLCP represents rigid contact with Coulomb friction, and is solved using a Projected Gauss-Seidel (PGS) solver. PGS attempts to iteratively resolve each contact constraint separately, keeping the remaining constraints fixed. Unlike Drake or MuJoCo, Bullet's PGS solver is allowed to return with an intermediate computation when a fixed maximum number of iterations is reached. In this case, Bullet relies on linear spring-damper Baumgarte stabilization [21] to enforce the contact constraint. This feature can cause a lack of robustness in the event the PGS solver fails to converge.

### B. Simulator Considerations

The simulators evaluated in this paper are by no means exhaustive, with several other engines finding popularity in the robotics community. The three chosen here, however, provide good coverage of the different styles of contact dynamics formulations. We note that we considered IsaacGym [22], a popular simulator for reinforcement learning applications, including sim-to-real transfer on a legged robot [23]. However,

[1] https://mujoco.readthedocs.io/en/latest/modeling.html

at the time of publication, IsaacGym does not support modification of its compliance parameters. Without this capability, the accuracy of the IsaacGym simulator was quite poor on our dataset.

For the Cassie dataset, we limit our comparisons to Drake and MuJoCo. Bullet does not natively support modeling the reflected inertia from the motors and gearboxes, which plays a significant role in the dynamics of Cassie.

## III. Experimental Setup

### A. Cube Toss

*1) Data:* There are 550 trajectories of a 10 cm acrylic cube tossed onto a wooden table as a dataset for parameter identification and performance evaluation. The data collection procedure is described in [24] and the data is available as part of an open source code repository[2]. We name these the ground truth cube trajectories and denote a ground truth trajectory of length $T$ as $\boldsymbol{x}^* = (x_t^*)_{t=1\ldots T}$ where $x_t^* = [q_t^*; v_t^*]$ is the state of the cube at each time step. The configuration $q$ consist of position $p \in \mathbb{R}^3$ and orientation $R \in SO(3)$, and the velocity $v$ consists of linear and angular velocities $\dot{p}$ and $\omega$. Physical properties of the cube-table system are given in Table I. The cube was weighed to determine the mass, with the inertia determined from the measured mass and geometry. Friction was determined via a tilt-test, and restitution was determined by minimizing the prediction error in [24].

TABLE I
Cube Physical Parameters

| Mass | Inertia | Friction | Restitution |
|---|---|---|---|
| 0.37 kg | .0081 kg m$^2$ | 0.18 | 0.125 |

*2) Simulation Environment:* Each cube toss simulation is designed to match the real experiment as closely as possible. The measured dimensions and inertial parameters of the real cube are specified in a Universal Robot Description Format (URDF) file for Bullet and Drake, and as XML text in the MJCF format for MuJoCo. The timestep for each simulator is set to 1480 Hz, and the resulting trajectories are down-sampled to match the data collection frequency of 148 Hz. We found that decreasing the simulator timestep further did not improve the prediction capability of any simulator.

### B. Cassie Jumping

*1) Data Collection:* We use state and input data from 22 jumping experiments performed with the Cassie bipedal robot. The jumping trajectories were generated using the jumping controller detailed in [5], and are available as a public dataset[3]. The state measurements, sampled at 2000Hz, are composed of the joint positions and velocities as well as the floating base state of the pelvis. Although we treat the state data as ground truth, there is uncertainty in these measurements as a state estimator [25] is used to compute the floating-base state and joint velocities are subject to encoder noise and resolution. To account for state estimation errors, we offset the pelvis vertical height so that the feet are in contact with the ground at impact. This ensures that the impact timing between the real and simulated data matches. For context, the maximum correction applied is $0.02m$, which corresponds to approximately 20% of the total loss for both Drake and MuJoCo.

We use the same notation for Cassie trajectories as with the cube, though Cassie is modeled as a floating-base Lagrangian system with $q \in \mathbb{R}^{n+7}$ and $v \in \mathbb{R}^{n+6}$ where $n = 16$ is the number of joints. The motor input data, also sampled at 2000Hz, are the torques measured at the motors as opposed to the torques commanded by the controller. We make this distinction to account for the delays between the output of the controller and when the motor actually is supplied the commanded current.

*2) Simulation Environment:* Simulated data is generated from Drake and MuJoCo, and Bullet is not used on the Cassie data for reasons explained in II-B.

The description of the robot is specified in a URDF for Drake and as a XML for MuJoCo. For both simulators, we initialize the state of the robot just prior to impact and apply the measured motor torques at the corresponding times. This decision to simulate in open-loop eliminates the dependency on the controller. For both simulators, the state was sampled at 2000Hz, consistent with the hardware data.

There are slight differences in the Cassie models used by Drake and MuJoCo. The physical Cassie robot contains two four-bar linkages per leg that enable the control of leg length through the knee motor and the control of the toe joint through an actuator located at the ankle. The MuJoCo simulator provided by Agility Robotics [26] includes the achilles and plantar rods shown in Fig. 2 that form the loop closures for the linkages.

Drake does not natively support loop closure constraints, thus the URDF does not include the connecting rods but accounts for their inertial contribution by adding lumped masses at the anchor points. The upper loop closure is modeled as a stiff spring with spring constant chosen to enforce the loop closure but not interfere with the performance of the simulator. The lower loop closure is handled by applying actuator efforts directly at the toe joint, a common convention used in many Cassie models [27], [28].

While the state of the connecting rods are fully defined by the other states, the MuJoCo model includes these redundant states. For fair comparison, we map the full MuJoCo states to the Drake states when comparing state trajectories.

## IV. Parameter Identification

To evaluate the performance of each simulator, we identify a simulator and system-specific set of contact parameters that best enables each simulator to reproduce real-world trajectories. To maintain some comparability across simulators, we only tune the friction, stiffness, and damping parameters for each simulator. The parameters and their physical significance are given in Table II. Beyond differences in units, these parameters may not be physically equivalent, as the contact laws vary between simulators. We combine static and dynamic

[2]https://github.com/DAIRLab/contact-nets
[3]https://github.com/DAIRLab/cassie_impact_dataset

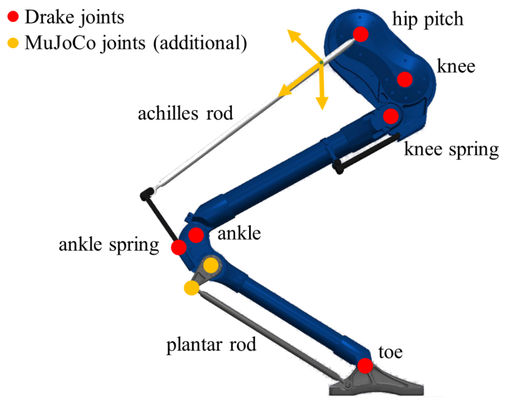

Fig. 2. The configuration states for a leg of the Cassie robot. The additional joints included in the MuJoCo model are indicated in yellow, which include the orientation of the achilles rod and the angles of the foot crank linkage.

friction into a single constant $\mu$ as we do not see higher accuracy from distinguishing between coefficients.

TABLE II
CONTACT PARAMETERS IDENTIFIED

| | Parameter | units | Physical Interpretation |
|---|---|---|---|
| **Drake** | $\mu$ | - | Friction coefficient |
| | $k$ | N / m | Contact stiffness |
| | $b$ | s / m | Contact dissipation |
| **MuJoCo** | $\mu$ | - | Friction coefficient |
| | $k$ | N / (kg m) | Contact stiffness |
| | $b$ | N s / (kg m) | Contact damping |
| **Bullet** | $\mu$ | - | Friction coefficient |
| | $k$ | N / m | Contact stiffness |
| | $b$ | N s / m | Contact damping |

## A. Evaluation Metrics

For a ground truth trajectory $\boldsymbol{x}^*$, the corresponding simulator trajectory $\hat{\boldsymbol{x}}(x_0^*, \theta) = (\hat{x}_t)_{t=1\dots T}$, is simulated with contact parameter vector $\theta$ from the initial condition $x_0^*$. We define the following error metrics, which will be used in the loss function for parameter identification. From here on we omit the dependence of $\hat{\boldsymbol{x}}$ on $x_0^*$ and $\theta$ for brevity.

*1) Cube Metrics:* The cube configuration error is

$$e_q(\boldsymbol{x}^*, \hat{\boldsymbol{x}}) = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{2}{l}\,\|p_t^* - \hat{p}_t\|_2^2 + \text{Angle}(R_t^*, \hat{R}_t)^2\right), \quad (3)$$

where $l$ is the side length of the cube and $\text{Angle}(R_1, R_2)$ is the angle of rotation of the relative rotation between $R_1$ and $R_2$. We scale the position error by $2/l$ to give identical units and similar magnitudes to position and orientation error. Since the velocity in the cube toss data set is generated by filtering differences of positions, and is therefore influenced by filter dynamics, we do not calculate velocity errors, focusing instead on long term position and orientation accuracy. Due to the second order dynamics of the cube, this will capture the effect of finding the correct contact dynamics while not biasing toward incorrect velocity estimates in the ground truth data.

*2) Cassie Metrics:* The joints on the Cassie robot have a wide range of inertias; therefore, a naive loss function, such as the L2-norm, would lead to over-weighting of the low-inertia joints such as the toes. To direct the parameter identification algorithm to prioritize capturing the bulk motion of the robot, we use a weighted norm

$$e_{cassie}(\boldsymbol{x}^*, \hat{\boldsymbol{x}}) = \sum_{t=1}^{T} \tilde{x}_t^T W \tilde{x}_t, \quad (4)$$

where $W$ is a diagonal matrix, and $\tilde{x}$ is the error between the simulated and measured state. The elements of $W$ are all 10 for the position indices. For the velocity indices, we use a weight of 5 for the floating base rotation, 100 for the floating base translations, 0.01 for the hip roll, knee spring, and toe joints, and 1 for the remaining joints. There are no measurements for the velocity of the ankle spring deflection, which is therefore omitted from the loss function.

The data indicates significant but unknown model differences between the simulators and the physical robot. These differences may include incorrect model parameter values for the spring constants and joint damping. Better modeling these effects would likely improve simulation accuracy, but the complexity of Cassie makes full system identification infeasible and is outside the scope of this paper.

To mitigate the effects of model uncertainty mentioned above, and to focus on capturing the impact event, we evaluate the trajectories for a brief 50 ms time window (T = 100 when sampled at 2000 Hz) around the impact event. This is under the assumption that the contact forces are large and the time window is short enough that error from incorrect model parameters will have a relatively small effect. To account for timing variations between jumping experiments, we manually select the time window for each log to include the first impact event for each jump.

## B. Optimization Procedure

TABLE III
CONTACT PARAMETER OPTIMIZATION DOMAINS

| | Parameter | $\Theta_{cube}$ | $\Theta_{cassie}$ |
|---|---|---|---|
| **All Simulators** | $\mu$ | [0, 1] | [0, 1] |
| **Drake** | $k$ | [1e2, 1e5] | [1e3, 1e6] |
| | $b$ | [0, 2] | [0, 3] |
| **MuJoCo** | $k$ | [1e2, 1e4] | [0,1e6] |
| | $b$ | [0, 1e3] | [0, 1e3] |
| **Bullet** | $k$ | [1e2, 1e4] | - |
| | $b$ | [0, 1e3] | - |

We identify contact parameters using NGOpt, a gradient-free meta-optimizer provided through the `nevergrad` Python library [29]. The appropriate contact parameters are the solution to the optimization problem

$$\theta^* = \arg\min_{\theta\in\Theta} L(\theta). \quad (5)$$

The optimization domains are shown in Table III. The loss functions for the cube and Cassie are

$$L_{cube} = \frac{1}{N}\sum_{i\in I} e_q(\boldsymbol{x}_i^*, \hat{\boldsymbol{x}}_i) \quad (6)$$

and

$$L_{cassie} = \frac{1}{N} \sum_{i \in I} e_{cassie}(\boldsymbol{x}_i^*, \hat{\boldsymbol{x}}_i), \tag{7}$$

where $I$ is the dataset and $N$ is the total number of trajectories in the dataset.

## V. Results

### A. Cube

*1) Parameters:* The optimal contact parameters $\theta^*$ for each simulator are reported in Table IV.

TABLE IV
Identified Cube Toss Parameters

| | Stiffness | Dissipation/ Damping | Friction |
|---|---|---|---|
| ***Drake*** | 10800 | 0.4 | 0.10 |
| ***MuJoCo*** | 3300 | 45 | 0.22 |
| ***Bullet*** | 9300 | 36 | 0.39 |

*2) Performance:* As shown in Table V, the simulators are all able to accurately reproduce cube toss trajectories, with Drake and Bullet being more accurate than MuJoCo. In addition to the minimum $e_q$ observed for each simulator, we report average position and rotation error along with standard deviation $\sigma$ for each metric.

TABLE V
Summary of Cube Errors

| | Position Err. $\pm\sigma$. (% Cube Width) | Rotation Err. $\pm\sigma$ (Degrees) | $e_q$ $\pm\sigma$ |
|---|---|---|---|
| ***Drake*** | 13.5 $\pm$ 8.2 | 16.5 $\pm$ 20.0 | 0.27 $\pm$ 0.57 |
| ***MuJoCo*** | 25.1 $\pm$ 10.8 | 21.7 $\pm$ 21.4 | 0.38 $\pm$ 0.63 |
| ***Bullet*** | 14.9 $\pm$ 8.9 | 16.5 $\pm$ 20.2 | 0.27 $\pm$ 0.57 |

### B. Cassie

The optimal contact parameters $\theta^*$ and the average loss for Drake and MuJoCo are reported in Table VI. Although we report a single set of optimal parameters, we observe that a wide range of parameters for both simulators achieve similar performance as shown in Fig. 3. The parameter set that works well for MuJoCo is a damping value near the optimal value, and depends very little on the stiffness parameter. Drake, on the other hand, has a band of optimal parameters that shows a clear relationship between stiffness and dissipation.

TABLE VI
Identified Cassie Jumping Parameters

| | Loss | Stiffness | Dissipation /Damping | Friction |
|---|---|---|---|---|
| ***Drake*** | 38.0 | 9400 | 3.0 | 0.45 |
| ***MuJoCo*** | 30.0 | 1600 | 270 | 0.43 |

## VI. Discussion

Drake, MuJoCo, and Bullet are all capable of recreating the bulk motion of the trajectories seen in our datasets. While the following section discusses potential areas for improvement, the salient point should be that modern robotics simulators are able to capture dynamic impacts with friction for a range of contact parameters.

### A. Cube

*1) Sources of Error:* Drake and Bullet have similar errors, as they both recreated near inelastic collision observed in the cube toss system, though they produced less restitution than observed in the real system. For most tosses this resulted in a minor contribution to the average position error, though occasionally caused the simulated cube to slide rather than bounce and settle on a different face (Figure 4). High rotation error in these instances is responsible for the large standard deviations in Table V. MuJoCo exhibited more elastic behavior, though this was accompanied by softer impacts and larger penetration depths (Figure 5).

*2) Simulator Comparison:* As seen in Table V, MuJoCo was the least accurate in matching cube trajectories. Due to the interplay between friction and contact dynamics, MuJoCo produces longer contact events with lower peak forces, dissipating energy more slowly than the other simulators. While this effect can be tuned out for individual logs, we found no parameter change which was able to improve the average $e_q$ across all logs.

### B. Cassie

The velocities of the primary load-bearing joints (hip pitch, knee, and ankle), are captured well by both simulators. As a result, the vertical velocity of pelvis, which is the combination of the load-bearing joints, is similarly captured as shown in Fig. 6. This is surprising, because the simulators are not only able to predict the velocity at the end of the time window, but also able to model the rate at which the velocity changes as well. This is promising evidence in support of the use of simulators to evaluate the performance of controllers *during* the impact event, when the impact has not fully resolved.

Certain logs, where Cassie landed distinctly with the rear parts of the feet first, have significantly high losses across all contact parameters. The additional error is attributed to a twisting motion at the pivot points which caused the orientation of the pelvis to shift quickly. While this twisting motion is present in those actual trajectories, the actual motion was not as severe. This may be due to the contact patches on the physical robot able to exert moments about the contact normal, whereas in both simulators, contact forces are represented as point contacts and unable to produce moments about the contact point.

### C. Sensitivity Analysis

In addition to identifying the parameters for each simulator and system, we also perform a sensitivity analysis of the parameters by sweeping each parameter value individually, holding the remaining values at $\theta^*$. The results of this analysis are shown in Fig. 3.

*1) Stiffness and Damping:* It is surprising that MuJoCo appears to be insensitive to small stiffness values, given that stiffness is responsible for enforcing non-penetration. Our hypothesis is that, in the small time window around the impact event, the damping forces provide sufficient contact force for our datasets due to the large impact velocity.

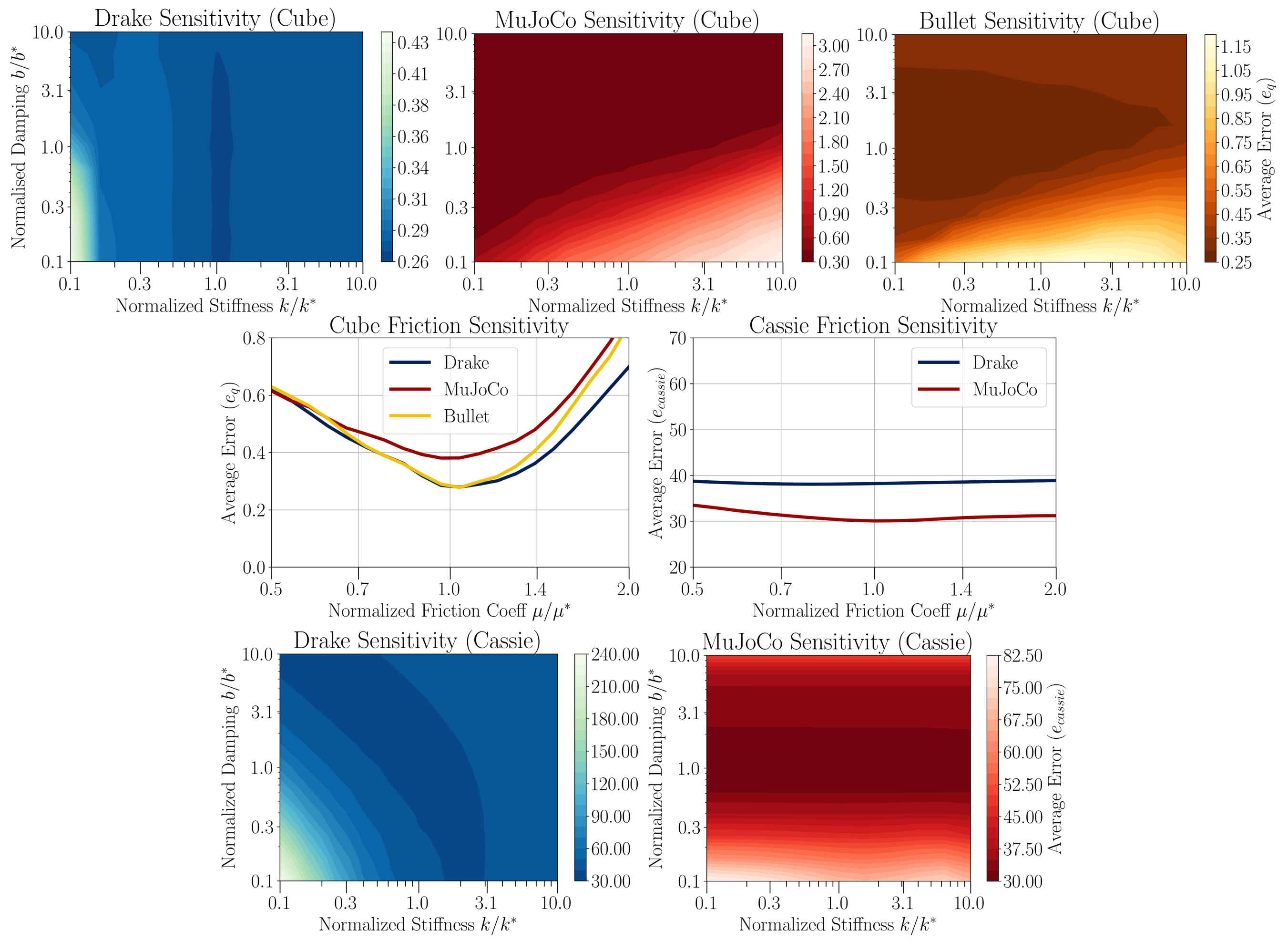

Fig. 3. We perform a sensitivity analysis by holding friction fixed at the optimal value while sweeping a range of stiffness and damping values, and vice-versa. ***Cube Toss*** *(Top Row):* All three simulators are mostly insensitive to stiffness above a given threshold, while MuJoCo and Bullet require sufficient damping as well. ***Friction*** *(Middle):* Cube toss error is sensitive to the friction coefficient parameter due to large amounts of sliding in the dataset, while Cassie jumping is not, due to mostly experiencing static friction. ***Cassie Jump*** *(Bottom):* Drake and MuJoCo display relatively low sensitivity to the contact parameters and have a wide range of parameter values that achieve low error. MuJoCo shows almost no sensitivity to its stiffness parameter.

For the cube toss dataset, the sensitivity of Bullet and MuJoCo to insufficient damping, and the correlation of this dependence with the amount of stiffness, suggests that there is an optimal damping ratio for MuJoCo's soft contact dynamics and Bullet's constraint stabilization for this system. Note that this is not the real damping ratio of the contact dynamics, due to trade-offs between friction and contact dynamics in MuJoCo, and the fact that Bullet only uses a spring-damper law to stabilize the simulation of rigid contact.

*2) Friction:* The cube toss prediction error is sensitive to the coefficient of friction, which is unsurprising given the large amount of sliding contact in the dataset. Perhaps more surprising is that every simulator is so sensitive to the friction coefficient while not necessarily being close to the experimentally measured value in Table I. Frictional impacts are difficult to accurately model [30] [31], and the three contact modeling paradigms explored here may need different values to achieve low prediction error and account for the un-modeled effects of each framework.

The Cassie trajectories did not enter the slip-regime for friction, which is necessary to characterize the friction coefficient. This is why the sensitivity to the friction coefficient is relatively flat.

## VII. Conclusions

We observe that for both systems, the simulators tested are able to reproduce the bulk motion observed during impact. This suggests that simulators can indeed be an appropriate tool for controller design and verification in impact-rich settings. Additionally, we observe that accurate simulator performance can be achieved by a wide range of contact parameters, which suggests that extensive identification of the contact parameters is not necessary.

While the simulators perform well on a significant portion of the cube dataset, there were several cube tosses that all simulators struggle on. We were unable find any property that nicely distinguishes these tosses from the rest of the dataset, though we noted that these tosses often involve frictional impact with an edge or corner of the cube. We believe that further investigating and characterizing these particular cases can result in improved simulator performance.

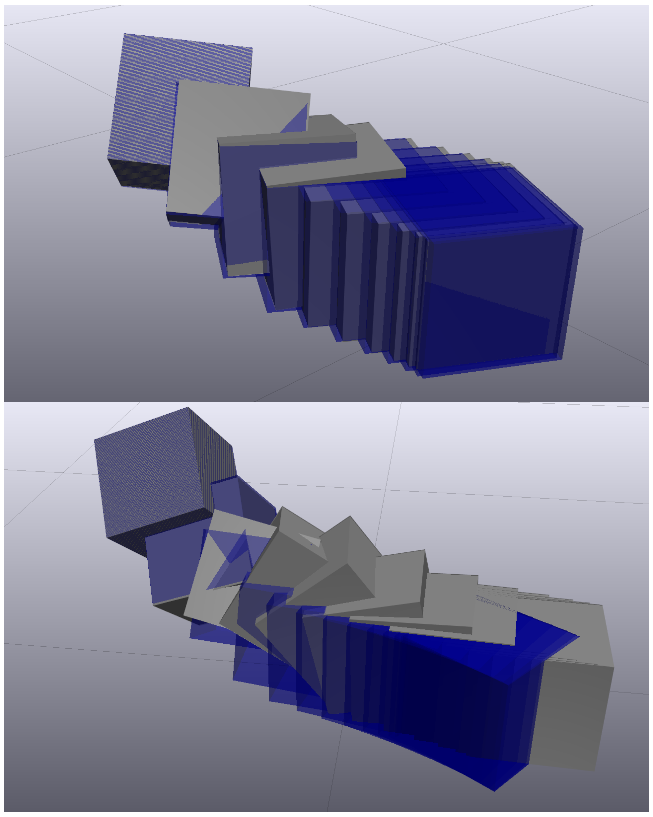

Fig. 4. Motion patterns showing the real cube (grey) and the simulated cube (blue) for an inelastic (top) and elastic (bottom) collision in Drake.

Although the identified cube parameters are fairly stiff, the identified parameters for Cassie are much softer. This may be due to the rubber pads on Cassie's feet as well as other mechanical compliance associated with a larger system. This compliance results in impacts resolving over tens of milliseconds, which is important to model when using simulators for validation or in learning situations.

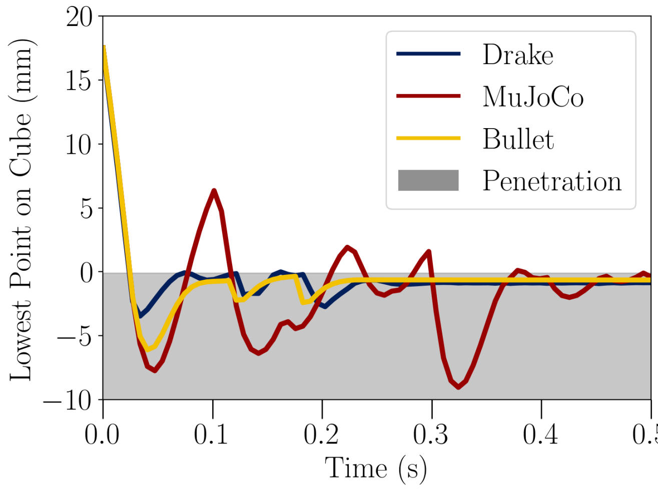

Fig. 5. The lowest point on the cube for the three simulators on a typical toss, simulated using the optimal parameters for each simulator. When the lowest point of the cube is below the ground (<0), the system is in penetration. Hand-tuning MuJoCo's damping parameter can reduce the "bouncing" for this particular toss, but as seen in Figure 3, there is no parameter change which could improve the MuJoCo's performance across all logs.

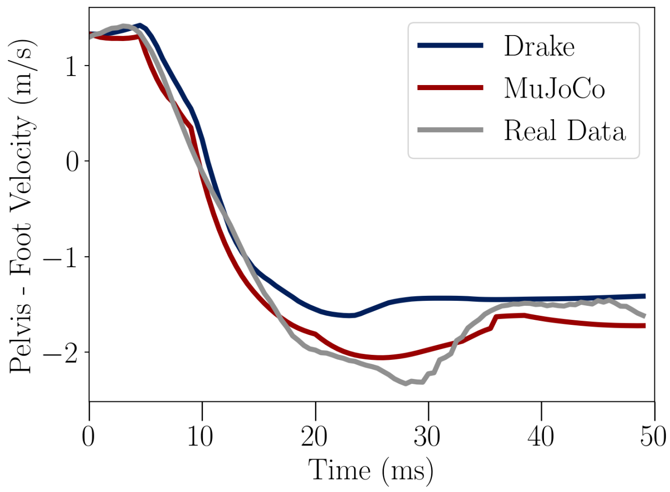

Fig. 6. The vertical pelvis velocity of Cassie with respect to its feet for the 50ms duration. In this direction, which is composed from the combined kinematics of the load-bearing joints, both simulators do a remarkable job of capturing the real data. This level of agreement is present in many of the tested logs.

## Acknowledgements

We thank Mathew Halm for helpful discussions and assistance with the cube-toss dataset and Alejandro Castro for his insight on simulator contact models.

## References

[1] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 7309–7315.

[2] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid Motor Adaptation for Legged Robots," *arXiv:2107.04034 [cs]*, July 2021, arXiv: 2107.04034. [Online]. Available: http://arxiv.org/abs/2107.04034

[3] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. Panne, "Learning locomotion skills for cassie: Iterative design and sim-to-real," in *Conference on Robot Learning*. PMLR, 2020, pp. 317–329.

[4] W. Ubellacker, N. Csomay-Shanklin, T. G. Molnar, and A. D. Ames, "Verifying safe transitions between dynamic motion primitives on legged robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8477–8484.

[5] W. Yang and M. Posa, "Impact Invariant Control with Applications to Bipedal Locomotion," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[6] Y. Gong and J. W. Grizzle, "One-step ahead prediction of angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-inspired controller," in *IEEE International Conference on Robotics and Automation (ICRA), 2021*, 2021, pp. 2832–2838.

[7] D. E. Stewart and J. C. Trinkle, "An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction," *International Journal for Numerical Methods in Engineering*, vol. 39, no. 15, pp. 2673–2691, 1996.

[8] M. Anitescu and F. A. Potra, "Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems," *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.

[9] T. Erez, Y. Tassa, and E. Todorov, "Simulation tools for model-based robotics: Comparison of Bullet, Havok, MuJoCo, ODE and PhysX," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 4397–4404, iSSN: 1050-4729.

[10] N. Fazeli, E. Donlon, E. Drumwright, and A. Rodriguez, "Empirical evaluation of common contact models for planar impact," in *2017 IEEE*

*International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3418–3425.

[11] A. Chatterjee, "On the Realism of Complementarity Conditions in Rigid Body Collisions," *Nonlinear Dynamics*, vol. 20, no. 2, pp. 159–168, Oct. 1999. [Online]. Available: https://doi.org/10.1023/A:1008397905242

[12] N. Fazeli, A. Ajay, and A. Rodriguez, "Long-Horizon Prediction and Uncertainty Propagation with Residual Point Contact Learners," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 7898–7904, iSSN: 2577-087X.

[13] M. Parmar, M. Halm, and M. Posa, "Fundamental Challenges in Deep Learning for Stiff Contact Dynamics," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

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

[15] K. H. Hunt and F. R. E. Crossley, "Coefficient of Restitution Interpreted as Damping in Vibroimpact," *Journal of Applied Mechanics*, vol. 42, no. 2, pp. 440–445, June 1975. [Online]. Available: https://doi.org/10.1115/1.3423596

[16] A. M. Castro, A. Qu, N. Kuppuswamy, A. Alspach, and M. Sherman, "A transition-aware method for the simulation of compliant contact with regularized friction," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1859–1866, 2020.

[17] M. Anitescu, "Optimization-based simulation of nonsmooth rigid multibody dynamics," *Mathematical Programming: Series A and B*, vol. 105, no. 1, pp. 113–143, Jan. 2006. [Online]. Available: https://doi.org/10.1007/s10107-005-0590-7

[18] E. Todorov, "Convex and analytically-invertible dynamics with contacts and constraints: Theory and implementation in MuJoCo," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 6054–6061, iSSN: 1050-4729.

[19] H. Mazhar, D. Melanz, M. C. Ferris, and D. Negrut, "An Analysis of Several Methods for Handling Hard-Sphere Frictional Contact in Rigid Multibody Dynamics," in *ECCOMAS Thematics Conference on Multibody Dynamics*, 2015.

[20] "SimBenchmark. Physics engine benchmark for robotics applications: RaiSim vs. Bullet vs. ODE vs. MuJoCo vs. DartSim." [Online]. Available: https://leggedrobotics.github.io/SimBenchmark

[21] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 1, no. 1, pp. 1–16, 1972. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0045782572900187

[22] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[23] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 91–100.

[24] S. Pfrommer, M. Halm, and M. Posa, "Contactnets: Learning of discontinuous contact dynamics with smooth, implicit representations," *Conference on Robotic Learning (CoRL)*, 2020.

[25] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[26] Agility Robotics, "cassie-mujoco-sim," 2018. [Online]. Available: https://github.com/osudrl/cassie-mujoco-sim

[27] J. Reher, W.-L. Ma, and A. D. Ames, "Dynamic Walking with Compliance on a Cassie Bipedal Robot," in *2019 18th European Control Conference (ECC)*, June 2019, pp. 2589–2595.

[28] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, "Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 4559–4566.

[29] J. Rapin and O. Teytaud, "Nevergrad - A gradient-free optimization platform," https://GitHub.com/FacebookResearch/Nevergrad, 2018.

[30] M. Payr and C. Glocker, "Oblique frictional impact of a bar: analysis and comparison of different impact laws," *Nonlinear Dynamics*, vol. 41, no. 4, pp. 361–383, 2005.

[31] M. Halm and M. Posa, "Modeling and analysis of non-unique behaviors in multiple frictional impacts," in *Robotics: Science and Systems*, 2019.